Universidade do Porto

Faculdade de Engenharia

FEUP

Building Volumetric Maps with Cooperative Mobile Robots and Useful Information Sharing

A Distributed Control Approach based on Entropy

Ph.D. Thesis

Rui Paulo Pinto da Rocha
M.Sc.

Porto – Portugal
October 2005

A thesis submitted to the University of Porto in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical and Computer Engineering.

Porto, Portugal, October 2005.

Supervisors:

Doctor Adriano da Silva Carvalho
Associate Professor

Dep. of Electrical and Computer Engineering
Faculty of Engineering
University of Porto

Doctor Jorge Manuel Miranda Dias
Associate Professor

Dep. of Electrical and Computer Engineering
Faculty of Sciences and Technology
University of Coimbra

Thesis Committee:

Doctor Jorge Leite Martins de Carvalho, FEUP (Chairman)
Doctor Wolfram Burgard, Albert-Ludwigs-Universität Freiburg, Germany
Doctor Maria Isabel Lobato de Faria Ribeiro, Instituto Superior Técnico, Lisboa
Doctor José Manuel Araújo Baptista Mendonça, FEUP
Doctor Paulo José Cerqueira Gomes da Costa, FEUP
Doctor Jorge Manuel Miranda Dias, University of Coimbra (Supervisor)
Doctor Adriano da Silva Carvalho, FEUP (Supervisor)

Approved by unanimity on May 8, 2006.

In the first two years (1999–2000), the research work reported herein was funded by the Ph.D. scholarship PRAXIS XXI/BD/18518/98, granted by the Portuguese Government's department FCT — Fundação para a Ciência e a Tecnologia, under the European Union's Third Support Framework (FSE, III Quadro Comunitário de Apoio).

To my adorable wife Inês
and to my parents.

"A ladder's step is not simply for someone remaining on it, but for sustaining one of the feet of a man for the sufficient time, until he puts his other foot slightly higher."

Thomas Huxley, biologist, 1825–1895.

Acknowledgments

This thesis is the result of a long, challenging and, sometimes, painful walk that I have done for the past six years. Nine years ago, after my graduation in Electrical and Computers Engineering, at the University of Porto, I started to feel my vocation for an academic career and, in particular, for scientific research. In spite of the many difficulties that I have found during my research work towards this Ph.D. thesis, I have also found pleasure on doing this walk. That vocation was certainly determinant on giving to me the necessary strength to overcome many difficulties, but it would be not sufficient without the support of many people, to whom I will be grateful for the rest of my life.

I would like to thank to Professor Jorge Dias, one of my supervisors, for his qualified and tireless support for the last five years, since I moved from Porto to Coimbra. At that time, although we did not know each other, he received me in a very friendly way and we started to work together at ISR Coimbra, due to our common interest on mobile robotics. Since then, he has introduced me to almost every colleagues in Coimbra and has been my protector. I have learned a lot about robotics with him, thanks to his long and internationally recognized contributions to this research field, and to the many valuable scientific discussions that we have had in the course of the research work herein reported. It was also him that motivated me to start working on cooperative robotics.

To Professor Adriano Carvalho, who has been also my supervisor, I would like to thank for his wisdom, encouragement, friendly support, and endless availability to me. I am sure he was the seed for my academic vocation, particularly in the field of mobile robotics, when I was still a graduate student at University of Porto. For this reason, I will always consider him as my “academic father”.

To the ISR Coimbra — Institute of Systems and Robotics of Coimbra —, and particularly to its director Professor Traça de Almeida, I would like to thank for the valuable resources made available to me, which allowed to pursue my research.

To Professor Isabel Ribeiro from IST, Lisbon, I would like to thank for the opportunity she gave to me of attending the interesting EURON Summer School on Cooperative

Robotics, which was held in IST, in September 2002. In that occasion, I knew other Ph.D. students and many international skillful researchers on cooperative robotics, such as Ron Arkin, Manuela Veloso, Rachid Alami and Milind Tambe.

Thanks to the Naval Research Laboratory, Washington DC, U.S.A., and particularly to Doctor Alan Schultz, for funding my attendance to the Multi-Robot Systems Workshop, in March 2003. This high level scientific meeting provided me with an excellent background about research on multi-robot systems carried out in the United States, with a special emphasis on its application to military domains.

To the many anonymous reviewers of my papers, thanks for the constructive comments.

Before February 2000, until I started to work at the University of Coimbra as a Teaching Assistant, the research work reported herein was funded by a Ph.D. scholarship granted to me by the Portuguese Government's department FCT — Fundação para a Ciência e a Tecnologia. This official support was crucial to start doing this Ph.D.

Many thanks to Luis Almeida for the gratuitous help he gave to me when I started to use stereo-vision sensors in the Scout mobile robots. Thank you very much for the many hours you spent with me, trying to configure a Linux driver for the frame grabber. I would like also to thank to other colleagues of Mobile Robotics Lab of ISR Coimbra — Jorge Lobo, João Alves, Alberto Valejo, João Filipe, Carlos Simplício, Filipe Ferreira, Paulo Menezes, Jöerg Rett, Rui Bernardes, Luiz Mirisola, *etc.*— for their outstanding support. Thank you for the many interesting discussions we had, especially during our internal lectures.

Last, but not the least, I wish to express my deepest gratitude to my close family, including my grandmother, uncles, parents-in-law, sisters-in-law, and brothers-in-law, for the constant support. I would like to thank to my sister for revising the translation of the thesis' abstract to French. I would like also to thank in particular to my parents, who have always motivated me to face this challenge and have given to me all their endless love since I exist. And, most of all, I offer my deepest thanks to my adorable wife Inês for her love, and for always helping me to overcome *those* difficult moments, when I was truly depressed. You have always believed that I would get done this thesis; even when I did not believe. I will never forget it.

Porto, October 2005,

Rui Paulo Pinto da Rocha

Building Volumetric Maps with Cooperative Mobile Robots and Useful Information Sharing

A Distributed Control Approach based on Entropy

by

Rui Rocha

Thesis submitted to the University of Porto on October 2005, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical and Computer Eng.

Abstract

This thesis addresses the problem of how to share efficiently information within a robotic system comprised of several mobile robots, which are programmed to exhibit cooperative behavior in the context of building volumetric maps of unknown environments. More specifically, it addresses the following issues: representing a probabilistic map and improving it through efficient exploration, based on information gain maximization; distributed control of teams of cooperative mobile robots, based on an information utility criteria; and coordinated exploration, aiming at avoiding redundant sensory information and robots' interference.

Robots have been developed essentially to help or substitute humans in tasks which are either repetitive or dangerous. For many of these tasks, especially those that are intrinsically distributed and complex, a team of several cooperative mobile robots — a cooperative multi-robot system (MRS) — is required to either make viable the mission accomplishment or, at least, accomplish the mission with better performance than a single mobile robot.

In spite of potential advantages related with space distribution, time distribution, complex problems decomposition, robustness, reliability and cost, a MRS requires that each robot maintains a sufficient and consistent level of awareness about the mission assigned to the team and about its teammates, in order to attain effective cooperation. The main challenge is that information is distributed and thus each robot has only partial and, sometimes, inconsistent knowledge about the environment. *Sharing* efficiently information via communication is thus crucial for robots' cooperation.

Building maps is indeed a relevant robotics' application domain. Firstly, in many other application domains than building maps (*e.g.* search and rescue, surveillance, planetary exploration, *etc.*), a robot usually needs a map to support safe and efficient navigation based on a world model. Secondly, robots may substitute humans on building detailed models, such as: fastidious maps of indoor environments (*e.g.* buildings); maps of buried utilities (*e.g.* gas pipes, power or communication lines, *etc.*), in order to avoid getting too close to them in construction activities; or detailed maps of hazardous environments (*e.g.* abandoned underground mines, nuclear facilities, *etc.*), in order to support monitoring or maintenance procedures.

The contributions of this thesis include a compact grid-based probabilistic representation model of a volumetric map, which allows to explicitly model uncertainty through the entropy concept. A frontier-based exploration method is also formulated using entropy, so that each robot uses its current map to select a new exploration viewpoint with maximum information gain.

This probabilistic framework is used to devise a distributed architecture model for building volumetric maps with teams of cooperative mobile robots, whereby each robot is altruistically committed to share useful measurements with its teammates. The information sharing is based on the formal definition of a measure of information utility, developed upon the concepts of entropy and mutual information, whereby sensory data is as useful as it contributes to improve the robot's map. The distributed architecture is further refined with a mechanism to coordinate the exploration actions of different robots, thus improving the team's performance.

The proposed methods were implemented in mobile robots, equipped with wireless communication and a stereo-vision system providing range measurements. These robots were used to carry out a set of experiments in a physical environment, which successfully validated the proposed framework and demonstrated the performance improvement yielded by the robots' cooperation. The results obtained with mobile robots were complemented with extensive computer simulations, which demonstrated those methods with varying team sizes.

Keywords: Cooperation, mobile robotics, volumetric maps, entropy, information utility, mutual information.

Construção de Mapas Volumétricos com Robôs Móveis Cooperantes e Partilha de Informação Útil

Uma abordagem de Controlo Distribuído baseada em Entropia

por

Rui Rocha

Tese submetida à Universidade do Porto em Outubro de 2005, para satisfação parcial dos requisitos do grau de Doutor em Engenharia Electrotécnica e de Computadores.

Resumo

Esta tese aborda o problema da partilha eficiente de informação no seio de um sistema robótico composto por vários robôs móveis, os quais são programados para obter um comportamento cooperante no contexto da construção de mapas volumétricos de ambientes desconhecidos. Mais especificamente, são abordados os assuntos seguintes: representação de um mapa probabilístico e sua construção incremental, através da exploração eficiente do ambiente baseada na maximização do ganho de informação; controlo distribuído de equipas de robôs móveis cooperantes, baseado num critério de utilidade da informação; e coordenação da exploração, com o objectivo de se evitarem a aquisição de informação sensorial redundante e a interferência entre robôs.

Os robôs têm sido desenvolvidos essencialmente para ajudar ou substituir pessoas em tarefas que sejam repetitivas ou desempenhadas em ambientes perigosos. Muitas destas tarefas, particularmente aquelas que são intrinsecamente distribuidas e complexas, requerem a utilização de uma equipa de vários robôs móveis cooperantes — um sistema de múltiplos robôs cooperantes (SMRC) — de forma a torná-las viáveis nalguns casos ou, noutros casos, de forma a obter um desempenho melhor do que com um único robô.

Apesar de apresentar vantagens potenciais relacionadas com distribuição espacial, simultaneidade, decomposição de problemas complexos, robustez, fiabilidade e custo, um SMRC requer que cada robô mantenha um nível de conhecimento suficientemente completo e consistente, acerca da missão atribuída à equipa em que se encontre integrado e acerca dos outros robôs da equipa, para que a cooperação entre os robôs se torne efectiva. O principal desafio prende-se com o facto da informação estar dispersa e pelo facto de cada robô possuir apenas um conhecimento incompleto, e muitas vezes inconsistente, acerca do ambiente. A *partilha* eficiente de informação através de comunicação inter-robôs é, portanto, crucial para a existência de cooperação.

A construção de mapas é de facto um domínio de aplicação relevante para a robótica. Por um lado, em muitos outros domínios de aplicação em que o mapa não é o fim em si mesmo (*ex.* busca e salvamento, vigilância, exploração planetária, *etc.*), cada robô precisa habitualmente de manter um mapa que sirva de suporte à navegação segura e eficiente através do ambiente. Por outro lado, os robôs podem substituir as pessoas na obtenção de mapas detalhados, tais como: mapas fastidiosos de ambientes interiores (*ex.* edifícios); mapas de infra-estruturas enterradas no solo (*ex.* condutas de gás, cabos eléctricos, linhas de comunicação, *etc.*), para se evitar a sua danificação durante trabalhos de construção civil; ou mapas detalhados de ambientes perigosos (*ex.* explorações mineiras abandonadas, instalações nucleares, *etc.*), que sirvam de suporte à sua monitorização ou manutenção.

As contribuições da tese incluem um modelo probabilístico de representação compacta de mapas volumétricos, baseado em grelhas, que permite modelizar explicitamente a incerteza através do conceito de entropia. Também é formulado um método de exploração *frontier-based* usando entropia, segundo o qual cada robô utiliza o seu mapa actual para seleccionar uma nova pose de exploração que maximize o ganho de informação.

Estes métodos probabilísticos servem de base ao desenvolvimento de uma arquitectura distribuída para a construção de mapas volumétricos com equipas de robôs móveis cooperantes, pela qual cada robô partilha altruisticamente com os outros robôs da equipa as medidas de distância mais úteis. Esta partilha de informação é baseada na definição formal de uma medida de utilidade de informação, desenvolvida a partir dos conceitos de entropia e informação mútua, segundo a qual a informação sensorial é tanto mais útil quanto mais contribui para melhorar o mapa do robô. A arquitectura distribuída é ainda dotada de um mecanismo para coordenar as acções de exploração levadas a cabo pelos diferentes robôs, melhorando assim o desempenho global da equipa.

Os métodos propostos foram implementados em robôs móveis, dotados de comunicação sem fios e de um sistema de visão binocular utilizado para medir distâncias. Os robôs foram usados para levar a cabo um conjunto de experiências num ambiente físico, que demonstraram a validade dos métodos e a melhoria de desempenho proporcionada pela cooperação entre robôs. Os resultados obtidos com os robôs móveis foram complementados com a realização de numerosas simulações em computador, que demonstraram a utilização daqueles métodos em equipas de diversos tamanhos.

Palavras chave: Cooperação, robótica móvel, mapas volumétricos, entropia, utilidade da informação, informação mútua.

Construction des Cartes 3-D avec des Robot Mobiles Coopératives et Échange d'Information Utile

Une Approche de Contrôle Distribué basé en Entropie

par

Rui Rocha

Thèse soumise à l'Université de Porto en Octobre de 2005, pour la satisfaction partielle des conditions du degré de Docteur en Génie Électrique et Informatique.

Résumé

Cette thèse aborde le problème de l'échange efficace d'information au sein d'un système robotique composé par plusieurs robots mobiles, qui sont programmés pour obtenir un comportement coopérative dans le contexte de la construction de cartes volumétriques d'environnements inconnus. Plus spécifiquement, on aborde les matières suivantes : la représentation d'une carte probabiliste et sa construction à travers de l'exploration efficace de l'environnement, basé sur la maximisation du gain d'information ; le contrôle distribué d'équipes de robots mobiles coopératives, basé sur un critère d'utilité de l'information ; et coordination de l'exploration avec l'objectif d'éviter l'acquisition d'information sensorielle redondant et l'interférence entre robots.

Les robots vienne a être développés essentiellement pour assister ou substituer des personnes pour des tâches qui sont répétitives ou exécutés dans des environnements dangereux. Plusieurs de ces tâches, particulièrement celles que sont intrinsèquement distribués et complexes, demandent l'utilisation d'un système de multiples robots coopératives (SMRC), pour assurer la viabilité de l'accomplissement de ces tâches, ou pour obtenir une performance meilleur que un robot singulier.

Nonobstant les avantages potentielles présentées, comme distribution spatial et temporel, décomposition des problèmes complexes, robustesse, fiabilité et coût, un SMRC demande que chaque robot maintienne un niveaux de connaissance suffisamment complet et consistant, au sujet de la mission attribué à l'équipe où il est intégré et au sujet de l'autres robots de l'équipe, pour que la coopération entre les robots soit effective. Le principal défi est le fait de l'information être dispersée pour les différents robots et, aussi, le fait de chaque robot avoir seulement une connaissance incomplète et, plusieurs fois, inconsistante au sujet de l'environnement. L'*échange* efficace d'information a travers de communication inter-robots est donc cruciale pour l'existence de coopération.

La construction des cartes est de fait un domaine d'application importante pour la robotique. D'un coté, il y a d'autres domaines pour lesquels la carte n'est pas l'objectif principal (*ex.* recherche et sauvetage, vigilance, exploration planétaire, *etc.*), mais où chaque robot demande fréquemment de maintenir une carte pour aider la navigation sécurisante et efficiente a travers l'environnement. Aussi important, les robots peuvent remplacer les personnes dans l'obtention des cartes détaillés, comme : cartes fastidieuses d'environnements intérieures (*ex.* édifices) ; cartes d'infrastructures enterrés dans le sol (*ex.* conduites des gaz, câbles électriques, lignes des communications, *etc.*), pour éviter son endommagement pendant le travails de la construction civile ; ou cartes détaillés des environnements dangereux (*ex.* mines abandonnées, installations nucléaires, *etc.*) qui serve de support pour son accompagnement et maintenance.

Les contributions de cette thèse inclue un model probabiliste de représentation compacte des cartes volumétriques, basé en grils, qui permette de modéliser explicitement l'incertitude a travers du concept d'entropie. Aussi, on proposé une méthode d'exploration *frontier-based* utilisant l'entropie, q'un robot peut utiliser pour choisir une nouvelle pose d'exploration a partir de sa carte actuelle, avec l'objectif de maximiser le gain d'information.

Ces méthodes probabilistes sont la base pour le développement d'une architecture distribué, pour construire des cartes volumétriques avec des équipes de robots mobiles coopératives, qui échangent par altruisme les donnés sensorielles plus utiles avec l'autres robots de l'équipe. Cet échange d'information est basé sur une mesure formelle d'information utile, développée à partir des concepts d'entropie et d'information mutuelle, qui détermine que l'information plus utile est celle qui contribuait plus pour la qualité de la carte. L'architecture a aussi un mécanisme pour coordonner les actions d'exploration exécutés par les différentes robots, et donc améliore la performance globale de l'équipe.

Las méthodes proposées ont été mis en application sur des robots mobiles, équipés avec une communication sens fils et un système de vision stéréo pour mesurer des distance. Les robots ont été utilisés pour exécuter en ensemble d'expériences dans un environnement physique. Ces expériences ont démontré la validité des méthodes et l'amélioration de la performance due à la coopération entre robots. Les résultats obtenus ont été complémentés avec la réalisation de plusieurs simulations en ordinateur, qui ont démontré l'utilisation de celles méthodes en équipes avec un nombre variable de robots.

Mots clés : Coopération, robotique mobile, cartes volumétriques, entropie, utilité de l'information, information mutuelle.

Contents

List of Figures

List of Tables

Notation

Symbol	**Description**
General notation	
a, $a()$	Scalar, scalar function
$\mathbf{a}$	Bold lower case letters represent vectors (usually 3-D column vectors)
$\mathbf{A}$	Bold upper case letters represent matrices
$\overrightarrow{\mathbf{a}}$	Applied vector
$\hat{\mathbf{a}}$	Unitary vector
A	Upper case letters may represent tuples or random variables
$\mathcal{A}$	Upper case letters written in calligraphy represent sets
Probabilities and entropy-related notation	
$p(x)$	Probability distribution / probability density function (pdf) of X
$p(x,y)$	Joint probability distribution / pdf of X and Y
$p(x \mid y)$	Conditional probability distribution / pdf of X given Y
$H(X)$	Discrete (marginal) entropy of the discrete random variable X
$h(X)$	Differential (marginal) entropy of the continuous random variable X
$H(X,Y)$	Joint entropy of the random variables X and Y
$H(X \mid Y)$	Conditional entropy of X given Y
$D(p\\|q)$	Kullback Leibler distance between the probability distrib. p and q
$I(X;Y)$	Mutual information between X and Y
$I(X;Y \mid Z)$	Mutual information between X and Y given Z
X^{Δ}	Quantized version of the continuous random variable X
$H(\mathcal{X})$	Joint entropy of the set of discrete random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$
$I(\mathcal{X};\mathcal{Y})$	Mutual information between sets $\mathcal{X}$ and $\mathcal{Y}$
$N(\mu,\sigma)$	Gaussian pdf with mean μ and standard deviation σ

Symbol	Description
Main symbols and functions	
$\mathcal{F}$	Fleet of n robots $\{1, \ldots, n\}$
t_k	Time instant when the k-th batch of measurements is obtained, $k \in \mathbb{N}$
t_0	Initial time instant before any measurements, with $t_0 \leq t_k,\ \forall_{k \in \mathbb{N}}$
$\mathcal{T}$	Set of time instants $\{t_k : t_k \in \mathbb{R},\ k \in \mathbb{N}_0,\ t_{k-1} \leq t_k,\ \forall_{k \in \mathbb{N}}\}$
$\{W\}$	Global coordinates reference frame
$\mathbf{x}(t)$	Robot's (sensor) position $[x, y, z]^T$ as a function of time t
θ, ϕ, ψ	Yaw, pitch and roll angles; positive in the counterclockwise direction
$\mathbf{a}(t)$	Robot's attitude $[\theta(t), \phi(t), \psi(t)]^T$ as a function of time t
$Y(t)$	Robot's pose $(\mathbf{x}(t), \mathbf{a}(t))$ as a function of time
Y_k	Robot's pose $(\mathbf{x}_k, \mathbf{a}_k)$ when the k-th batch of measurements is obtained
$\mathcal{V}_k$	Set $\{\vec{\mathbf{v}}_{k,i} \in \mathbb{R}^3 : i \in \mathbb{N},\ i \leq m_k\}$ in the k-th batch of measurements
M_k	k-th batch of measurements $(\mathbf{x}_k, \mathcal{V}_k)$
$\mathcal{M}_k$	Sequence $\mathcal{M}_k = \{M_i : i \in \mathbb{N},\ i \leq k\}$ up to the k-th batch
$\mathcal{M}_0$	Empty sequence of batches ($\mathcal{M}_0 = \emptyset$)
$\mathcal{Y}$	Volumetric grid comprised of a set of voxels
l	Index of a voxel belonging to the volumetric grid $\mathcal{Y}$
$\epsilon \in \mathbb{R}$	Voxel's edge
$v : \mathbb{R}^3 \rightarrow \mathcal{Y}$	Function that determines what grid's voxel a given point belongs to
$\mathbf{w} : \mathcal{Y} \rightarrow \mathbb{R}^3$	Function that computes the center coordinates of a given voxel
Y_i^s	Pose $(\mathbf{x}_i^s, \mathbf{a}_i^s)$ *selected* by robot $i \in \mathcal{F}$
$\vec{\mathbf{u}}(\mathbf{x}, l)$	Vector connecting the robot's position $\mathbf{x}$ to the center of voxel l
$\vec{\mathbf{u}}(Y_1, Y_2)$	Vector connecting the center of mass of a robot with pose Y_1 to the center of mass of another robot with pose Y_2
$t_{k_{max}}$	Mission execution time; time instant associated with the k_{max}-th batch
$t_{k_{max}}(n)$	Mission execution time with a fleet $\mathcal{F}$ of n robots
$\nu(n)$	Speedup measure for a fleet $\mathcal{F}$ with n robots
d_T	Traveled distance by the robot during a mapping mission

Symbol	**Description**
NOTATION RELATED WITH PROBABILISTIC MAPS	
$C_l \in [0,1]$	Continuous random variable representing the coverage of $l \in \mathcal{Y}$
c_l	Coverage estimate for a voxel $l \in \mathcal{Y}$: $c_l \in [0,1]$
$d_j \in \mathbb{R}_0$	Distance between the sensor and the detected obstacle
$d_j^l \in \mathbb{R}_0$	Distance between the sensor and the center of $l \in \mathcal{Y}$
D_j^l	Individual measurement (d_j, d_j^l) influencing the coverage of $l \in \mathcal{Y}$
$n_k(l)$	Number of measurements influencing the coverage of $l \in \mathcal{Y}$ after k batches
$\mathcal{D}_k^l$	Sequence of measurements $\mathcal{D}_k^l = \{D_j^l : j \in \mathbb{N},\ j \leq n_k(l)\}$ influencing the coverage of a voxel $l \in \mathcal{Y}$, after k batches of measurements
$\mathcal{D}_n^l$	Sequence of n measurements $\mathcal{D}_n^l = \{D_1^l, \ldots, D_n^l\}$ influencing the coverage of a voxel $l \in \mathcal{Y}$; $\mathcal{D}_{n_k(l)}^l$ and $\mathcal{D}_k^l$ are equivalent notations
$\mathcal{D}_0^l$	Empty sequence of influencing measurements ($\mathcal{D}_0^l = \emptyset$)
$p(c_l \mid \mathcal{M}_k)$	Coverage pdf of $l \in \mathcal{Y}$, after k batches; equal to $p(c_l \mid \mathcal{D}_k^l)$
$p(c_l^\triangle \mid \mathcal{M}_k)$	Histogram of the quantized version $C_l^\triangle$ of C_l
b	Number of bins for the quantized version of the voxel's coverage
$\mathcal{C}$	Volumetric map represented through the set $\{C_l : l \in \mathcal{Y}\}$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$	Set $\{p(c_l \mid \mathcal{M}_k) : l \in \mathcal{Y}\}$ of pdf describing statistically the map
$H(\mathcal{C})$	Map's entropy
$H(t_k)$	Map's entropy at $t = t_k$ (after k batches of measurements)
H_{th}	Map's entropy threshold used as a stopping criteria: $H(t_{k_{max}}) \leq H_{th}$
NOTATION RELATED WITH SHARING INFORMATION	
$I_{k,i}$	Information utility of the measurement $\overrightarrow{\mathbf{v}}_{k,i} \in \mathcal{V}_k$
I_k	Information utility of the k-th batch of measurements M_k
$\mathcal{U}_k$	Set $\{\overrightarrow{\mathbf{u}}_{k,1}, \ldots, \overrightarrow{\mathbf{u}}_{k,s_k}\} \subseteq \mathcal{V}_k$ of s_k useful measurements
$\mathcal{U}_k'$	Set of u_k useful measurements received from other robot at $t = t_k$
S_k	k-th batch $(\mathbf{x}_k,\mathcal{U}_k)$, $\mathcal{U}_k \subseteq \mathcal{V}_k$ sent to other robots at $t = t_k$
R_k	k-th batch $(\mathbf{x}_k',\mathcal{U}_k')$ received at $t = t_k$ from other robot located at $\mathbf{x}_k'$
m_T	Total number of measurements processed by the robot
u_T	Total number of useful measurements received from other robots
I_{min}	Minimum allowable inform. utility for a communicated measurement
$\max(s_k)$	Maximum number of allowable communicated measurements per batch

Symbol	**Description**
NOTATION RELATED WITH TRAVERSED, OCCUPIED AND VISIBLE VOXELS	
$\mathcal{Z}(\overrightarrow{\mathbf{u}},\mathbf{a}) \subset \mathcal{Y}$	Set of voxels traversed by $\overrightarrow{\mathbf{u}}$ when applied in point $\mathbf{a}$
$\mathcal{Z}_{k,i}$	Traversed (influenced) voxels by the measurement $\overrightarrow{\mathbf{v}}_{k,i} \in \mathcal{V}_k$
$\mathcal{O}^j(Y)$	Set of voxels occupied by a robot $j \in \mathcal{F}$ with pose $Y = (\mathbf{x}, \mathbf{a})$
$\mathcal{O}^{\mathcal{G}}$	Set of voxels currently occupied by a subset of robots $\mathcal{G} \subset \mathcal{F}$
$\mathcal{O}(Y_1, Y_2)$	Set of traversed voxels when moving between poses Y_1 and Y_2
r_i, α_i	Visibility parameters of robot $i \in \mathcal{F}$: range and aperture, respectively
$\mathbf{V}(\mathbf{x}_i, \mathbf{a}_i, r_i, \alpha_i)$	Robot's visibility region; a subset of $\mathbb{R}^3$
$\mathcal{Z}^i$	Set of visible voxels by robot $i \in \mathcal{F}$
$\mathcal{W}^i$	Set of visible voxels by other robots $\mathcal{F}\backslash i$ than robot i
$\mathcal{W}$	Set of visible voxels by the fleet $\mathcal{F}\backslash i$; equal to $\mathcal{Z}^i \cup \mathcal{W}^i$

Symbol	**Description**
NOTATION RELATED WITH EXPLORATION	
$\mathcal{N}(\mathbf{x}, \varepsilon)$	Set of voxels in a neighborhood of position $\mathbf{x}$ with a radius ε, 6 DOF
Γ	Robot's sensor motion plane for a 3 DOF robot
$\hat{\mathbf{p}},\ \hat{\mathbf{q}}$	Orthogonal axes defining the robot's motion plane Γ
$\underset{\Gamma}{\mathrm{proj}}\ \overrightarrow{\mathbf{u}}$	Projection of a vector $\overrightarrow{\mathbf{u}}$ in the robot's sensor motion plane
$\mathcal{N}_\Gamma(\mathbf{x}, \varepsilon)$	Set of voxels in a neighborhood of position $\mathbf{x}$ with a radius ε, 3 DOF
$H(l)$	Entropy of a voxel $l \in \mathcal{Y}$: $H(l) = H(C_l)$
$\overrightarrow{\nabla} H(l)$	Entropy gradient at a voxel $l \in \mathcal{Y}$
$\left\lVert \overrightarrow{\nabla} H(l) \right\rVert_N$	Magnitude of the entropy gradient
$\overrightarrow{\nabla} H_\Gamma(l)$	Entropy gradient at a voxel $l \in \mathcal{Y}$ projected on plane Γ
$\left\lVert \overrightarrow{\nabla} H_\Gamma(l) \right\rVert_N$	Normalized magnitude of vector $\overrightarrow{\nabla} H_\Gamma(l)$; a value $\in [0, 1]$
$\hat{\mathbf{p}}(l)$	Robot's gaze direction at a voxel $l \in \mathcal{Y}$
$\mathcal{C}^i$	Visible map of a robot $i \in \mathcal{F}$
$H(\mathcal{C}^i)$	Visible map's entropy of a robot $i \in \mathcal{F}$; uncertainty covered by it
$\mathcal{T}^i$	Visible map of other robots $\mathcal{F}\backslash i$ than robot i
$H(\mathcal{T}^i)$	Entropy of the visible map of other robots $\mathcal{F}\backslash i$ than robot i
$\mathcal{T}^i$	Visible map by the fleet $\mathcal{F}$
$H(\mathcal{T}^i)$	Entropy of the visible map of the fleet $\mathcal{F}$
$I(\mathcal{C}^i; \mathcal{T}^i)$	Mutual information between robot i and the other robots $\mathcal{F}\backslash i$
$\lambda(l)$	Non-redundancy coefficient at a candidate voxel $l \in \mathcal{Y}$
Y^l	Robot's pose associated with a candidate voxel $l \in \mathcal{Y}$
$\rho(\mathbf{x}, l)$	Reachability coefficient when moving between $\mathbf{x}$ and a voxel $l \in \mathcal{Y}$
$\hat{\mathbf{p}}(Y)$	Robot's gaze direction associated with a pose Y
$\eta(l)$	Non-interference coefficient for a candidate voxel $l \in \mathcal{Y}$
$\vartheta(\mathbf{x}, l)$	Cost coefficient when moving between $\mathbf{x}$ and a voxel $l \in \mathcal{Y}$
l^s	Selected voxel to be the next robot's sensor exploration viewpoint

Chapter 1

Context, Questions and Method

This thesis addresses the question of how to share efficiently information within a robotic system comprised of several mobile robots, towards the stimulation of a cooperative behavior of the multi-robot system. The aim of this introductory chapter is to give the reader an overview of the document, by answering three fundamental questions about the research herein reported: In what context does it appear? What research question does it intend to answer? What method was used to pursue the research?

This chapter starts by putting the research reported in the later chapters of the thesis in the context of *robotics* and, more specifically, in the context of robotic systems comprised of more than one robot, *i.e. multi-robot systems*. Given the context, the main research question and the associated subsidiary research questions are clearly stated, including a preliminary discussion of the associated issues. Then, the method that was used to answer those questions is also briefly presented. At the end of the chapter, the contributions and novelties of the thesis are clearly enumerated and an outline of the document's chapters is given.

1.1 The context: Robotics and Multi-Robot Systems

Robotics emerged as a research field a few decades ago and has known important developments since then. Its fascination lies in the attempt to create machines — *robots* — that may imitate some of the complex behaviors found in biological systems, especially in humans. Most of the people, especially those that are not experts in robotics, remember popular fictional movies, such as Star Wars and Star Trek, and have built the idea that a robot is a complex machine, having some of the human skills and looking like people, which are able to navigate and localize autonomously, use legs to walk and run, learn,

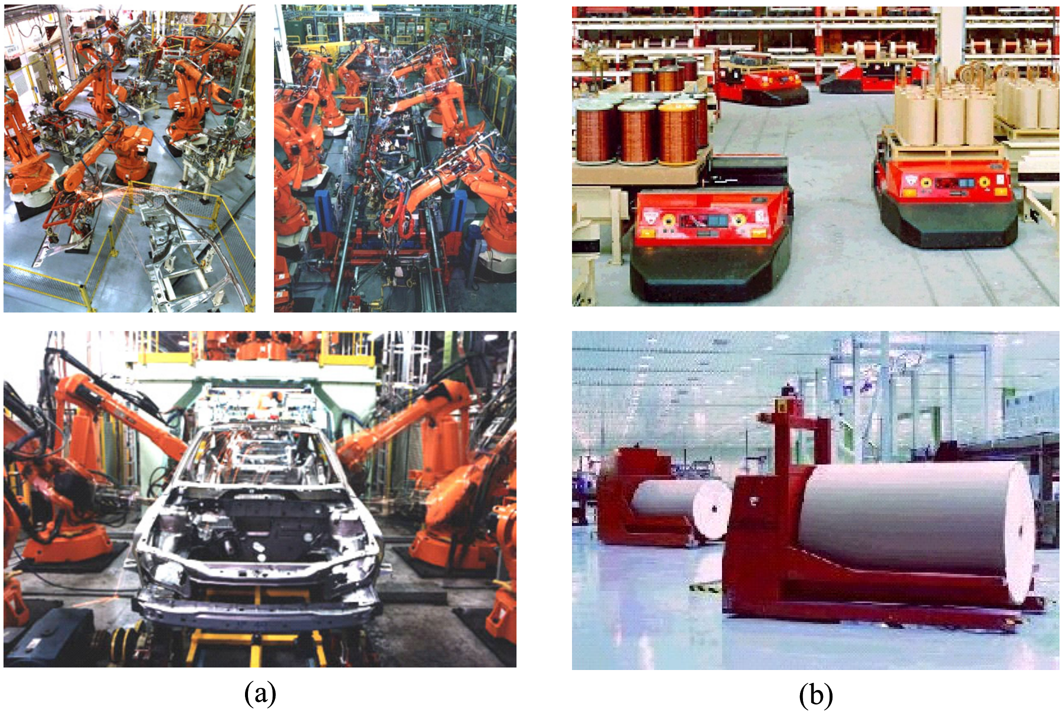


Figure 1.1: Examples of industrial robots: (a) robotic manipulators in automotive industry [WCfMJT05]; (b) automatic guided vehicles in automatic material handling systems [Roc01].

talk, reason and interact in a friendly way with humans. The expectation created by those futuristic movies was however easily disappointed in any visit to a research lab, in the beginning of robotics, because the *real* mobile robots were quite far from those skills and had limited capabilities.

Robots have been developed essentially to help or substitute humans in tasks which are either repetitive or dangerous. The first industrial robots appeared in the sixties and were used in the automotive industry to execute assembly tasks, such as soldering or painting the chassis (see Fig. 1.1-a). They were robotic manipulators, similar to a human's arm, which executed repetitively a sequence of operations at high speed and with good precision.

Since then, other types of robots than manipulator arms have been also developed for different applications, wherein many of them are not fixed platforms and have the ability to move within an environment: these are *mobile robots*. One of the first application domains for these mobile robots was to automate the materials transportation in manufacturing systems through automated guided vehicles (see 1.1-b).

Besides locomotion, other important resources have been integrated in mobile robots, namely more computational power, wireless communication, actuators (other than loco-

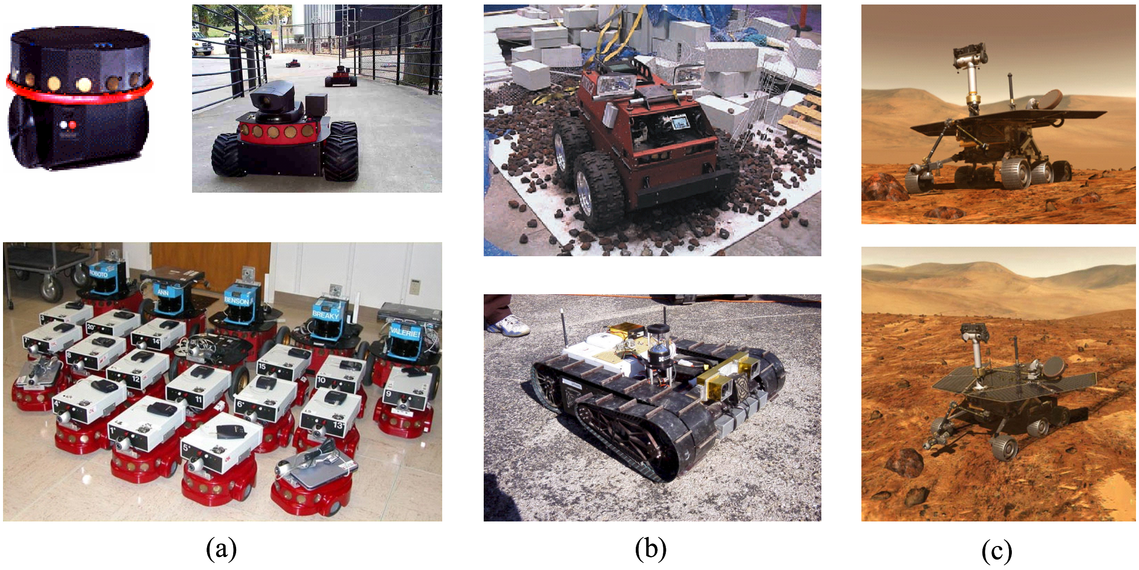


Figure 1.2: Examples of ground mobile robots: (a) some examples of research mobile robots, commonly used by the robotics research community in either indoor experiments (top-left and bottom) [Nom99, KFO+04] or less structured outdoor environments (top-right) [UA04]; (b) mobile robots for non-structured outdoor environments, using either wheels (top) [KT01] or tracker wheels (bottom) [NAS05]; (c) Spirit rover from NASA, [NAS05], which landed in Mars at Jan. 2004 for exploration missions on the planet's surface.

motion) and sensors of many types. Generally, the mobile robot uses its sensors to gather data from the surrounding environment, process sensory data to perceive the environment, reasons based on the models created by perception, uses its locomotion capabilities to move within the environment and uses its actuators to act on the environment (*e.g.* gripper, robotic hand, manipulator arm, *etc.*).

For terrestrial applications, mobile robots use either wheels to move within structured environments, essentially comprised of flat, even surfaces (*e.g.* rooms, buildings with elevators), or other more complex locomotion devices, such as articulated wheels, tracker wheels or legs, to move within less structured environments, especially outdoors, having steps or pronounced slopes (see Fig. 1.2). These mobile robots have plenty of application domains, besides being used in industry within manufacturing systems. Examples are indoor and urban surveillance, nuclear power plant maintenance and decommissioning, cleanup of toxic waste, planetary exploration, search and rescue, mine clearing, military applications, *etc.*

For applications wherein mobile robots are not always in contact with the ground, submarine, boat-like and aerial robots have also been developed (see Fig. 1.3). These robots can perform important missions such as research and exploration of oceans, fire

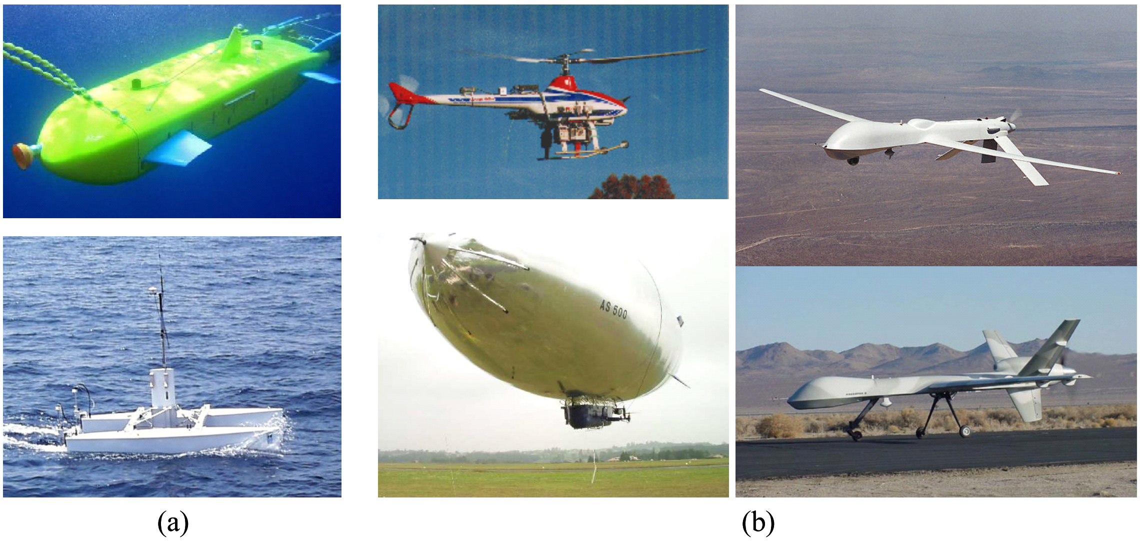


Figure 1.3: Examples of marine and aerial mobile robots: (a) an autonomous underwater vehicle (AUV) and an autonomous surface craft, both developed at ISR Lisbon, Portugal [DSO05], to be used in oceanography; (b) an autonomous helicopter for building aerial maps (top-left) [AHP05], an autonomous airship for terrain mapping at low altitude (bottom-left) [HJSL04], and the Predator unmanned aircraft system (right), which has been used in military reconnaissance and surveillance missions [Pre05].

fighting, terrain mapping, military reconnaissance and surveillance, *etc.*

Sensors based on different technologies have been developed, which allow a robot to measure many different physical variables.

For localization of a ground mobile robot, it is usual to couple digital encoders to its motors' axes, for measuring the robot's instantaneous velocity ([1]) or the robot's displacement.

To analyze the environment, it is also usual to use digital cameras that convert images into arrays of numbers — pixels — representing the color of a set of sampled points in the image. Stereoscopic vision systems, comprised of two or more digital cameras, allow to associate a distance to each pixel of the image of one of those cameras, *i.e.* to obtain a depth map.

Laser scanners, integrating a laser with a sweeping mechanism, measure distances

[1] The robot's velocity may be estimated through the integration of the motors' velocity, a method known as odometry. Nevertheless, the robot's localization using odometry accumulates error along the time, due to non-linear phenomena such as wheels slippage. There are other more robust localization methods, such as: detecting and measuring the robots' position relative to distinguishable landmarks whose localization is known, a method which is known as trilateration; or GPS — Global Positioning System — that uses the same principle with a constellation of satellites.

using the time-of-flight principle, whereby the light propagation velocity is related with the time needed by a ray of structured light to return to the laser emitter after being partially reflected by an obstacle, in order to compute the distance to the obstacle. Using the same principle, ultrasonic sensors (sonars) and infra-red sensors measure distance by computing the propagation delay in the medium of, respectively, ultra-sounds and infra-red light.

Inertial sensors, such as magnetometers, accelerometers and gyroscopes, may support the robot's navigation with information about its accelerations. Other examples of sensors are pressure sensors (*e.g.* used, for example, for manipulating safely fragile objects), microphones, tactile sensors, infra-red cameras to detect warm bodies in search and rescue missions, *etc.* Different types of chemical sensors can also be used to study the composition of rocks, ground, dust and gases in space exploration missions (*e.g.* spectometers).

Those sensors can be used by robots to perceive the environment in an intelligent and powerful way. Robots use their computational power to extract and interpret sensory data, in order to perform complex reasoning and useful actions. For example, a range sensor providing distance information can be used by a moving robot to perceive how far away it is from obstacles and thus implement a collision avoidance behavior. Using color images provided by a digital camera, a mobile robot performing a surveillance mission inside a building may use an internal data base containing the faces of known people to decide whether a found person is authorized to be therein.

Nowadays, the laymen's idea about robotics that was referred at the beginning of this chapter, which is often inspired on popular fictional movies, is not hopefully so futuristic, because robots are becoming *indeed* complex machines, having important capabilities, such as control autonomy (operation without any human intervention), intelligent perception based on powerful sensors and high on-board computation power, the ability of adapting to non-structured and unknown environments and learn, intelligent planning and reasoning, biped locomotion, friendly interaction with humans, *etc.*

Although these robots' human-like skills are still of limited complexity and ability, being based on emerging and non-mature technologies, important and promising achievements have been reported for the past few years. Some of the cutting-edge commercial robots, as the anthropomorphic mobile robots shown in Fig. 1.4, look very like humans, being comprised of body, head, legs and arms. These robots are usually denoted as humanoids or androids and have important capabilities related with biped locomotion, such as walking, climbing and jumping. Moreover, substantial research is being pursued to develop human-robot interfaces, so as to get those robots in close interaction with humans

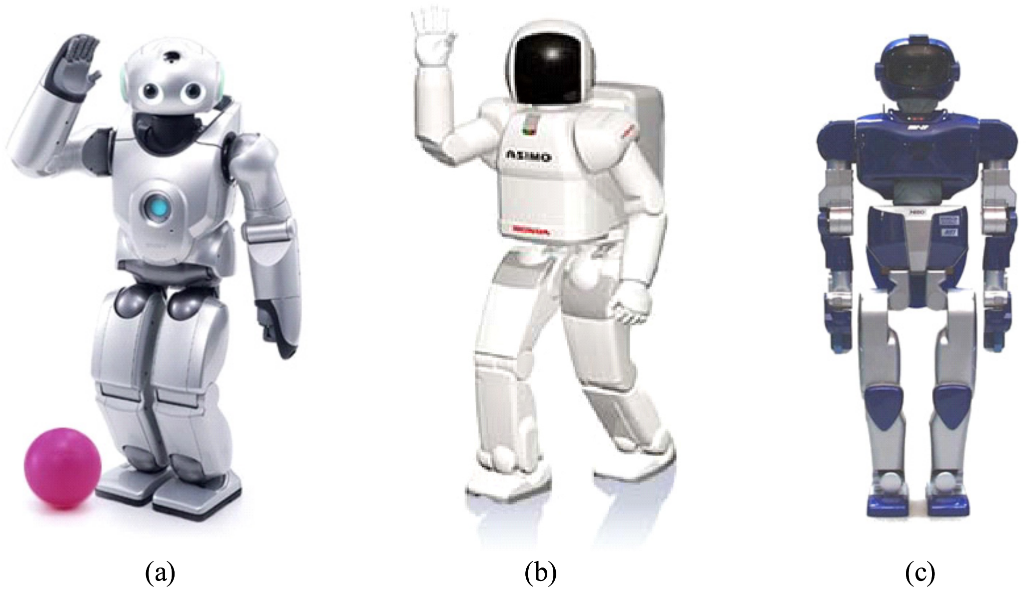


Figure 1.4: Some of the commercially available humanoid robots: (a) QRIO from Sony [Son05]; (b) ASIMO from Honda [Hon05]; (c) H2-RP from Kawada Industries [H2R05].

in many daily tasks.

The current most essential robotics' goal is to conquer the people's trust on robots, so as to generalize the use of robots in close interaction with humans, at home, at the work-place, for education, as well as in other emerging areas, such as medical robots, social robots, *etc.* This challenging goal is already being pursued by making more reliable, friendly and useful robots, which may assist humans in their daily life. This is usually denoted as *service robotics*. Its goal is to build a bridge from the industrial robots stage to the personal robot stage, *i.e.* robots that can be important assistants to our lives in the future, whose objective will be to provide humans and organizations with services that relieve the human being from some unpleasant tasks, such as cleaning the house, cooking, serving meals, looking for and picking up things, *etc.*

Service robots are intended to be used in many domains, such as household, care and rehabilitation, cleaning, agriculture, construction, demining, entertainment, medical, mining, monitoring, office, refueling, *etc.*

In housekeeping, they may automate unpleasant tasks related with house keeping at home or gardening (*e.g.* robotic vacuum-cleaners, lawn-mowers, *etc.*). In care, they may act as social robots and provide children and elderly people with friendly interaction and company (*e.g.* robotic pets). In rehabilitation, they may be used to support elderly and handicapped people, so as to improve their mobility, autonomy and quality of life.

They may be used to clean stations, office buildings, planes, boats, *etc.* In agriculture, they may automate operations such as collecting fruits or sowing seeds, pesticides or fertilizers. In entertainment, they may be used to serve as guides in museums, exhibitions and amusement parks, or to serve as entertainment and educational toys for children. In medical applications, they may enable minimal invasive and remote or assisted surgery. In surveillance and inspection missions, they may be used to guard buildings or inspect sewer systems. They may be used for refueling in automatic gas, with important benefits related with saving time, reducing risks to health and correct selection of fuel. And many other examples related with the life of either individual persons or organizations.

1.1.1 Why using Multi-Robot Systems?

For some robotic tasks, especially those that are intrinsically distributed and complex (*e.g.* covering a wide area in a surveillance mission, transporting large objects by using more than one robot, finding victims in the debris of a catastrophe in less than a given time, *etc.*), a team of several cooperative mobile robots — a cooperative multi-robot system — is required to either make viable the mission accomplishment or, at least, accomplish the mission with better performance than a single robot.

Cooperative multi-robot systems (MRS) have received significant attention by the robotics community for the past two decades, because their successful deployment have unquestionable social and economical relevance in many application domains.

Due to the expendability of individual robots, MRS may substitute people in risky scenarios, which are comprised of tasks that are typified by the high potential for damage to individual collective elements, such as cleanup of toxic waste, nuclear plant maintenance and decommissioning, planetary exploration, fire fighting, search and rescue, *etc.* While loosing one or more robots in those risky scenarios may be tolerable, human casualties are certainly not.

Furthermore, in less risky scenarios, MRS may still relieve people from collective tasks that are intrinsically monotonous and repetitive (*e.g.* surveillance, forest patrolling and fire detection, handling materials in a a manufacturing system, *etc.*) and allow them to be occupied by more nobler tasks (*e.g.* coordinating several teams of robots, development and innovation tasks, *etc.*).

When compared with single robot-based solutions, MRS provide important advantages related with space and time distribution, complex problems decomposition, reliability, robustness and cost.

Space distribution has the important advantage of many robots being in many places at the same time. Since each robot comprises individually a set of useful resources — sensors, perception capabilities, computational power, communication, locomotion, and actuators —, it is sufficiently autonomous to perform valuable subtasks. The simultaneous operation of many of these robots allow to perform many, perhaps different and complementary, tasks at the same time.

By taking advantage of the computational power distribution yielded by a multi-robot system, certain complex problems (*e.g.* exploring an unknown environment, finding the best route to a target position, pursuing and capturing a set of intruders, *etc.*) may be decomposed in simpler sub-problems, which are then allocated to the individuals robots. Therefore, a complex mission may be assigned to a team of intelligent and autonomous robots, which decompose the mission in simpler tasks and cooperate in order to successfully accomplish the mission.

In complex missions, requiring a considerable amount of different resources (*e.g.* different sensory capabilities, both ground and aerial mobility, *etc.*), MRS enable high flexibility by distribution of the risk. If the resources required by a complex mission are distributed along different robots, which have heterogeneous capabilities and present some mutual overlapping or redundancy, the robotic team becomes more robust and reliable, because it remains usable if some of the robots fail or become damaged, though it suffers a graceful degradation of its performance.

Because each individual robot of the team tends to be simpler and, thus, more reliable, than a monolithic single robot having all the capabilities required by the mission, a multi-robot system may present a more reduced cost.

The acceptance of any robotic system is highly dependent on its reliability, especially in critical application domains (*e.g.* security, military missions, rescue, planetary exploration, *etc.*). MRS allow to judiciously integrate some redundancy level, which in turn improves the robotic system's overall reliability and fault tolerance. In this way, even if some individual robot fails, the robotic team may be still capable of accomplishing the mission assigned to it, which is a very appealing characteristic.

1.2 The questions: What is this thesis about?

Although cooperative multi-robot systems present the aforementioned important advantages over single robot-based solutions, they are quite challenging because, besides inheriting all the problems associated with developing a single robot, they raise new research problems.

A not exhaustive list of these problems includes: cooperative perception by fusing noisy observations from different robots; the team's architecture (*e.g.* centralized vs. distributed control) and how to achieve coherence and performance with distributed control and coordination, given that information is usually distributed and each robot has only a partial view of the world; cooperative planning by decomposing complex tasks and assigning subtasks to the individual robots; ensuring a coherent team behavior when unexpected events occur (*e.g.* robots failures, environmental disturbances, *etc.*).

This thesis mainly addresses the multi-robot systems' problem of coping with the information distribution along different robots, which is indeed a transverse issue. And this is because of several reasons.

Firstly, since information is usually distributed, building a consistent model of the robotic team's environment depends on fusing sensory data from different robots, which is noisy and sometimes inconsistent with the current model. Secondly, the way the distributed information is managed within the team is interconnected with whether the team is centrally controlled or has distributed control. Moreover, cooperative planning depends on a sufficient level of awareness, about the robots' state and about the tasks required by the mission assigned to the team, which, in turn, is related with the ability of getting access to the distributed information in a purposive, efficient and consistent way.

Research Question: How to foster cooperation among intelligent robots, based on sharing useful information and proper coordination?

The question is sufficiently general to be studied within any application domain of multi-robot systems. An alternative would be to maintain this level of abstraction and to study it mathematically and try to validate it afterwards in some specific application domain. Nevertheless, at the start of the research work, it seemed not to be viable to pursue the research in that way, without loosing the practical sense.

As it will be made clearer later on in this document, it would not be viable to address the question conveniently without taking some target application, because answering it certainly requires to be able to quantify and measure the performance of the team of robots in order to evaluate the proposed approaches. Measuring performance, in turn, is a concept that is tailored to the target application. Furthermore, robotics is essentially the integration of several knowledge areas and the associated technologies (*e.g.* mechanics, electronics, computer sciences, artificial intelligence, *etc.*) with the aim of solving *practical* problems wherein the automation provided by robots is required to substitute or assist human beings.

Therefore, the alternative of configuring the research in the scope of *applied robotics* seemed to be more reasonable and, thus, in order to reduce the problem's abstraction to a level closer to reality, a specific application domain was chosen at the beginning of the research.

Cooperative multi-robot systems have received significant attention by the robotics community for the past two decades. However, most of the research body has been quite informal and there is still a big lack of formalisms to answer most of the aforementioned important design questions.

The main goal of the thesis is to give a contribution to formally address the aforementioned research question in a specific application domain of a multi-robot system: *building volumetric maps*. This is a loosely-coupled task, in the sense that it might be performed with a single robot, *i.e.* it does not require tight cooperation when executed by a team of robots, but performance can be significantly improved through cooperation (*e.g.* attaining a map in less time). There are several reasons why the robotic mapping task was chosen to pursue the research.

Firstly, because it is an important robotics' application domain. It is a valuable task for supporting robots' navigation based on a world model and, thus, it must be present in many useful robotic applications. Moreover, if the goal is the map *itself*, there are still relevant target applications, such as: building fastidious maps of indoor environments (*e.g.* buildings); building maps of buried utilities (*e.g.* gas pipes, power or communication lines, *etc.*), in order to avoid getting too close to them in construction activities; substituting humans on building detailed maps of hazardous environments (*e.g.* abandoned underground mines or nuclear facilities), in order to support monitoring and maintenance procedures.

Secondly, because it is easy to build a realistic robotic mapping testbed inside a research laboratory, since it just requires some workspace and robots having at least one range sensor and wireless communication. Moreover, the mission's result yielded by the testbed is tangible and it is the *same* as in a real application: the map.

Thirdly, because the research group wherein the work was conducted has an important know-how about computer vision and stereo-vision, which can be readily used in robotic mapping to measure distances and build detailed 3-D maps.

Answering the aforementioned research question within the domain of building volumetric maps raises a number of other subsidiary questions. This thesis focus on five of these questions.

Subsidiary question 1: How to represent a volumetric map and its uncertainty?

An obvious requirement of building a map is obviously a model for the maps, *i.e.* a mathematical method to represent and update the robots' knowledge about the environment, based on the data provided by range sensors. This issue is thoroughly addressed in chapter 4.

Subsidiary question 2: How to use the uncertain and partial information contained in the map to completely explore the environment and minimize the time required to do this?

As range sensors have limited range, are subject to occlusions and yield noisy measurements, robots have to navigate within the environment in order to survey it sufficiently and reduce the map's uncertainty. Therefore, a principled exploration method is needed to select the robot's next exploration viewpoint, based on its current map. The goal is to maximize the information gain associated with the new exploration viewpoint, so as to reduce the map's uncertainty as fast as possible. This issue is addressed in chapter 4.

Subsidiary question 3: How to control a cooperative multi-robot system without any centralized control?

This thesis is based on the assumption that distributed control should be the preferred control paradigm on controlling a multi-robot system. This design option is justified by the pitfalls of centralized controllers.

Centralized controllers neither cannot scale up to teams of many robots nor deploy reliable robotic systems, due to the concentration of the decision power on a single machine. Moreover, they require extensive communication in order to concentrate sufficient information on a single machine and, thus, take principled decisions. And, most of all, they cannot cope with complex tasks due to the exponential increase of the associated decision methods' search space. This is thoroughly discussed in chapters 2 and 5.

Distributed control raises however the problem of attaining a global coherent behavior of a team, wherein each robot intervenes in the decision process, though its sensors give it only a partial view of the world. Therefore, a method to share efficiently information among robots is needed to ensure the viability and success of the distributed approach.

Subsidiary question 4: How to assess information utility so as to communicate useful data among robots?

This thesis proposes that robots should share efficiently information based on a criteria of information utility, so as to avoid communicating redundant or unnecessary information. Therefore, any architecture model based on this criteria requires a formal method of assessing information utility. This issue is addressed in chapter 4 and is used in chapter 5 to devise a distributed architecture model for robotic mapping, whereby robots share altruistically useful sensory data.

Subsidiary question 5: How to coordinate the robots' actions so as to achieve more effective cooperation?

Besides ensuring that each robot has a consistent global view about the mission and the team's state, it is also needed to endow the team with a suitable coordination mechanism, aiming at synchronizing the robots' individual actions and attaining a globally consistent and cooperative behavior. This issue is addressed in chapter 6 in the context of robotic mapping, with the aim of coordinating the exploration actions of individual robots.

1.3 The method: How are the questions addressed?

The research questions stated in the previous sections are incrementally addressed along this thesis. The document's structure reflect indeed this progressive process. Besides this introductory chapter, there are two chapters essentially concerned with research background and methodology, three chapters presenting contributions, aiming at answering the research questions, and a final chapter with conclusions.

The first phase of the research included a thorough review of the state of the art about multi-robot systems, with a special emphasis in the cooperation issue, and about robotic mapping, the chosen application domain to be the testbed. Although this effort towards a solid background presented a peak in the beginning of the research, it continued during all the research process until writing this document, so as to be aware of the contributions in the aforementioned research areas, which have known rapid developments for the past few years.

Special attention was also given to the main mathematical tool used herein: entropy and its associated definitions. The contributions of this thesis depend to a great extent on the usefulness of that mathematical entropy on formalizing relevant concepts, such as the uncertainty of a belief or the mutual information between two random variables. These efforts resulted in the chapters 2 and 3.

The following phases of the research, which are reported in chapters 4, 5 and 6, are indeed tightly interconnected and are a tangible manifestation of the aforementioned incremental research process.

Given the research questions and the volumetric mapping application domain, chapter 4 reports the first research step, which can be informally summarized as follows: endow an autonomous mobile robot, equipped with a range sensor, whereby it is able to represent a map, update iteratively the map upon sensory data and survey efficiently the environment, so as to build a map of it as fast as possible.

Taking the volumetric mapping framework as a basis, chapter 5 reports the second research step, which can be informally summarized as follows: give a distributed control architecture to a set of autonomous mobile robots, which are individually able to build a volumetric map of the environment, whereby the team exhibits a global cooperative behavior, being able to attain a map in less time than a single robot, by taking advantage of the robots' space distribution and cooperation through sharing useful sensory data via communication.

Given the cooperative distributed model, chapter 6 reports the third research step, which can be informally summarized as follows: although the team of cooperative mobile robots is able to attain a map of the environment in less time than a single robot, by sharing useful sensory data, give the robots a proper exploration coordination mechanism to achieve more effective cooperation, by taking maximum advantage from their spatial distribution; this can be done by avoiding the situations wherein either more than one robot senses the same volume or robots interfere each other.

In the course of this research process, special attention was given to the implementation and validation of the proposed approaches in mobile robots. Computer simulations were also used in chapter 6 to carry out extensive experiments with teams of robots of different sizes. Although both empiric methods have the same purpose — to validate a theory through experiments — they represent very different approaches.

Computer simulations have the obvious advantage of allowing to obtain experimental results much more quickly than with mobile robots, because the researcher is not restricted by the time and expense associated with using mobile robots. Anyone who has already worked with *real* robots to test some research theory, knows how time-consuming is to debug a robot's program or solve any problem in the hardware. Moreover, even in the absence of these implementation problems, an experiment may be simulated in a computer in much less time than the time required to carry out a similar experiment with mobile robots. Since the researcher is able to abstract the implementation problem, he or she can focus more tightly on the precise aspect of the problem being studied.

Nevertheless, the advantages of implementing the ideas in mobile robots may outweigh largely the aforementioned disadvantages. Firstly, the researcher has to abstract some features of the system in order to build a computer model of it, which necessarily involves some degree of simplification. This may lead to a significant distortion of the real phenomena and the results obtained through simulations may become invalid. Secondly, sooner or later, the developed theories must be validated within the *real* world, with mobile robots, otherwise their validation is likely to be questionable.

In this thesis, the approaches proposed in theory were always validated through their implementation in mobile robots, and by carrying out experiments in a physical environment, which contained the most essential characteristics of a real scenario, though some implementation aspects were simplified, especially those aspects that were not relevant to answer the research questions.

Nevertheless, in chapter 6, a hybrid empiric approach is followed: firstly, results obtained in experiments with mobile robots are used to tune a simulation model, in order to ensure that results obtained through computer simulations are indeed valid; afterwards, in order to reduce the time needed to obtain extensive experimental results, computer simulations are used to preview the performance of teams of robots with varying team sizes.

1.4 Preview of novelties and contributions

Given the context related with building volumetric maps with teams of cooperative mobile robots, the contributions of the thesis include:

- *Probabilistic maps* – A grid-based probabilistic model of a volumetric map is proposed, which enables to model explicitly uncertainty. The main novelty concerning grid-based maps are a more compact representation than histograms [SB03] for the belief about the state of each cell and an efficient Bayes filter to update the map upon measurements taken at different instant times and from different locations. A probabilistic model of a range sensor is also presented, which enables to convert range measurements into coverage estimates. This probabilistic framework was implemented in mobile robots and resulted in the contributions reported in [RDC05d, RDC05a].

- *Entropy gradient-based exploration strategy* – The frontier-based exploration concept [Yam98] is reformulated using entropy, by proposing an entropy gradient-based exploration method, whereby the robot's sensor is directed to frontier voxels between more explored and less explored regions. The method can be used by a robot to survey the environment and build the map iteratively. It was implemented and successfully validated in mobile robots. This contribution is reported in [RDC05d, RDC05a].

- *Distributed architecture model* – It is devised a distributed architecture model for building volumetric maps with a team of cooperative robots, whereby each robot is altruistically committed to share useful measurements with its teammates. The architecture was successfully implemented and validated in mobile robots and resulted in the contributions reported in [RDC05b, RDC05a].

- *Entropy-based measure of information utility* – In order to make viable the robots' cooperation through sharing useful information, an entropy-based measure of information utility is formulated, whereby sensory data is as useful as it contributes to improve the robot's map. The successful use of this measure in cooperative mobile robots is reported in [RDC05b, RDC05a].

- *Coordination mechanism based on the minimization of mutual information* – An exploration coordination mechanism is proposed to overcome the typical problems that arise in a multi-robot mapping mission due to lack of coordination: robots sensing regions that overlap each other; selecting not reachable exploration viewpoints; or not avoiding exploration points characterized by partial occlusions, due to the presence of other robots in front of the robot's sensor. It is demonstrated that overcoming these problems require the robot to be sufficiently aware about the other robot's state. The most important feature of the coordinated exploration method is to avoid sensing overlapping regions with different robots. This problem is formulated as a mutual information minimization problem, after deriving mathematical expressions for computing the mutual information between sets of random variables. The validity of the aforementioned coordinated exploration method is demonstrated in [RDC05c].

The use of the mathematical concept of entropy to formally represent several important concepts is transverse to every contributions of the thesis. Entropy, either in its simple form or as mutual information ([2]), is used in the thesis with different purposes: to measure the amount of uncertainty associated with a volumetric map [RDC05d, RDC05a]; to devise a straightforward exploration method [RDC05d, RDC05a]; to propose a formal measure of information utility for mapping missions [RDC05b, RDC05a]; and to measure the overlapping between the sensed regions by different robots [RDC05c].

1.5 Outline of the thesis

The thesis is organized in seven chapters. After this introductory chapter, which presents the context, research questions and methodology of the research herein reported, chapter 2 reviews thoroughly the most relevant state of the art related with cooperative multi-robot systems and robotic mapping. The purpose is to go deeply into the motivation, taxonomies and main issues related with cooperative robotics and robotic mapping, so as to put properly in context the contributions of the thesis, which are presented in the following chapters.

Chapter 3 provides the reader with the basics of information theory, so as to contribute to the readability of the following chapters. The basic definitions of entropy, joint entropy, conditional entropy and mutual information for discrete random variables are presented. It is also presented the entropy definition for continuous random variables: differential entropy.

Chapter 4 proposes a grid-based probabilistic model for a volumetric map based on entropy. In order to survey the environment and build the map iteratively, an entropy gradient-based exploration method is also proposed. The framework is illustrated and validated through the presentation of results obtained within experiments using a mobile robot equipped with stereo-vision, wherein volumetric maps of a real indoor environment were built.

Chapter 5 proposes a distributed architecture model for building volumetric maps with teams of cooperative robots. The information utility concept is defined as a balance between performance improvement and cost increasing; the information is as useful as the ratio of these two variables takes high values. Using this definition, an entropy-based measure of information utility is formulated for robotic mapping missions, in order to propose a cooperation scheme based on sharing efficiently sensory data via communi-

[2] Mutual information is an entropy-based measure of the amount of information that a random variable has about another. This and other information theoretical concepts are thoroughly presented in chapter 3.

cation. The cooperative distributed model is validated through cooperative volumetric mapping experiments, using mobile robots equipped with stereo-vision, in a real indoor environment.

Chapter 6 extends the cooperative distributed model proposed in chapter 5, so as to overcome the problems that arise due to the lack of coordination of the exploration actions taken by different robots of a team. With this purpose, the entropy gradient-based exploration method proposed in chapter 4 is extended with a coordination mechanism, whose main novelty is to formalize the sensing overlapping problem as a mutual information minimization problem. The coordinated exploration method is validated and compared with its uncoordinated counterpart, through volumetric mapping experiments with mobile robots and computer simulations.

Chapter 7 concludes the thesis. It makes a summary of the document, underlines the main conclusions obtained in the course of the research herein reported, and discusses the advantages and limitations of the presented contributions. The chapter ends by pointing out perspectives on future research.

Chapter 2

Background and related work

This chapter covers a significant part of the state of the art related with cooperative multi-robot systems and robotic mapping. The purpose of the chapter is to give the reader some background and motivation about cooperative robotics and robotic mapping, and put properly in context the research reported in the following chapters.

After focusing on the cooperation concept itself, the advantages and challenges posed by cooperative robotic systems are addressed. Some relevant taxonomies, the main issues of multi-robot systems and some cooperative robotic architectures often referred in the literature are also briefly presented and discussed. A special emphasis is given to the communication structure used in the multi-robot system, since the main question addressed in this thesis is how to share efficiently information via explicit communication. Then, the robotic mapping research topic is introduced along its main issues and known solutions in the literature. At the end of the chapter, the assumptions and contributions of the research reported herein are made clear.

2.1 The essence of cooperation

This section is focused on the *cooperation* concept itself. The aim is to understand more clearly what means being cooperative, what cooperative behavior does imply, and how can we take inspiration on cooperation examples observed in nature and human societies, to successfully develop human-made intelligent cooperative systems and, particularly, cooperative multi-robot systems.

One of the knowledge areas that has paid special attention to *cooperation* is Distributed Artificial Intelligence, a sub-field of Artificial Intelligence (AI), since it addresses problems related with constructing large, complex and knowledge-rich systems. It advocates that such systems should be decomposed into a number of autonomous *agents* that communicate and cooperate with one another within a decentralized control regime. These systems are denoted as *multi-agent systems* (MAS) [SV00].

The concept of *agent* plays a central role on these systems. Although there is no generally accepted definition of agent in AI, it may be defined as an autonomous and intelligent entity (*e.g.* a robot) with goals, actions and domain knowledge, situated in an environment. The way it acts is often called its *behavior*.

For the last two decades, MAS scientists have developed extensive work, providing both principles for constructing such complex systems, involving multiple agents, and mechanisms of coordination of independent agents' behaviors. Much of the research on non-robotic MAS is relevant to robotic MAS, which are usually denoted as *multi-robot systems*.

For some problems, especially those that are intrinsically distributed and complex, fostering cooperation among intelligent machines is driven by the assumption that multi-agent solutions have advantages over single agent solutions ([1]).

Research on this issue has been mainly conducted by roboticists in the context of multi-robot systems. However, besides knowledge about building single robot systems, this research area is multi-disciplinary and integrates a huge number of distinct fields, outside the Engineering Sciences, where it bears inspiration to obtain a cooperative collective behavior upon engineering the behavior of individuals. Thus, it integrates Engineering Sciences, Artificial Intelligence, Social Sciences (Organization Theory, Economics, Cognitive Psychology) and Life Sciences (Theoretical Biology, Animal Ethology). On the other hand, successful cooperative methods developed within robotics might be generalized to other knowledge areas.

2.1.1 Definitions of cooperation

Before the word *cooperation* can be applied to rule the behavior between intelligent agents and between them and humans, some thought about its meaning must be given. Although its literal reading — simultaneous operation — is quite general, the word has historically been used primarily to refer to the joint behavior of humans, and sometimes animals. The specific mechanisms of cooperation that we can find in the animal and human sphere depend on behavioral tendencies that effect the willingness to cooperate [Jun98].

[1] The word *agent* will sometimes substitute the word *robot* throughout the text, in order to keep the discussion as much general as possible.

Robotics researchers often distinguish between two types of cooperation: *collective robotics* and *cooperative robotics* [CFK97]. The former is often denoted as *swarm cooperation* and the latter as *explicit cooperation*. They are two different approaches to the same problem: how to obtain a desired *collective behavior* upon engineering the behavior of individuals. The term *collective behavior* denotes any behavior in a system having more than one agent (a multi-agent system). *Cooperative behavior* is a subclass of collective behavior that is characterized by cooperation.

Explicit definitions of cooperation in the robotics literature include [CFK97]:

- Joint collaborative behavior that is directed toward some goal in which there is a common interest or reward;
- A form of interaction usually based on communication;
- Joining together for doing something that creates a beneficial result, such as increasing the overall system performance.

These definitions emphasize three important dimensions of cooperative behavior, namely *task*, *mechanism* and *performance*.

The *task* is directed toward a goal shared by all the agents of the community, in which there is a common reward or interest beneficial to all agents.

The *mechanism* of cooperation, perhaps supported on some distributed control architecture and some explicit communication, rules the interactions among agents, so that the actions of individual agents are coherent with the system goal and beneficial to the system as a whole ([2]).

The *performance* of the system, as a whole, is enhanced through the existence of cooperation, creating a beneficial result that is a reward for all agents (*e.g.* reducing time to complete a task, increasing resources utilization, reducing energy waste, *etc.*). This means that cooperation yields a globally rewarding utility which is greater than the sum of the individual utilities.

[2] This statement has an implicit distinction between the goals of an individual agent and the system goal. Each agent may have its local goals and a system goal common to all agents in the team. In this context, a given agent may have to choose some actions that, not representing a direct reward to its individual (local) goals, are required in order to benefit the system as a whole and to achieve the system goal.

A possible definition which encompasses all the three dimensions may be [CFK97]:

> *"Given some task specified by a designer, a system exhibits cooperative behavior if, due to some underlying mechanism (mechanism of cooperation), there is an increase in the total utility of the system."*

A more precise definition of cooperation from Artificial Intelligence is [DFJN97]:

> *"To cooperate is to act with another or others for a common purpose and for common benefit."*

There are two primary ways to give an agent a purpose: the agent is provided with a set of behaviors that are designed in such a way that the agent pursues some *implicit purpose* — *goal-oriented* or purely *behavior-based control*; alternatively, the agent is motivated by *explicit goals* and employs decision-making processes (planning, negotiating, *etc.*) to direct its action towards the achievement of those goals — *goal-directed control*.

Sharing the same purpose (whether implicit or explicit) is not a sufficient condition for agents to achieve explicit or intended cooperation. They must intend also to act together or to have a commitment to joint activity. This is only possible with some goal-directed control because it is not possible without internal state and purely goal-oriented control.

A common definition of Distributed Problem Solving (a branch of Artificial Intelligence) is the cooperative solution of problems by a decentralized and loosely coupled collection of knowledge sources, located in a number of distinct processor nodes [Smi80]. The knowledge sources cooperate in the sense that no one of them has sufficient information to solve the entire problem. Mutual sharing of information is necessary to allow the group, as a whole, to produce an answer. Here decentralized means that both control and data are logically and often geographically distributed and there is neither global control nor global data. Loosely coupled means that individual knowledge sources spend most of their time in computation rather than communication.

The way whereby a group of intelligent agents shares information — the main question addressed in this thesis — is crucial to any of the aforementioned definitions of cooperation. It allows every agents to be aware about what is the global task and to maintain a sufficient level of awareness on each individual agent, in spite of information being dispersed and, sometimes, inconsistent. It is crucial to coordinate the behavior of the individual agents through some coordination mechanism, in order to obtain a joint collaborative behavior and suitable global performance.

Table 2.1: Payoff matrix of the *prisoner's dilemma* game used by Axelrod [Axe84]: T=Temptation to defect; R=Reward for mutual cooperation; P=Punishment for mutual defection; S=Sucker's payoff.

		Player B	
		Cooperate	**Defect**
Player A	**Cooperate**	$R = 3$, $R = 3$	$S = 0$, $T = 5$
	Defect	$T = 5$, $S = 0$	$P = 1$, $P = 1$

2.1.2 The prisoner's dilemma

Political sciences' researchers have already studied the emergence of cooperation using game theory models. The *prisoner's dilemma* is a classic of game theory, which has been used to study interactions based on reciprocal altruism in different areas, such as political and social sciences, economy and biology.

The situation inherent to the prisoner's dilemma occurs when selfish individuals, pursuing their own interests, lead to a poor outcome for the collective. In the prisoner's dilemma game, there are two players that have two different choices in each interaction: cooperate or defect. Each must make the choice without knowing what the other will do. No matter what the other does, defection yields a higher payoff than cooperation. The dilemma is that if both defect, both do worse than if both cooperate.

This game was used by Axelrod to identify under what conditions cooperative behavior emerges in a group of selfish individuals without a central authority, where pursuing self interests does not imply the group welfare [Axe80a, Axe80b, Axe84].

Table 2.1 depicts the prisoner's dilemma game that Axelrod used to pursue his work [Axe84]. One player chooses a row, either cooperating or defecting. The other player simultaneously chooses a column, either cooperating or defecting. Together, these two choices result in one of the four possible outcomes shown in the matrix. If both players cooperate, both get the reward for mutual cooperation ($R = 3$ points). If one player cooperates but the other defects, the defecting player gets the temptation to defect, while the cooperating player gets the sucker's payoff ($T = 5$ points and $S = 0$ points, respectively). If both defect, both get the punishment for mutual defection ($P = 1$ point). The four parameters were chosen so as to get $T > R > P > S$ and $R > (S + T)/2$. These conditions ensure that mutual cooperation gets a higher cumulative payoff in consecutive interactions than alternating between exploiting and being exploited (exploiting each other).

An iterated prisoner's dilemma game (IPD) is a sequence of interactions, whose length is not known by the players. Each player knows the complete history of previous interactions with the other player, but it does not know the decision that will be chosen by the other player in the current interaction.

Given that individuals have a sufficiently large chance to meet again, so that they have a stake in their future interaction, Axelrod used IPD to explore the following general conditions for the evolution of cooperation: firstly, cooperative strategies must have success on a given environment, so that they can be adopted by agents; secondly, these strategies must have success in dynamic environments with learning capabilities (learning agents), so that they can thrive and propagate in a population; thirdly, once cooperation is established in a population on the basis of the reciprocity, it must protect itself from invasion by less cooperative strategies.

Axelrod reported experiments with the IPD [Axe84] wherein different programs, implementing strategies ranging form very simple (*e.g.* a completely random strategy) to mathematically very complex, were confronted in a round robin tournament with a fixed number of iterations (200 iterations each), including confronts of each program with itself. At the end of an iteration, each program summed a score accordingly with Table 2.1. After all rounds, it was summed the overall score accumulated by each program to determine the winner. Surprisingly, the winner program implemented a strategy named “tit for tat” (TFT) which, excluding the program implementing a completely random strategy, was the simplest program.

TFT is a very simple strategy that always cooperates in the first iteration. In the following iterations, it simply does whatever the other player did on the previous iteration: if the opponent defected in the previous iteration, TFT retaliates (defects); if, however, the opponent cooperated in the previous iteration, showing good will or regret, TFT cooperates as a way to establish a reciprocal cooperative relationship, beneficial to both players.

This surprising result led to some conclusions about a successful cooperative strategy: *niceness*, *retaliation*, *forgiveness* and *clarity*. *Niceness* means that the strategy is never the first to defect. *Retaliation* means that it retaliates immediately after its opponent has defected, showing that it is willing to cooperate but not to be exploited. *Forgiveness* means that, after retaliating, punishment is ended as soon as the opponent cooperates. *Clarity* means a strategy that is easily identifiable and coherent, favoring the establishment of a cooperation relationship based on reciprocal confidence.

Further studies with an evolutionary version of the previous experiment [Axe84], whereby several generations of a tournament were simulated, so that more successful strategies in a given generation were more likely to be submitted in the next generation, showed that TFT is an evolutionary stable strategy, which can thrive and protect itself with a cluster of individuals who rely on reciprocity.

These studies put on evidence that groups of individuals ruled by cooperation strategies based on reciprocal altruism yield good overall performance, even if the individuals are intrinsically selfish. Moreover, they show that groups of altruistic individuals may thrive against other individuals not willing to cooperate, which are eminently adversarial. This knowledge inspired the development of the cooperative strategy for sharing useful information presented in chapter 5, which is based on reciprocal altruism [RDC05b, RDC05a].

2.1.3 Biological inspiration

Cooperation between simple organisms on earth is almost as old as life on earth itself. Biologists have long understood that bacteria live in colonies, but only recently it has become evident that most bacteria communicate using a number of sophisticated chemical signals and engage in altruistic behavior [Jun98]. They emit and react to chemicals in a genetically determined way that associates chemical and elicited behavior.

This can be considered an *interaction via the environment*, as the chemical environment, immediately surrounding each bacterium, acts as a communication channel for information implicit in the emitted chemicals that must be sensed and reacted to. These chemical signals only have meaning when interpreted in a behavioral context and they are an explicit signaling and a consequence of the evolutionary history of bacteria [JZ00]. The resulting cooperative behavior emerges as a consequence of the behavior policy genetically encoded in each individual. This mechanism of cooperation is simple as there is no recognition of other individuals, neither explicit communication.

Social insect societies have been thoroughly studied by biologists, especially ants, termites, bees and wasps [BG00]. For example, termites collectively build huge nests and ant colonies plan shortest paths between their nest and a food source, using a powerful signaling mechanism, which is also a kind of interaction via the environment: the exuding of a pheromone — a chemical substance — attracts other ants.

When ants forage food sources, they lay and follow trails of pheromone. The first ants returning to the nest from the food source are those that have taken the shorter path in both directions. Because this route is the first to be doubly marked with pheromone, the other ants are attracted to it and tend to follow the optimized route. Path planning is an emergent characteristic of the ant colony not present at the level of the individual.

Ants also extensively use cooperative mechanisms that involve explicit (intentional) sensing of other individuals. Ants achieve the identification of castes of other individuals using chemicals sensed with their antennae (they cannot identify specific individuals). Although this *interaction via sensing* is a more sophisticated interaction than broadcast style of interaction via environment, the former is built upon the same mechanisms of the latter (chemical signals). The interaction via sensing is built and layered upon interaction via environment. As ants, animals in general, which use more sophisticated schemes, have also more basic schemes upon which the more sophisticated ones are built.

Scientists who study the behavior of social insects have found that, although the individual activities appear seamlessly integrated without any supervision, the group cooperation at colony level is largely self-organized. The coordination simply arises from interactions among individuals. Although these interactions might be simple (*e.g.* following the trail left by another), together they can solve difficult problems (*e.g.* finding the shortest route among countless possible paths to a food source) and emerge a beneficial collective behavior, denoted as *swarm intelligence* [BG00].

The wolves, social mammals of the canine family, are carnivores that usually hunt in packs, formed upon strict social hierarchies and mating systems. They organize themselves through demarcating territories. Territory marking is done through repeated urination on objects on the periphery and within territories. Wolves also communicate with pheromones excreted via glands near the anus and the dorsal surface of the tail. As the chemical trails of ants, these are also examples of schemes based on interaction via environment.

Wolves also interact via sensing when they hunt in packs: they cooperate by closely observing the actions of each other and, in particular, the dominant male who directs the hunt. Each wolf knows all the pack members and can identify them individually, visually and by smell. Wolves can also interact *via explicit communication*, as they communicate explicitly with a particular individual using a combination of specific postures and vocalizations [Jun98].

As with ants, wolves' communication exhibits meaning preservation for the signals, but with a significant difference: the shared grounding that enables the uniform interpretation of some signals (*e.g.* postures and vocalizations) is not wholly genetically determined. Instead, the grounding is partially learnt during development in a social environment similar to both individuals that ensures a shared meaning.

Primates also use each of the three mechanisms referred above — *interaction via environment*, *interaction via sensing* and *interaction via explicit communication*. The main difference between primates and other animals is their sophistication in learning and representing the internal goals, plans and actions of others, and their ability to construct cooperative plans jointly and flexibly adapt and repair them in real time [Jun98].

A joint plan can be defined as a sequence of actions, each enacted by a particular member of the group. Each individual assesses the goals, actions and plans of others, and adjusts its own goals, actions and plans to achieve a more coordinated interaction where joint goals are satisfied.

Non-human primates use extensively the passive observation of others (interaction via sensing), via visual and auditory cues interpreted as actions and intentions. The interaction is simultaneous and occurs within visual or auditory range. The signaling is implicit (side effect of the behavior) but the sophistication of its interpretation is considerable [JZ00]. As with the wolves, the communication also exhibits meaning preservation through a shared grounding. However, the grounding is more complex, as is the development process required to attain it.

Humans own the heritage of our primate ancestors, using many types of signaling for communication [JZ00]. Like primates, we make extensive use of implicit communication, such as posturing and explicit gesturing (*e.g.* pointing), but we also make extensive use of explicit communication, both written and spoken, that is explicitly evolved or learnt. Posturing, gesturing and speaking all involve simultaneous interaction.

Humans have developed symbolical communication, which enables long-term interactions (*e.g.* written language). It requires considerable sophistication in interpretation, but we also use signals that are more easily interpreted, like laughing. Being the shared groundings for human symbolic communication more complex, our cultural language learning can be seen as an extension of the process present in our non-human primate ancestors.

Based on these communication mechanisms, humans display a basic level of altruism and cooperate in many and varied ways toward humans and sometimes animals, fostering symbolic contracts with mates, kin, friends, organizations and societies, whereby we exchange resources for mutual benefit. In some cases, we cooperate and provide resources with no immediate reward, except the promise that the other party will honor the contract and will provide resources when we need them.

As a conclusion of the previous biological examples, we may say that the sophistication of cooperation increases as we go from bacteria cooperation to primate and human cooperation, and this seems to have a high correlation with the increase of the sophistication of the communication schemes. It is likely that sophistication of cooperation scales with that of communication [Jun98].

Parker made a broad classification of animal societies, which has particular interest for research in cooperative robotics, as it parallels two possible approaches to cooperative systems development [Par94]. She grouped animal societies into two broad categories: those that *differentiate* and those that *integrate*.

Insect societies are an example of societies that *differentiate* because they arise due to an innate differentiation of blood relatives that creates a strict division of work and a system of social interactions among members. Members are formed within the group according to the needs of the society. The individual exists for the good of the society and is totally dependent upon the society existence. A group can make accomplishments that are not possible to achieve individually.

On the other hand, societies that *integrate* depend upon the formation of groups of individuals that are independent animals to each other. Such groups do not consist of blood relatives that stay together, but instead consist of individuals of the same species that come together by integrating ways of behavior. These individuals are driven by a selfish motivation that leads them to seek the group life, because it is in their own best interests. Wolves that hunt in packs are an example of this kind of cooperative societies. Another example is breeding colonies of many species of birds, in which birds do not come together due to any blood relationship, but instead they thrive the support provided by the group. Rather than the individual existing for the good of the society, these societies exist for the good of the individual.

The concept of cooperation is a human, and possibly animal, symbolic concept whose meaning is intimately related to the behavioral and cultural references to which it is grounded [Jun98]. For this reason, the specific mechanisms that we find employed for cooperation in the animal and human sphere depend on behavioral tendencies that effect the willingness to cooperate.

The conditions under which organisms cooperate are complex and closely tied to the ecology of individual genes. Although the design of cooperative human-made systems are in a different context, knowing why organisms cooperate can help to identify particular conditions under which those systems may benefit from cooperation.

The most obvious question that arises when trying to understand why biological organisms cooperate is why do they cooperate, knowing that Darwin's natural selection theory implies that they behave in a completely selfish manner to increase their own fitness. Although individuals are genetically selfish, the main mistake of Darwin's theory is the belief that they are necessarily selfish. There are three main reasons that explain why cooperation emerges in biological societies: pair bonding, kin selection and reciprocal altruism [Jun98]. The first two reasons have a genetic basis, whereas the latter explains cooperation between unrelated individuals.

Sexual reproduction is an evolutionary advantage because it allows for faster adaptation to changing environmental conditions. Long-term *pair bonding* provides a willingness of males and females to cooperate to achieve a variety of tasks related to secure reproductive opportunities and child rearing.

Based on the application of Darwin's theory, the theory of *kin selection* is a selfish-gene approach that postulates that individuals cooperate to varying degrees with kin because they have genes in common.

Reciprocal altruism is the process by which altruistic relationships arise between unrelated individuals. A given altruistic relationship is an *evolutionary stable strategy* if: the cost of an altruistic act is low in relation to the received benefit; individuals are able to recognize each other as individuals and to keep track of their history of previous dealings; the group is stable, giving the individuals the chance to encounter each other repeatedly in situations that present opportunities for altruistic acts [Jun98]. Two individuals can profit by forming a relationship based on reciprocal altruism because it provides the opportunity to barter resources and information for mutual benefit.

Human societies encourage a basic level of altruism through cultural controls over behavior, such as legal systems and social conventions. Moreover, this basic level of reciprocal altruism dictates how people expect other people to behave and is often inherent in what is meant when they talk of cooperation. Obviously, it is unlikely that human-made cooperative systems can cooperate by pair bonding or kin selection, but they can benefit from displaying reciprocal altruism toward each other, and toward humans.

This section has shown how cooperation emerges in biological systems. The main ideas are: cooperation sophistication increases with that of communication; and, likewise in many biological examples, teams of robots may benefit from cooperation through reciprocal altruism, as a means to barter resources and information. In chapter 5, this thesis shows how a set of robots may use efficiently explicit communication to share useful information, by following a cooperation scheme based on reciprocal altruism [RDC05b, RDC05a].

2.2 Cooperative multi-robot systems

We've seen in the previous section that cooperation studies in human societies and biological systems may be a useful source of inspiration to develop human-made cooperative systems displaying reciprocal altruism. Multi-robot systems are certainly an important family of these human-made systems, since their successful deployment have unquestionable social and economical relevance in many application domains. Besides making clear the motivation to study and develop multi-robot systems, this section gives the reader some background about those systems and covers a significant part of their current state of the art.

2.2.1 Advantages and motivation

One of the key driving forces in the development of mobile robotic systems is their potential for reducing the need for human presence in dangerous applications, in which human casualties are possible or even likely. Examples of such applications are: the cleanup of toxic waste, nuclear power plant decommissioning, planetary exploration, fire fighting, search and rescue missions, security, surveillance, *etc.* In these applications, it is desirable to reduce the risk to humans through the use of autonomous technology. In manufacturing systems, although being less risky, the use of autonomous technology may increase the work efficiency, due to the highly repetitive and monotonous nature of the inherent tasks.

One alternative for those autonomous systems is to create a single robot-based solution. This robot would have all the capabilities necessary to accomplish the specified mission on its own. This solution may be feasible for small-scale applications, however it is impossible or disadvantageous for the real world applications referred above.

Usually, solutions for those applications employ the use of multiple human workers cooperating and complementing each other [Par94]. Some tasks, which are typified by the high potential for damage to individual collective elements, seem to be ideally suited to multi-robot systems, and thus it is the expendability of collective elements that is identified as the major reason for proposing robot collectives for the task [DJM02].

For some specific robotic tasks, such as exploring an unknown planet [AB98b], pushing large objects [Par94, MNS95, RDJ95], or cleaning up toxic waste [Par98], it has been suggested that, rather than sending one very complex robot to perform the task, it would more effective to send a number of smaller and simpler robots. Such a collection of autonomous agents is sometimes described as a *swarm* [JLB94], a *colony* [DMC96], or as

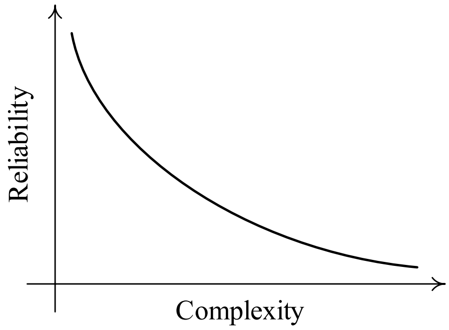


Figure 2.1: Tradeoff between reliability and complexity. Figure reproduced from [Jun98].

a *collective* [KZ94], or the robots may be said to exhibit *cooperative behavior* [Par93].

There are some tasks that are more effectively performed by a single robot, namely those that neither are spatially distributed nor require some sort of synchronization [DJM02]. However, this is not the most usual case and there are several reasons why a multi-robot solution performs better than a single robot [AB98a]:

- *Space Distribution* — many agents can be in many places at the same time;
- *Parallelism or Time Distribution* — many agents can do many, perhaps different, subtasks at the same time;
- *Divide and Conquer* — certain problems are well suited for decomposition and allocation among many agents;
- *Cost, Reliability and Robustness* — often, each agent in a team can be simpler than a more comprehensive single agent solution.

Spatial and time *distribution* is related with simultaneous operation, respectively in space and time. It is whether required by the application — *e.g.* surveying a wide area or tracking moving targets — or it enables better performance in tasks intrinsically distributed, such as cleaning an area or performing a large-scale assembly in less time. The latter reason means that, although some tasks do not require a multi-robot solution, it is very difficult to realize a single robot simultaneously complex and robust [Jun98], because there is usually a tradeoff between performance and reliability (see Fig. 2.1).

Multi-robot solutions give greater flexibility in managing complexity by distribution of risk. For example, instead of building a monolithic robot designed to have all the sensing, perceptual and reasoning capabilities required for a particular task, a multi-robot system can be a much more reliable solution. If there is an overlap in the individual robot's capabilities, then the system has a greater robustness, because a failure of any particular robot will not necessarily mean the failure of the whole system.

Tasks that require traditionally multi-robot systems are typically parallelized and require small amounts of coordinating communication (*e.g.* pushing a large box). Between this extreme and the tasks that are more suited to a single robot, there are tasks that could be performed faster or more reliably with a collective rather than with a single robot, *e.g.* finding a particular object in a finite region, performing a search and rescue mission or building a volumetric map (see Fig. 2.3 in page 78).

Collectives of simple robots may be simpler in terms of individual physical design than a larger and more complex robot. Thus, the resulting system may be more economical, more scalable and less susceptible to overall failure [DJM02]. These distributed systems are more *scalable* and can be more economical because more robots might perform faster missions and adding more robots with complementary features might give to the team the ability to perform more complex missions.

This suggests another dimension of MRS, which is *heterogeneity*. Heterogeneous MRS [Par94, Par98, Par03, Bal00, SSH$^+$02, MLT$^+$02, MFP04, HPS04] are interesting because complex tasks are partitioned into sub-tasks requiring different capabilities and behaviors, which are further performed by robots with different capabilities. Using both heterogeneity and redundancy, multi-robot systems can achieve better cost efficiency, reliability, robustness and performance, even in complex tasks.

2.2.2 Application fields

As other automatic and autonomous systems (*e.g.* a single robot or a computer numerical control-based milling machine), a multi-robot system (MRS) is useful for assisting or substituting humans with autonomous technology [Par94].

As it was already noticed in the previous section, multi-robot systems allow to take advantage of collective elements' expendability in applications with high damage potential. Unless a MRS is comprised of completely heterogeneous robots in terms of their sensing and motion capabilities, it usually integrates some degree of redundancy which yields robustness and graceful degradation when failures of individual robots occur. Thus, a MRS can be very useful for missions carried out within hazardous environments for human beings, where human casualties are possible or even likely. Examples of this kind of applications are: cleanup of toxic waste, nuclear power plant decomissioning [MMO$^+$98], planetary exploration [Vol99, MS01, SHP$^+$01, OMW$^+$04], fire fighting [MCdDO05], search and rescue [CM03, KT01], mine clearing [ZSAC01, GMGH04], military reconnaissance, *etc.*

For instance, Casper and Murphy [CM03] described the use of robots for assisting humans in the search and rescue operation carried out in the World Trade Center towers, immediately after the September 11, 2001 catastrophe in New York, USA. This work were mainly focused on studying human-robot interaction in a real search and rescue scenario, since a huge amount of human rescue teams were required to face the situation and mobile robot technology was not advanced enough at that time for fully autonomous and regardless rescue robots.

Robots were mainly tele-operated by humans and were employed with two main goals: in the rescue phase, they were used to search for victims and examine voids that could not be examined by humans; in the inspection phase, they were used to examine areas beneath the rubble pile under the direction of structural engineers.

The motivation for using robots in this kind of catastrophic scenarios is multifold. Miniature robots can go into places that humans or dogs cannot due to size, extreme heat, toxicity of the environment, *etc.* The operational scenario is dangerous and very risky and, thus, casualties among human rescuers are very likely; on the other hand, robots are expendable and can be deployed very quickly. The amount of trained professionals to perform the multitude of tasks during a rescue — search, extract, examine, inspect, or medically treat — is often scarce and robots can be a valuable help. Furthermore, those tasks may require a group of specialized rescuers for many time; for instance, removing an entombed victim takes an average of ten trained professionals during ten hours [CM03].

The application domain of a cooperative MRS has a strong influence on its design, since there are applications with diverse cooperation and requirements. There are generally two broad types of application domains of multi-robot systems:

- applications wherein multi-robot systems are an advantageous alternative to a single robot;
- applications wherein teamwork is strictly required or using a single robot is not viable.

Table 2.2 gives an overview of research focusing on multi-robot systems, classified by testbeds and application domains.

In the first type of application domains enumerated in Table 2.2, tasks can be accomplished with a single robot, though better performance can be attained with a MRS. In these tasks, if all robots work independently and are unaware of the existence of other

Table 2.2: Overview of testbeds and application domains of multi-robot systems. Each line of the table associates testbeds with a given set of real applications for which they are relevant.

	Testbeds	Application domains
Achievable with a single robot, though multi-robot systems may yield better performance	coverage [BA94, Cho01, CMKB04]	snow removal, lawn mowing, car-body painting, painting a wide wall, cleaning a room
	foraging [Par94, BA94, RDC03, MLT+02, JZ00]	mine clearing [ZSAC01, GMGH04], cleanup of toxic waste
	map building [SF93, Yam98, BMF+00, BMW+02, FMP02, RDC05d, RDC05b, RDC05c, RDC05a], robust and distributed localization [FBKT00, Thr01, RB02, SHB+02, WGD+02, MFP04, RR04, SDH+04, MPS05], SLAM [DNC+01, THF+03, HJSL04, HBFT03, FJC05, NMS05, BR05, SGB05]	planetary exploration [Vol99, MS01, SHP+01, OMW+04], mapping indoors, outdoors, buried utilities [BVS02] and abandoned mines [MDDW98, HV03, THF+03], nuclear power plant decomissioning [MMO+98]
Teamwork is strictly required	multitarget observation [Par99, Tou00, WM00, GM02, VSK+02, SOD+02, FLS02], formation control [BA98, BH00, FM02, ÖEH02, TPK04, MVB04], robot soccer [VSHA98, VUF+98, SV99, SRV00, LVAC99, SHB+02, WGD+02, RM02, HRW+03, BHK+03, BSH+04, SBV04, MRLD05, RBL+05], RoboCup rescue [KT01]	surveillance, security patrols and reconnaissance [SOD+02, FLS02, UA04], motion coordination in industrial applications [BSWL04], fire fighting [MCdDO05], agricultural coverage tasks, search and rescue [CM03]
	box pushing [Par94, Par95, MNS95, RDJ95, GM02], object transportation [Seq99, SSH+02, PPCC02, YAOA03, BSWL04, CKC04, HTOA+04]	transporting heavy objects in industry, assembly of large-scale structures [SSH+02], transshipment operations in large facilities [AFH+98]

robots, the robotic team is a simple extension of a single agent solution. This kind of weak cooperation, without an explicit coordination mechanism, is simply an emergent property. However, although explicit cooperation and coordination among robots is not strictly required, it enables to achieve much better performance.

Examples of these tasks are: demining, snow removal, lawn mowing, car-body painting, painting a wide wall, cleaning a room, exploration of unknown and hazardous environments (*e.g.* planetary exploration), *etc.* Typical research testbeds for this kind of applications are foraging [Par94, BA94, RDC03, MLT$^+$02, JZ00], coverage [BA94, Cho01, CMKB04] and exploration.

Map building (see Fig. 2.3 in page 78) and distributed simultaneous localization and mapping are two typical testbeds that address the exploration problem. While in the former case the emphasis is put on fusing sensory data and active sensing [SF93, Yam98, BMF$^+$00, BMW$^+$02, FMP02, RDC05d, RDC05b, RDC05c, RDC05a], in the latter case the emphasis is put on robust localization [FBKT00, Thr01, RB02, SHB$^+$02, WGD$^+$02, MFP04, RR04, SDH$^+$04, MPS05]. Building volumetric maps of unknown environments was the chosen testbed to investigate important issues in this thesis [RDC05d, RDC05b, RDC05c, RDC05a], such as representing probabilistic maps, exploration and active sensing, cooperation and information sharing and coordinated exploration.

In order to investigate the tradeoffs between no communication (implicit communication) and explicit communication, and in the latter case, between state communication and goal communication, Balch and Arkin [BA94] performed simulations of three of these loosely-coupled tasks: forage (looking for objects), consume (looking for objects and then do work there removing it) and graze (fully covering an area to consume everything).

Within this work, they proposed the *speedup measure* [BA94], which reveals how much more efficient several robots are than just one in completing a task. If $f(n)$ is the performance function for n robots ([3]), the speedup measure is given by

$$\nu(n) = \frac{f(n)}{n \cdot f(1)}. \tag{2.1}$$

If the speedup measure is equal to 1.0, n robots complete the task with a performance that is exactly n times better than one robot. This is called *linear improvement*, *i.e.* performance proportional to the number of robots. Speedup values less than 1.0 reflect *sub-linear performance* and values greater than 1.0 reflect *super-linear performance*.

[3] It is assumed in this definition that, for a given performance criteria, an higher value of the function f means better performance. For instance, if the performance measure is related with the mission execution time, the function $f(n)$ could be defined as the inverse of the mission execution time with n robots.

When agents execute similar tasks in parallel, working independently and without a coordination mechanism (emergent cooperation), the system performance depends on whether exist resources contention (traffic contention, shared tools, *etc.*) and goal conflicts (interference) or not. In the former case, the performance speedup will be sub-linear, whereas in the latter case the performance will be linearly speedup. In the former case, efficient resources contention mechanisms or coordination mechanisms to avoid interference should be used to attain better speedup performance.

In the second type of application domains enumerated in Table 2.2, tasks necessarily require tight coordination, strong cooperation and, usually, explicit communication. The achievement of these tasks when robots work independently will be always accidental and very unlikely. Examples of these tasks are: surveillance and reconnaissance [SOD+02, FLS02, UA04], transporting heavy objects in industrial environments, assembly of large-scale structures (*e.g.* terrestrial buildings, planetary habitat) [SSH+02], transshipment operations in large facilities (harbors, airports, marshalling yards) [AFH+98], motion coordination in industrial applications [BSWL04], search and rescue [CM03, KT01], *etc.* Typical research testbeds for this kind of applications are multitarget observation, box pushing, object transportation, formation control and robot soccer.

Multi-target observation consists in maximizing the time during which moving targets are observed by, at least, one of the robotic agents [Par99, Tou00, WM00, GM02, VSK+02, SOD+02, FLS02]. The multi-target observation task is similar to the foraging task with the addition of dynamic targets that must be continuously tracked. It has many similarities with security, surveillance and recognition problems.

The box-pushing task [Par94, Par95, MNS95, RDJ95, GM02] requires tight coordination among robotic agents and has analogies with other practical problems, *e.g.* storage and retrieval or truck loading and unloading.

Object transportation tasks with multiple robots [Seq99, SSH+02, PPCC02, YAOA03, BSWL04, CKC04, HTOA+04] requires tightly coupled cooperation among robots and some kind of communication whether implicit or explicit, in order to successfully manipulate a rigid object (*e.g.* a rigid bar or a box) or efficiently transport a large object requiring several robots (more that two robots). This testbed encompasses problems such as task allocation, coordination, motion planning, obstacle avoidance and synchronized motion control. Implicit communication means, for instance, a robot observing the state of other teammate while performing the task.

Formation control [BA98, BH00, FM02, ÖEH02, TPK04, MVB04] can be viewed as a constrained motion control problem of multiple robots. In surveillance tasks, it allows individual team members of a MRS to concentrate their sensors across a portion of the environment, while their partners cover the rest. In military scenarios, it helps to protect robots from enemy units in a similar way that, for instance, an animal in a herd benefits by minimizing its encounters with predators. Formation maintenance is applicable in many other domains, such as search and rescue, agricultural coverage tasks and security patrols.

A very popular and challenging group of testbeds is RoboCup [KANM98, BDF$^+$02], which is an international initiative aiming at fostering intelligent robotics research. Robotics competitions under the RoboCup initiative have given a significant boost to research work on multi-robot systems for the last decade [VSHA98, VUF$^+$98, SV99, SRV00, LVAC99, SHB$^+$02, WGD$^+$02, RM02, HRW$^+$03, BHK$^+$03, BSH$^+$04, SBV04, MRLD05, RBL$^+$05]. It provides standard problems requiring the integration of important technologies, such as autonomous robots, multi-agent cooperation, real-time reasoning, sensor-fusion, computer vision, mechatronics, artificial intelligence, adaptation and learning *etc.*

The RoboCup's dominant testbeds are robotic soccer and rescue. Both represent robotic applications without human intervention, occurring in adversarial, highly dynamic and uncertain environments, and requiring cooperation among robotic agents.

Robotic soccer is an exciting domain for intelligent multi-agent robotics, requiring real-time sensing, action and decision making in a harsh and dynamic environment, where the teammates of a multi-robot team must cooperate to defeat the adversarial team. Because the decision making involves simultaneously cooperation and competition, robotic soccer is already considered a benchmark for the progress of robotics and artificial intelligence [BHK$^+$03]. Due to its connection with the very popular soccer game, it has been growing very fast for the past few years, with increasing number of competitions and participant teams from different countries all over the world. Robotic soccer competitions are organized along different modalities, which put on emphasis different design aspects (*e.g.* dynamics and speed in the small-size league or sensory perception and localization in the middle-size league).

RoboCup rescue [KT01] aims at specifically promoting research in socially significant issues, through the simulation of search and rescue scenarios, both with mobile robots and software simulations.

2.2.3 Main issues and taxonomies

A multi-robot system (MRS) cannot be simply regarded as a generalization of the single robot case because it is supposed to have an internal organization intended to achieve *coordination* and *cooperation*. A MRS can be defined as a set of robots operating in the same environment, which are *situated*, *flexible*, *autonomous* and *mobile* agents [FIN04].

Situated agents are those that sense and act in a dynamic or uncertain environment. *Flexible agents* are both reactive and deliberative: they are reactive when responding to changes in the environment and deliberative when planning and acting ahead of time. *Autonomous agents* can automatically perform useful tasks without human intervention, even for long periods of time. *Mobile agents* are able to transport themselves within the environment. There are some other agents' properties that may be present, such as being *social* and having *learning* abilities.

Previous sections showed evidence of the many advantages of MRS over a single robot solution. Nevertheless, MRS also present important challenges when operating in dynamic, uncertain and complex domains:

- *Cooperative perception* — understanding the world by fusing noisy observations from multiple robots to build a world model shared by the team, which should be purposeful to the team's overall task and as much accurate and comprehensive as possible.

- *Coordination* — coordinating in order to achieve overall coherence and performance with distributed control, given that each agent has only a partial view of the world and cannot make the best decision alone.

- *Cooperative planning* — decomposing complex tasks in a partial ordered set of subtasks, assigning subtasks to individual agents, conflict resolution and re-planning in the presence of contingencies (*e.g.* arrival of new tasks, failures or deadlocks).

- *Teamwork* — ensuring that robots act in a coherent manner, even when unexpected events occur in uncertain environments, such as: robots' failures; receiving new information that makes current processing obsolete; receving unexpected requests that motivate the abandon of previous requests without maintaining the teamwork consistency; violation of inter-agent synchronization points [Jen95, Tam97].

- *Cooperative learning* — cooperating in order to learn coordinated behaviors and to adapt to uncertain and dynamic environments.

Most of these issues are addressed throughout this section. Although cooperative learning is an important issue, especially if a MRS has to adapt to a continuously changing environment (not static), it is out of the scope of this thesis and thus it will not be addressed herein.

2.2.3.1 Taxonomies

A key difficulty in the design of cooperative MRS is the size and complexity of the space of possible designs. Thus, an understanding of the many possible system configurations is essential to take principled design decisions [DJM02]. This knowledge provides a succinct description of systems and results reported in the literature and allows to map out the space of possible designs for a collective. This suggests that is useful to find a descriptive taxonomy for describing and classifying MRS along different research axes. There have been a number of efforts to develop descriptive categories for describing robot collectives.

Yuta and Premvuti [YP92] subdivided collectives based on the interactions of collective elements, *i.e.* whether individual elements work towards a common objective or they work independently towards their own objectives.

Arkin *et al.* [ABN93] examined different collectives dedicated to foraging activities (retrieval tasks) along several dimensions, such as reactive vs. hierarchical planning and no communication vs. state communication.

Cao *et al.* [CFK97] made an extensive survey of research in cooperative robotics, presenting a comprehensive set of references. They proposed a taxonomy focused on problems and solutions, along five research axes: group architecture, resource conflicts, origin of cooperation (how cooperation is motivated and achieved), learning and geometric problems (*e.g.* path planning and formation control). As these research axes are highly interdependent and very broad, it is difficult to identify isolated sample points within the taxonomy.

Parker [Par00] presented a survey of research areas distributed along the following topics: biological inspirations, communication, architectures, task planning and control, localization, mapping and exploration, object transport and manipulation, motion coordination, reconfigurable robots and learning.

Balch [Bal02] presented two highly focused taxonomies of MRS, illustrated with practical examples of multi-robot tasks and reinforcement learning configurations: one was a features' taxonomy of the multi-robot task to be accomplished, encompassing duration time, optimization criteria, subject of action, resource limits, group movement and platform dependencies; assuming a reinforcement learning framework, the other one was a

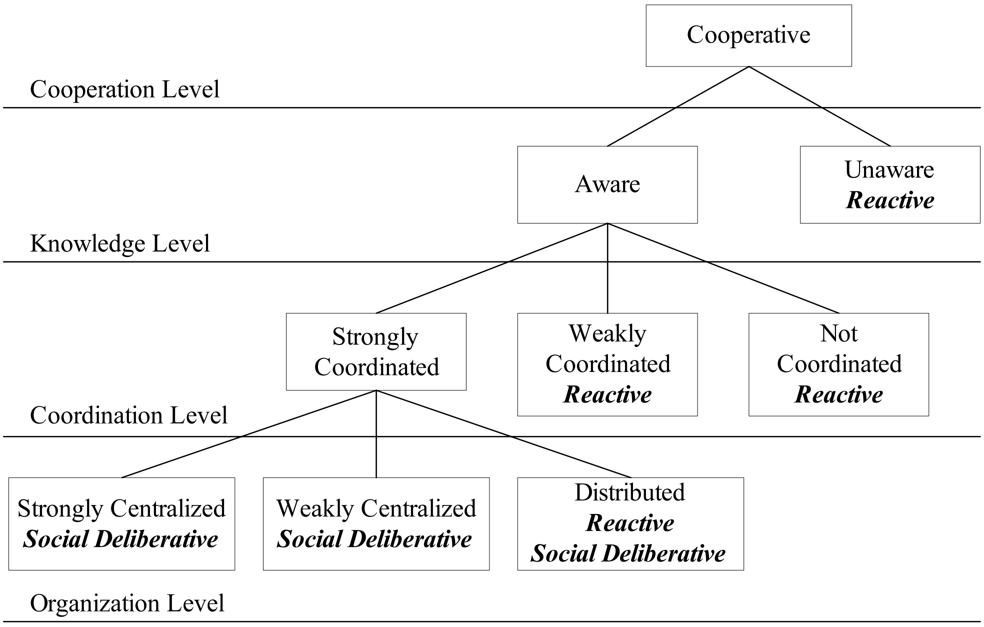


Figure 2.2: Taxonomy of multi-robot systems focused on coordination. Figure reproduced from [FIN04].

taxonomy of rewards, which included source of reward, relation to performance, time, continuity and locality (unique rewards for each individual robot or global rewards).

Dudek *et al.* [DJM02] concentrated on defining a taxonomy whereby different robot collectives could be compared and contrasted. Their taxonomy classifies robotic collectives along seven axes, which address characteristics of the collective as a whole, rather than the architectural characteristics of individual robots. The seven classification axes are: size of the collective, communication range, communication topology, communication bandwidth, collective reconfigurability (*e.g.* switching roles), processing ability of each collective unit and collective composition (whether it is homogeneous or heterogeneous).

Farinelli *et al.* [FIN04] presented a taxonomy of MRS focused on coordination mechanisms, using a top-down approach to refine the level of the system's structure representation. Their taxonomy includes four different representation levels: cooperation level, knowledge level, coordination level and organization level (see Fig. 2.2).

The *cooperation level* is concerned with the ability of the system to cooperate in order to accomplish a given global task. A MRS is considered a cooperative system if the team members operate together in the same environment and have a common goal to achieve.

The *knowledge level* characterizes how much knowledge (awareness) each robot has about the presence of other robots in the environment. If robots are completely unaware, they perform their tasks as if they were the only robots within the environment. Cooperation among unaware robotic agents is the weakest form of cooperation.

The *coordination level* classifies the coordination mechanisms used by robotic agents to take into account the actions executed by the other teammates, in such a way that the group operates in a coherent and efficient manner.

Coordination requires some awareness about others and enables explicit cooperation. Systems with no awareness are necessarily not coordinated. However, coordination is not a sufficient condition to achieve cooperation, *e.g.* robotic agents may coordinate to avoid interference though they have different and independent goals.

On the other hand, for some simple space distributed and highly repetitive tasks, cooperation may emerge in the absence of coordination mechanisms (*e.g.* foraging task). These weakly coordinated MRS tend to be more robust to failures (*e.g.* communication failures), but they lack of many organizational capabilities offered by coordination protocols, which can minimize waste of resources and interference. In general, the more the task or mission is complex the more a strongly coordinated system is required to effectively achieve the goal.

The *organization level* characterizes the way the decision system is organized, *i.e.* if it is centralized (strongly or weakly) or decentralized.

Farinelli *et al.* [FIN04] mentioned two more taxonomic dimensions, which are orthogonal to the previous ones, namely *communication* and *system composition*. The former has a strong influence on the coordination and organization levels (*e.g.* strongly coordinated systems necessarily require extensive communication). The latter is related with robots' differentiation, ranging from homogeneous to heterogeneous systems.

This taxonomy can be used to relate the four classification levels with reactivity and social deliberation. Figure 2.2 puts on evidence how the group architecture of the multi-robot system influences the implementation of reactivity and social deliberation.

Reactivity, which is typical in *swarm robotics*, is a system behavior wherein each team element copes with environmental changes, by providing a specific solution to reorganize its own task and fulfill the accomplishment of its originally assigned goal.

On the other hand, *social deliberation*, which is typical in *explicit cooperation*, is a system behavior that allows the team to cope with those environmental changes, by providing a strategy that, when adopted by all the team members, makes use of all the resources available to the system to effectively achieve the global goal.

In chapter 6, a cooperative distributed model [RDC05c] is proposed which, accordingly with the coordination taxonomy presented in Fig. 2.2, can be characterized as a

Table 2.3: Transverse issues related with multi-robot architectures.

Awareness	• Is each robot aware about the other robots' state and actions? • If so, do it perceive that information or do it use explicit communication? • How do awareness impact on the cooperative team's performance?
Centralized vs. Decentralized or Distributed	• What is the nature of the decision making process? • What agent(s) decide(s) what task shall be executed? • Is there some hierarchy or do every agents equally intervene in decisions? • What are the advantages and pitfalls of the different control approaches?
Differentiation	• Are the robots homogeneous or heterogeneous? • Does heterogeneity impact on the control complexity?

cooperative system comprised of a set of aware agents that are strongly coordinated using social deliberative rules.

Before presenting the basic types of multi-robot architectures, the transverse issues of multi-robot systems indicated in Table 2.3, and their impact on the group architecture, are discussed in the following sections.

2.2.3.2 Awareness

The awareness of an individual robot means its capability to model other teammates' beliefs, goals, states and actions, by using different communication structures. The awareness level of the team members has a strong influence on whether the group architecture is reactive or deliberative, *i.e.* it determines if whether cooperation is an emergent property or explicit.

One way to provide robots with sufficient awareness is to develop sophisticated perception abilities, relying mostly on implicit communication, and to minimize the need for explicit communications among team members. On the other hand, if a robot has limited perception abilities, it can be aware of other agents by constructing a world model mostly based on information explicitly communicated among team members, through a given communication channel.

There have been some studies about the correlation between performance and awareness in cooperative multi-robot tasks [Mat92b, Par95, Tou00]. The main challenge of multi-robot cooperation is to overcome interference and to achieve at least linear (break-even point), or preferably super-linear, improvement in efficiency [Mat92b] (see the definition of *speedup measure* in page 35).

Matarić [Mat92b] addressed the problem of distributing a task over a collection of homogeneous mobile robots in a homing task ([4]). A distributed control approach was applied together with different communication constraints. The main goal was to explore the interaction between computation and dynamics of the individual robots of a collective, taking the most advantage of the dynamics.

The robots either acted in ignorance of one another (no awareness), informed coexistence, or intelligently cooperating with one another. If robots had no awareness, they behaved as they were the only existing robots in the environment, *i.e.* all perceivable objects not related with the task were classified as obstacles (including other robots). In the informed coexistence case, the robots had the ability to sense each other, discriminating obstacles from other obstacles. In the latter case, each robot had a virtual sensor that provided a measure of the local population density and the population gradient.

It was experimentally demonstrated that the ability to distinguish other robots from the rest of other objects in the world (increasing awareness) provided sufficient power to overcome interference, because trading off individual autonomy for collective behavior rendered better efficiency than individual greedy strategies [Mat92b].

Parker investigated how the extent to which robot team members are aware of, or recognize the actions of their teammates, and the extent to which they use this information to effect their own actions, has impact on the cooperative team performance [Par95]. With this purpose, some experiments were performed with collectives whose members whether could or could not be aware of other collective members. Those experiments were puck-moving missions ([5]), varying the number of robots (redundancy) and the level of awareness the robots had of the actions of their teammates.

The study yielded the following conclusions [Par95]: according to an energy metric of performance, performance was improved with awareness, regardless of the team size, because replicated actions were prevented; and redundancy was generally more important than awareness, if a significant part of the mission consisted of tasks whose effects could

[4] Matarić defined a homing task as gathering initially spread robots in a specific location.

[5] Each mission was to locate pucks spread within an area and gather them in a pre-specified location.

be sensed through the world (implicit communication).

Touzet [Tou00] investigated awareness in the context of learning. Learning involves the exploration of the search space to gather information about the task, and exploitation of the data, usually through generalization. The main restriction to the use of learning comes from the size of the search space, which increases almost exponentially in the number of team members, if it is used high awareness and explicit communication. Awareness of other robots implies the addition of several dimensions to the search space.

It was investigated the impact on the search space size of cooperation mechanisms with various levels of awareness, through a cooperative multi-robot observation of multiple moving targets [Tou00]. Each level of awareness was evaluated by the number of inputs on each robot as a function of the number of robots, under the assumption that all robots had at least n sensor inputs (lower level awareness). For the higher-level awareness (complete awareness), each robot had $(n \cdot N)$ inputs when the team had N different robots (each robot was provided with the n inputs of each robot in the team).

The preferable situation would be to provide awareness independently of the number of robots, so as to ensure the scalability of the learning team. In this case, the number of inputs provided to each robot would be $n + \delta$, with $\delta < N$, where δ represents the knowledge about all the other members of the group. Although it was not specified how to obtain such knowledge, it was proposed a generic method for estimating δ as a function of the maximum number of neighbor robots, which depends on the workspace area, awareness range and robot policies [Tou00].

Given that coordination can be viewed as a means to reduce the interference among robots and obtain a globally coherent behavior, if coordination is required by a multi-robot task to achieve an adequate level of cooperation, these studies show that awareness is crucial to implement a coordination mechanism, whether strongly or weakly coordinated. In chapter 6, it is proposed a coordinated exploration method for building volumetric maps with teams of mobile robots. It is made clear therein that coordinating the robots' actions requires an increased level of awareness, which in turn leads to a substantial improvement of the team's performance.

2.2.3.3 Centralized vs. decentralized or distributed systems

The most fundamental decision when designing a group architecture is to decide whether the system is *centralized* or *decentralized* and, in the latter case, whether it is *hierarchical* or *distributed* [Bot00]. This characteristic of the group architecture defines the nature of the decision making process and how the system answers to questions like: "When there are several agents and missions in common, what agent decides what task shall be executed?" "Does each agent decide autonomously about its own role in the system?" "Does the system have some agents that are more specialized for taking such decisions?" Although many practical systems do not conform to a strict dichotomy between centralized and decentralized control, it is interesting to note the advantages and disadvantages of both approaches.

Centralized architectures are characterized by having a single control agent that is individually responsible for the decision-making process. It is presupposed that the central process has a global model of the world that enables it to produce, theoretically, optimal solutions for the problems. We may consider two different classes of centralized systems: *wholly centralized* and *partially centralized*.

In a *wholly centralized* group of agents, each agent receives its future actions from a central processor (agent) and transmits to it local information [BB01]. Apart from the central agent, other agents have not any control autonomy.

In a *partially centralized* system, there is also a central agent, but other agents can also take some autonomous local decisions. Apart the central agent, other agents act locally as central agents and generate local and partial plans, which reduce the complexity of the planning state space. Then, the central agent tries to conciliate the local plans so as to maximize their interaction towards global utility [ER94, ER95, AS97].

A centralized system is intrinsically coordinated, may lead to optimal, coherent and comprehensive solution, but it has numerous shortcomings. Depending on the group dimension, it is very difficult, or even impracticable, to have and maintain a global model of the world on a single agent, based on local and potentially inconsistent views among the local agents. Furthermore, it tends to be a costly solution on time and resources, as it uses massive communication between the local agents and the central agent, which may become a severe communication bottleneck [Bot00]. It has also a high design complexity and low reliability, since all the intelligence is generally concentrated on a central agent. Moreover, for tasks with NP complexity, the centralized approach is impracticable due to the search space's dimension [LH97].

Decentralized architectures are composed of a network of logically and physically independent agents. Each agent is able to reason about plans and decide its own actions [OJ96] and views the system dynamics as being determined by the interactions with and among other agents. Decentralized systems may be classified, by the degree of autonomy conferred to each individual agent, as hierarchic or distributed systems [CFK97].

Decentralized hierarchical systems can be viewed as locally centralized systems, wherein there is a hierarchy of “central agents”. Decisions are distributed across different hierarchical levels and there is a coordinator agent at each level, which can be viewed as a central agent that produces plans for the agents in lower hierarchical levels. Those plans are conciliated through a negotiation involving the coordinator agents [LH97].

Decentralized distributed systems endow all the agents with the same decision power and autonomy. Each agent produces autonomously its own plans as a function of its own goals and ensures coherent behavior through a coordination model [Ben88, Jen96]. In such decentralized systems, each agent is endowed with abilities that enable it to perceive the environment, reason about a complex task, take decisions to accomplish that task and execute plans.

Decentralized architectures have several recognized advantages over centralized architectures, including fault tolerance, reliability, robustness, natural exploitation of parallelism, flexibility and scalability [CFK97]. These systems based on distributed control and distributed data exhibit graceful degradation of performance and better robustness than centralized systems, since there is no a central controller. Therefore, they may be very flexible, since each agent’s role may change with context.

These advantages justify the option presented in chapters 5 and 6 of developing a distributed architecture for building volumetric maps with teams of mobile robots [RDC05b, RDC05a, RDC05c]. Due to the simplicity of the robotic task addressed therein, a decentralized distributed control system is preferred over a decentralized hierarchical system, whose greater complexity would be useless for that application.

Nevertheless, distributing control and data means that knowledge of the system’s overall state is dispersed throughout several entities and each individual has only a partial, incomplete and imprecise perspective. Thus, decentralized systems present an increased degree of uncertainty, making more difficult to attain coherent global behavior. Also, if there is no efficient coordination, the dynamics of such systems can become extremely complex, giving rise to nonlinear oscillations and chaos [Jen96].

Need for coordination in decentralized or distributed systems. In decentralized or distributed systems, coordination is the key for achieving coherent global behavior. *Coordination* can be defined as the process by which an agent reasons about its local actions and the (anticipated) actions of others to try and ensure the community acts in a coherent manner [Jen96].

There are three main reasons why coordination is necessary. Firstly, because there are dependencies between agents’ actions, *i.e.* local decisions have an impact on other agents and there is the possibility of harmful interactions (interference). Secondly, because there is a need to meet global constraints. Thirdly, because no one individual has usually sufficient capacity, resources or information to solve the entire problem.

The main objectives of the coordination process are to ensure that: all necessary portions of the overall problem are included in the activities of at least one agent; agents interact in a manner which permits their activities to be developed and integrated into an overall solution; team members act in a purposeful and consistent manner; and all of these objectives are achievable within the available computational and resource limitations.

In chapter 6, the distributed architecture proposed previously in chapter 5 is refined through a coordination mechanism [RDC05c]. Experimental results with mobile robots demonstrate a significant improvement of the team's performance due to coordination.

2.2.3.4 Differentiation

Another important characteristic of group architectures is *differentiation*. It is frequently dichotomized between homogeneous and heterogeneous groups. A group is *homogeneous* if the capabilities of the individual agents are identical, and *heterogeneous* otherwise. The degree of differentiation of a system has direct implications on the cooperation requirements of its group architecture. Parker introduced the *task coverage metric* [Par94] to give a measure of the number of capabilities on a team that might allow some team member to achieve a given task.

Consider a team of n robots $\mathcal{F} = \{r_1, r_2, \cdots, r_n\}$ and a mission composed of m independent subtasks represented by the set $\mathcal{T} = \{task_1, task_2, \cdots, task_m\}$. Consider also the behavior set $\mathcal{A}_i = \{a_{i1}, a_{i2}, \cdots\}$ of the high-level task-achieving functions possessed by the robot r_i and the set of n functions $\{h_1(a_{1k}), h_2(a_{2k}), \cdots, h_n(a_{nk})\}$, wherein $h_i(a_{ik})$ denotes the task in $\mathcal{T}$ that robot r_i is working on when it activates the behavior set a_{ik}. Parker defined the task coverage [Par94] as

$$task\_coverage(task_k) = \sum_{i=1}^{n} \sum_{j} \left\{ \begin{array}{ll} 1 & \text{if } (h_i(a_{ij})) = task_k \\ 0 & \text{otherwise} \end{array} \right\}. \tag{2.2}$$

Interestingly, if the team members are homogeneous, the task coverage is a positive multiple of the number of robots n. The task coverage index must be more or equal than one for all the tasks of its set $\mathcal{T}$, in order to the system have some likelihood to be able to carry out the mission through efficient cooperation among the team members ([6]). Greater values than one for that index indicate the existence of some redundant resources and

[6] Note that a mission can be viewed as the interaction of several tasks.

overlap in capabilities, thus increasing the reliability and robustness of the team amidst individual failures.

As it is proven in [Par94], task coverage is maximal in homogeneous teams and decreases as groups become more heterogeneous, towards a minimum limit wherein the index equals one for all the elements of set $\mathcal{T}$. The degree of differentiation measured by task coverage may be interpreted as an index of demand for cooperation. When task differentiation is low, tasks can be accomplished without much cooperation, but otherwise cooperation is strictly required in order to accomplish a mission. In general, heterogeneity introduces control complexity, because greater differentiation requires a more effective cooperative task allocation and a greater need to model other individuals.

Nevertheless, it is nearly impossible in practice to build an ideal homogeneous robot team, due to differences in sensor tuning, calibration, *etc.* Moreover, besides those physical deviations, several copies of the same model of robot can vary widely in its behavioral aspects, if it is endowed with learning and adaptation capabilities, which is specially vital in dynamic environments. This means that heterogeneity is usually present in multi-robot teams to some minimum degree.

Balch investigated the impact of diversity, specially behavioral difference, on performance of multi-robot teams, and conversely, the impact of other task factors on diversity [Bal98]. Thus, the degree of heterogeneity was treated by Balch as a result rather than an initial condition. In order to address a quantitative comparison of heterogeneity, a quantitative metric of diversity was proposed, denoted as *hierarchical social entropy*, which provided a continuous scale of diversity [Bal98, Bal00]. This metric was based on the following observation: the measured diversity of a multi-agent society depends on the number of homogeneous subsets that it contains and the proportion of agents in each subset.

Balch adapted Shannon's measure of information uncertainty [Sha49] to a measure of societal diversity and used some notions from biological literature related with numerical taxonomy, like maximum taxonomic distance and clustering methods, to provide a hierarchical metric of distribution of elements in a diversity classification space. When the metric is applied to a given system, a dendrogram ([7]) is created in a continuous scale of diversity (behavioral difference or taxonomic level), usually normalized between 0 and

[7] A *dendrogram* is a taxonomic tree which is frequently used in biology to classify organisms and groups of organisms, at various levels. At the lowest level, organisms are more likely to be classified together (*e.g.* gorillas and humans are both primates but not canines) but, at higher levels, more diverse organisms are grouped together through adequate clustering methods, due to some common characteristic or similarities (*e.g.* primates and canines are grouped in the class of mammals).

1, to provide an orderly hierarchical view of the classification. Integrating the diversity across all taxonomic levels, an overall measure of the system's diversity is produced.

Exploring heterogeneity in multi-robot systems is indeed an interesting issue, because it may increase significantly the system's robustness and utility. For instance, robots with complementary sensory modalities for the same variable (*e.g.* measuring distance with both laser scanners and sonars) may overcome the sensory limitations of individual sensors and can afford more reliable measurements, because if a given sensor cannot measure some feature, other sensors can measure it properly. Moreover, a team comprised of several heterogeneous robots with complementary capabilities has intrinsically a higher utility, because it is potentially capable of performing a wider variety of tasks, requiring different resources (*e.g.* different locomotion capabilities, different sensors, different sizes, different computational power, *etc.*). The differentiation issue is however out of the scope of this thesis, though it may be an interesting future extension of the contributions herein proposed.

2.2.4 Distributed group architectures: reactivity vs. deliberation

Robotics researchers are faced with the task of engineering machines that gather information about their world via sensors, reason and effect action via actuators [Jun98]. This definition led to the classical Artificial Intelligence approach to robotics, known as *sense-plan-act* paradigm, whereby a robot forms a loop containing its sensors and actuators, by sequentially performing perception, modeling, planning, task execution and motor control.

In the 1980s, many researchers began to realize that this approach failed to be scalable to some real environments. A control system for a completely autonomous mobile robot must perform many complex information-processing tasks in real time. Moreover, if that robot is designed to act in a dynamic and complex environment, it must process sensory information change rapidly [Bro86].

After analyzing the computational requirements for a mobile robot and recognizing that an iteration of the sense-plan-act cycle resulted in response times that were too long for many robotic tasks, Brooks proposed a different decomposition of the problem, which has been denoted as *behavior-based control* [Bro86]. Based on a vertical decomposition of the problem, Brooks proposed the *subsumption architecture* [Bro86], whose main properties are:

- *Multiple goals and sensors* — it is a functional decomposition that implements in each layer a level of competence, with higher level layers subsuming the roles of lower level layers when they wish to take control, or they execute concurrently;

- *Robustness* — the failure of a higher level behavior (probably more complex) does not mean that robot fails to execute its task and the robot continues to execute it at a lower level of competence;

- *Extensibility* — the functional decomposition allows to add new layers of control to an existing set, which can be debugged without disrupting the functioning of the lower ones that have been already well debugged.

Matarić defined the three basic types of control architectures [Mat92a, Mat99]: purely reactive, planner-based and behavior-based.

Purely reactive systems, typically encountered in swarm cooperation, achieve rapid real-time responses by embedding the robot's controller in a collection of preprogrammed, concurrent condition-action rules with minimal internal state. Such reactive systems are limited by their lack of internal state, which makes them incapable of using internal representations and learning new behaviors.

Planner-based systems, usually following the sense-plan-act paradigm, are top-down and require the robot to perform a sequence of processing *sense-plan-act* steps, which sometimes compromises their application on real-time applications.

Although *behavior-based* systems, (following the subsumption architecture) are also developed bottom-up, they overcome the reactive systems limitations, because they can store state through its underlying unit of representation: the behavior.

Any control architecture must answer the "what I do next?" question, which is known as the *action selection problem*. In behavior-based systems, this is known as the *behavior arbitration problem* or as *behavior coordination* [PM00]. This is a resource allocation problem because there is usually limited available time, energy, computation, sensors and actuators.

There have been many different action selection mechanisms employed in robot control systems, which can be divided on *arbitration* and *command fusion*. While in the former case a behavior is selected from a group of competing ones and give it ultimate control of the system until the next selection cycle (*e.g.* priority-based, state-based, winner-take-all, *etc.*), in the latter case, mechanisms combine recommendations from multiple behaviors to form a control action that represents their consensus (*e.g.* voting, superposition [Kha86, Ark89], fuzzy [Saf97, SR01] and multiple objective [PM00]).

Since fostering cooperation among robots necessarily requires adequate and effective control of each individual robot, the single-robot control frameworks referred above naturally influence the cooperative multi-robot architectures that have been proposed for the last two decades.

Robotics researchers often distinguish between two types of cooperative research (see Table 2.4): swarm robotics ([8]) and explicit cooperative robotics. While in the former case systems are mainly reactive, in the latter case systems are mainly deliberative. Although these two approaches look to cooperation from different angles, both address the same problem: how to obtain a desired group behavior from a multi-robot system, by engineering the behavior of individuals.

For the distributed architecture model proposed in chapters 5 and 6, the sense-plan-act paradigm could be readily used because the task — volumetric mapping — is by nature sequential, comprising gathering and processing measurements (sense), selecting a new exploration viewpoint (plan) and navigating within the environment (action). Nevertheless, a behavior-based approach was followed, so as to enable concurrent behaviors in some situations. This was especially important to allow the robot to receive and process measurements sent by other robots, while performing other actions (*e.g.* acquiring measurements or moving).

In the next two sections, some important examples of the aforementioned group architectures are briefly presented, so as to better understand the underlying paradigms and make a more detailed comparison of those different approaches.

2.2.4.1 Swarm robotics

Swarm robotics assumes that cooperation is not explicitly designed into the system: cooperation is not predefined but emergent [BHD94, JLB94, KZ94, DMC96]. It may be defined as the distributed control of homogeneous robot teams, whose collective dynamic is obtained as an emergent property of the local interaction between the behaviors designed in the individual robots [CFK97]. These behaviors are often reactive or behavior-based, using mainly local sensory information.

This approach relies on the anti-classical Artificial Intelligence's idea that a group of robots may be able to perform tasks without centralized control or the provision of a global model (explicit representation of the environment and of the other robots) and that predictive planning may be replaced by reactivity. These reactive systems does not present any decision-making process based on decomposition, allocation or accomplishment of

[8] Sometimes also denoted as collective robotics.

Table 2.4: Overview of cooperative multi-robot architectures.

Type	*Paradigm*	*Examples*
Swarm Robotics	Reactive local rules: • Reactive behaviors, using local sensory information (*e.g.* stigmergy) and no explicit communication. • Cooperation is an emergent property of the local interaction between simple behaviors designed in the robots, following a bottom-up approach. • Often inspired in the *eusocial* behavior found in insect colonies.	• Cellular Robotic System [Ben88, HB91, HB92, JLB94] • CEBOT [UF93a, UF93b] • Pursuit game [Kor92] • Group behavior inspired on ant colonies [BHD94, KZ94, KB00] • Synthesis of group behavior [Mat92b, Mat93, Mat94, Mat95] • Formation control using a potential fields method [Ark89, BA98, BH00] • Swarm-Bot architecture [MPG+04, DTS+04]
Explicit Cooperation	Sense-Plan-Act: • Each robot performs sequentially perception, modeling, planning, task execution and motor control. • Mainly deliberative and based on classical principles of Artificial Intelligence, following a top-down approach. • Often, does not deal with robot's concurrent tasks and reactivity (rapid response to unexpected events).	• ACTRESS [AMI89] • GOFER project [CCL+90] • Balance between autonomy and cooperation [YP92] • Architecture with three layers and a distributed blackboard [LVAC99] • MARTHA project [AFH+97, AFH+98, BA99, Bot00, BA00]
	Behavior-Based: • Bottom-up approach with multiple behaviors (goals) and behavior arbitration. • Different layers of competence, using or subsuming the roles of the lower level layers. • Both reactivity and deliberation.	• BLE [WM00, MS01] • Auction-based dynamic task allocation [MS01, GM02] • ABBA [Jun98] • Group theory [Seq99] • ALLIANCE [Par94, Par98, Par02].

tasks. The outputs of each agent are a direct function of the observations sensed on the environment [Bot00].

Swarm intelligence is defined as a "property of systems of non-intelligent robots exhibiting collectively intelligent behavior" [HB91, JLB94]. Another definition [BDG99] states that *swarm intelligence* "is the property of a system whereby the collective behaviors of (unsophisticated) agents interacting locally with their environment cause coherent functional global patterns to emerge".

This concept has inspired some researchers on the development of swarm solutions for complex problems from different areas: the cooperative interaction of ants working to transport a large food item may lead to more effective algorithms for robots; foraging of ants has led to a novel method of routing of traffic in a busy telecom network and new solutions for the classical traveling salesman problem; the way in which insects cluster their colony's dead and sort their larvae can aid in analyzing banking data; the division of labor among honeybees could help make more efficient assembly lines in factories [BG00].

Swarm cooperation is found in nature and it is denoted as *eusocial* behavior. Eusocial behavior is found in many insect species (*e.g.* colonies of ants or bees) as a result of genetically determined individual behavior [CFK97]. Although individual agents are not very capable, intelligent behavior arises out of their interactions and is vital for the survival of the individuals in the colonies. This kind of biologic knowledge about simple local control rules of various biological societies — particularly ants, bees and birds — has been often applied to the development of similar behaviors in cooperative robot systems [SWG04, WMSFS04, SSHH04]. For instance, Wilson *et al.* took inspiration on brood sorting in ant colonies to develop a similar collective behavior in a team of small, simple, homogeneous robots, which used only local interactions without explicit communication [WMSFS04].

Beni *et al.* [Ben88] proposed the concept of *Cellular Robotic System* (CRS). A CRS was a distributed system composed of a large (but finite) number of simple robotic homogeneous units capable of accomplishing, collectively, relatively complex tasks through cooperation. Its main characteristics were reliability and the ability to self-organize and self-repair.

Self-organization in a CRS was the ability to adequately distribute itself for a given task, *e.g.* via geometric pattern formation or structural organization. Interaction took place by each cell reacting to the state of its nearest neighbors. Mechanisms for self-organization were studied in different contexts, such as large-scale displays and distributed sensing [HB91, HB92, JLB94].

Ueyama and Fukuda *et al.* [UF93a, UF93b] proposed CEBOT — *CEllular roBOTics System* — a decentralized, hierarchical architecture based on CRS [Ben88], and inspired by the cellular organization of biological entities. CEBOT was dynamically reconfigurable, because basic autonomous cells (*e.g.* robots) dynamically reconfigured their structure to an adequate configuration in response to changing environments. CEBOT's hierarchy had master cells that coordinated subtasks and communicated with other master cells. The formation of structured cellular modules from a population of initially separated cells was studied [UF93b]. CEBOT's communication requirements were extensively studied and various methods were proposed that sought to reduce communication requirements by enabling them to model the behavior of other cells (*e.g.* [FS94]).

One of the popular domains for the study of cooperative behavior in distributed artificial intelligence is the pursuit game or the predator-prey game, wherein predators try to capture a prey or surround it so that it cannot move anywhere. Korf investigated a simple solution to the pursuit game based on an attraction force to the prey and a repulsive force from the other predators [Kor92]. The main conclusion of his work is that explicit cooperation is rarely necessary or useful in the pursuit game.

Kube and Zhang investigated mechanisms to invoke group behavior, allowing a system of robots to perform tasks as a swarm [KZ94]. They described a solution to the problem of cooperative transport of boxes by a group of robots [KB00], which was based on how ants cooperate in collective prey transport.

The experimental setup consisted of a set of identical mobile robots and various boxes placed along with two spotlights, which were used to indicate final goal positions. In total, over 100 box-pushing trials were run using from one to eleven robots, four different box types and three different venues. The robots did not make use of any form of explicit or direct communication. Given that the individual robots' behavior was based on ant behavior, the dynamics of the swarm of robots was very reminiscent of the emergent cooperative dynamics of ants.

Beckers *et al.* illustrated the concept of *stigmergy* (communication by means of modifying the environment) through the implementation of a robot experiment of collective pile formation [BHD94]. The robotic team was a loosely coupled team without explicit communication, wherein each robot was equipped with infra-red sensors to detect obstacles and a force sensor to detect when more than two pucks were pushed, which were used to implemented very simple behaviors. After being positioned in the center of the work area and oriented randomly, robots started to move pucks. Initially, a few small piles were formed. Gradually, the piles were aggregated, because when a robot detected that it was pushing more than two pucks, it dropped them. By adding pucks to a pile, a robot made the pile larger and was voting for that pile to be the largest. At the end of the experiment, all pucks were in a single pile.

Having the *subsumption architecture* [Bro86] as framework to control a single robot, Matarić addressed the synthesis and analysis of group behavior and learning in complex environments, based on the belief that intelligent collective behavior in a decentralized system could result from local interactions based on simple rules [Mat92b, Mat93, Mat94, Mat95].

Based on the definition of a set of basic behaviors — safe-wandering, following, aggregation, dispersion and homing — a methodology was developed that used basic behaviors to generate various robust group behaviors, like flocking and foraging, through combination operators. The generation of these more complex behaviors tried to maximize the synergy between agents, while minimizing inter-agent interference. Using behaviors as representation unit, a formulation of reinforcement learning was also introduced that allowed a group of agents to learn complex tasks by learning to select the basic behavior set [Mat94].

Balch and Arkin presented a behavior-based approach to robots' formation control [BA98]. The goal was to keep in formation teams of up to four unmanned ground vehicles intended to be fielded as scout units by the United States Army. The formation behaviors were implemented using *schema-based reactive control* [Ark89], which is a form of reactive control that fuses different behavioral outputs through vector summation, in a manner analogous to the *potential fields method* [Kha86]. Balch and Arkin implemented several motor-schemas — `move-to-goal`, `avoid-static-obstacle`, `avoid-robot` and `maintain-formation` — providing robots with the overall behavior of moving to a goal location while remaining in formation, avoiding obstacles and avoiding collisions with other robots [BA98].

Endemic problems of potential fields-based techniques, such as local maxima, minima and cyclic behavior, were dealt with an additional motor-schema — `noise` — which generated movement in a pseudo-random direction. Several robots' formations were considered, namely diamond, wedge, line and column and the approach was demonstrated on simulations, on laboratory robots and on real scout units of the United States Army. The work was further refined to overcome several limitations of the seminal work, such as extending the technique to larger groups and enabling robots to not be assigned to particular locations, but instead to be attracted to the closest position in the formation [BH00].

More recently, Dorigo *et al.* refined the concept of self-organizing teams and proposed *Swarm-Bot* [MPG+04], a distributed swarm-like robotic architecture which aims at deploying swarm robotic teams that are able to adapt to uncertain, dynamic environments and evolve self-organizing behaviors [DTS+04].

Swarm robotics is probably a promising approach in many potential applications based on new robot technologies characterized by miniaturization, like small, micro, nano and modular robots [YZD02], which have severely limited sensing and computation. These very small robots operating in large groups or swarms would be capable of performing complex tasks, like moving objects bigger than individual robots, in non-structured terrains and with high robustness, versatility and adapting quickly to rapidly fluctuating mechanisms.

Swarm intelligence approaches have been effective at performing a number of optimization and control tasks, but the systems developed have been inherently reactive and lack the necessary overview to solve problems that require in-depth reasoning techniques and some cognition [BG00]. Although this approach may maximize reliability, it fails to maximize performance, as members of the collective cannot be directed to uncompleted work that they cannot sense directly [DJM02]. For these reasons, a swarm-like approach was not followed in this thesis.

2.2.4.2 Explicit cooperation

Explicit cooperation means that cooperation is explicitly designed into the system through adequate mechanisms. Unlike eusocial behavior, explicit cooperative mechanisms are not motivated by innate behavior, but by an intentional desire to cooperate in order to maximize individual utility. It deals with achieving intentional and more purposeful cooperation among a limited number of typically heterogeneous agents, performing several distinct tasks, but it may be used also with homogeneous teams [Par94]. In the former case, individual agents have specialized capabilities complementing each other.

In contrast with the swarm approach, the agents often have to deal with some sort of efficiency constraints that requires a more directed type of cooperation, because the mission usually requires that several distinct tasks be performed concurrently. Although individual agents are typically able to perform some useful task on their own, groups of such agents are often able to accomplish missions that no individual robot can accomplish on its own.

In these systems, the cooperative dynamic is commonly achieved by planning and based on multi-agent systems' explicit models of teamwork coordination and negotiation mechanisms, using explicit communication [DFJN97]. The negotiation mechanisms rule the allocation of several distinct roles or subtasks to the individual agents and the resolution of conflicts.

This approach usually employs either central control or a mix of central and distributed control, supported on a global model [Jun98]. Besides action recognition, through perception mechanisms based on sensory information, explicit communication using the exchange of communicated messages provides the required awareness for modeling and reasoning about other agents' goals, actions and states. Thus, the reliability and fault tolerance is highly dependent upon the presence of a reliable communications medium with a sufficient bandwidth [Par95].

There are mainly two bodies of research applicable to intentional or explicit cooperation: firstly, several researchers addressed the cooperative problem by using traditional Artificial Intelligence *planner-based approaches*, usually based on *sense-plan-act paradigm*; secondly, *behavior-based approaches* that try to foster robustness and adaptability of the cooperative team, through situated, embodied, and sometimes learning physical robots.

Sense-plan-act based approaches. Asama *et al.* presented the *ACTRESS* (*ACTor-based Robot and Equipments Synthetic System*) robot system [AMI89], whose main objective was to develop the technology to synthesize multiple robotic elements based on the classical *universal modular ACTOR formalism* from Artificial Intelligence. Robotic components were defined as *robotors* (robotic actors), as being autonomous components that have at least two basic functions: an ability to make decisions and an ability to communicate with any other components for parallel tasks, so as to avoid interference and perform cooperative tasks. There were two possible communicating conditions: when a *robotor* acted independently and was only required to monitor the status of other *robotors* with occasional communication; and when a *robotor* executed a task cooperating with other *robotors* and was required to share the control signals with frequent communication. *ACTRESS* also addressed task assignment and path planning among heterogeneous *robotors*.

Caloud *et al.* described the *GOFER* project [CCL+90] whose goal was to control the operation of many mobile robots in an indoor environment (*e.g.* office, shop-floor, airport, *etc.*), using traditional Artificial Intelligence techniques. A *sense-plan-act architecture* was described which included a task planner, a task scheduler, a motion planner and an execution monitor.

In GOFER, there was a central planner which communicated with all robots and had a global view of the system. The task scheduler used a partially centralized method to allocate tasks to robots with respect to task characteristics and robot availability. Robots used a task allocation algorithm similar to *contract net protocol* [Smi80] to determine their roles and generate an instance of a plan, satisfying the constraints included in the plan structure. The motion planner was distributed and once a robot was allocated to a task it was responsible for its own motion planning. The architecture was successfully used with three physical robots performing simple tasks, such as box-pushing and tracking walls in a corridor [CCL$^+$90].

Yuta and Premvuti advised that the main challenge in designing an intelligent robot was to configure the proper balance between autonomy (in terms of decision making) and cooperation [YP92]. Based on this assumption, they proposed three levels for a multi-robot system, wherein there was a common objective and each robot also had its own objectives: a task specifying level, which specified the common objectives of the system and assigned subtasks or roles to each robot by a centralized decision making process; a robot objectives level, wherein a robot worked to achieve its own objectives; and a solving deadlock level, wherein two robots were responsible for solving a deadlock problem (a common objective of the two robots) that was centrally solved by one of them.

Lima *et al.* introduced an architecture for a team of fully autonomous mobile robots having three layers [LVAC99]: organizational, relational and individual layers. Complexity was reduced by the decomposition of team strategies (what should be done) into individual behaviors, which in turn were composed of primitive tasks. The set of behaviors assigned to each robot was designated as the tactics (how to do it) for a given strategy.

The *organizational level* was modeled as a state-machine that established the strategy to be followed by the whole team, given the team and world states. The *relational level* established relationships among robots through negotiation. The robots were endowed with both individual and team goals and negotiated the adequate tactics to pursue the strategy defined by the organizational layer, using concepts inspired on the *joint intentions framework* [CL91]. The *individual level* encompassed all the available robot behaviors and their relations. Each behavior was modeled as a state-machine, being each primitive task a sense-think-act loop [LVAC99].

The world model that provided information to the relational and organizational levels was implemented as a *distributed blackboard*, which could be viewed as a global shared memory updated upon event-based communication [LVAC99]. The architecture was validated in a robot soccer team, which participated in competitions of the RoboCup's middle-size league [KANM98].

Alami *et al.* reported the *MARTHA* project [AFH$^+$98], which addressed the control and management of fleets of autonomous mobile robots for transshipment tasks in harbors, airports and marshaling yards. A decentralized approach was proposed, which required local communication among robots and a low bandwidth intermittent communication with a central station.

In MARTHA project, vehicles were sufficiently autonomous, so as to allow them to cope with unexpected events and obstacles and inaccurate environment models [AFH$^+$98]. This autonomy was accomplished by providing robots with advanced sensory and perceptual capabilities — localization, obstacle detection and modeling — as well as planning and deliberation through local communication and coordination.

In order to refine, plan and coordinate route sections and crossings occupancy, and as well trajectories in open areas, they used the *plan-merging-paradigm* [AFH$^+$97]. Within this paradigm, whenever a robot produced a plan that made use of some kind of resources (*e.g.* trajectories), it had to validate it in the current multi-robot context, by merging it with the other robots' plans. The approach was validated with a large number of emulated robots under a Unix simulator and with three mobile robots within a laboratory.

Botelho and Alami extended previous work within *MARTHA* project in order to accomplish more complex and generic missions [Bot00, BA00], requiring autonomous and deliberative agents with the ability of planning their actions, perform their tasks in a coherent and non-conflict manner, and cooperatively enhance their performance. They claimed that any autonomous multi-robot system should address the decomposition of a mission into tasks (mission planning), the allocation of the obtained tasks among the available robots and the task achievement in a multi-robot context.

They developed *M+ cooperative task achievement* based on incremental planning [BA99] to cope with a multiplicity of uncertainties related with the task execution, such as task re-allocation, various robot planning the execution of redundant actions and identification of actions that could be achieved with better performance by several cooperating robots. The proposed cooperative multi-robot task achievement was demonstrated through a simulating model of a hospital wherein three robots executed servicing tasks [Bot00].

Behavior-based approaches. Werger and Matarić proposed the *broadcast of local eligibility* (BLE) [WM00], a general tool for coordination between robots, which extended the *port-arbitrated behavior* paradigm across a network of robots.

BLE was comprised of three specific ports: local, best and inhibit [WM00]. Each robot made a local estimate of its own eligibility for a given task, which was derived from the robot's own sensors. This eligibility was written to the appropriate behavior's *local port*, which was connected so as to broadcast the estimate to the best port of each behavior of the same name, on every robot on the local network. The *best port* filtered all the incoming messages for the maximum. A comparison was made between the locally determined eligibility (local port) and the best eligibility (best port). When a robot's local eligibility was best for some behavior, it wrote to its *inhibit port*, which was connected so as to inhibit the peer behaviors on all other robots. As this was an active inhibition, if a robot failed to execute a task, the task was immediately freed for potential takeover by another robot.

BLE was validated in a multi-target tracking task [WM00]. It was shown that BLE-based systems are able to dynamically reconfigure themselves in order to allocate resources in response to task constraints, environmental conditions and system resources.

Later, Gerkey and Matarić described a framework for inter-robot communication (interaction via communications) that was used to dynamically allocate tasks in teams of cooperative mobile robots. Towards this end, it was proposed *MURDOCH* [GM02], a principled, resource-centric, completely distributed, publish/subscribe communication model, which makes extensive use of explicit communication. It offers a distributed approximation to a global optimum of resource usage, which is equivalent to a greedy scheduler. The communication model is a broadcast-oriented blackboard model, wherein messages are addressed by content (subject) rather than by destination.

In order to allocate a given task in *MURDOCH*, a simple auction is used, somewhat similar to the *contract nets protocol* [Smi80], whereby each capable agent evaluates its own fitness for the task. The auction's winner is committed to perform the task until success or failure [GM02]. *MURDOCH* was validated in two different task domains: a short-term tightly-coupled cooperative box-pushing task by a team of three robots; and a long-term loosely-coupled multiple target tracking task, with many robots executing a collection of independent single-robot tasks.

Matarić and Sukhatme addressed the problem of dynamic task allocation in a group of multiple robots satisfying multiple goals [MS01], focusing on three studies: a first *opportunistic approach* using *broadcast of local eligibility* (BLE) and mutual inhibition among robots [WM00]; a second *commitment approach* using *MURDOCH*, a task allocation mechanism based on market-based auctions [GM02]; and a third approach for studying the *tradeoff between opportunistic-based and commitment-based* task allocation. All the three approaches to multi-robot coordination used communication among robots through a blackboard.

In the third approach, an emergency-handling problem, inspired on planetary exploration, was used to compare the impact of opportunistic vs. commitment: *commitment* meant that, once assigned, a robot stayed dedicated to handling a particular alarm, until the alarm could no longer be detected; instead, *opportunism* meant that a robot could switch alarms if, for example, it detected another alarm with greater intensity or priority. The study showed that the opportunistic strategy worked significantly better that the commitment-based strategy [MS01].

Jung proposed the *ABBA* architecture — *Architecture for Behavior Based Agents* — which supports the distributed planning of cooperative behavior in behavior-based multi-robot systems [Jun98]. The basic idea was to extend the action planning between behavior elements within a single agent (an action selection problem) to the cooperative action planning between behavioral elements distributed by different agents, which were able to communicate.

The *ABBA* architecture took inspiration in interaction schemes observed in biological systems, which are usually layered. Four layers were implemented for the solution of a cleaning task by two cooperative robots, ranging from the first layer having no awareness and no communication, to the fourth layer having communication to implement cooperation by planning, which considered speech acts as interactions that affected the other robots' actions [Jun98].

Sequeira proposed a behavior-based architecture using the group theory [Seq99]. It was based on the principle that a robot could be controlled by switching between *a priori* defined control laws, denoted as actions. Control laws specified reference trajectories in the robot's (C-space). A state composition operation was defined which gives to the set of actions the structure of an algebraic group. That operation was used to generate the robot's trajectory in the C-space, given a sequence of actions to be executed. The application of the architecture to a cooperative task was demonstrated through the transportation of a rigid bar along a corridor by two mobile manipulators.

One of the most referenced architectures for cooperative multi-robot systems is *ALLIANCE*, which was developed by Parker [Par94, Par98]. It is a behavior-based approach to control small-to-medium teams of multiple, loosely coupled and heterogeneous robots. It is supported in Brook's *subsumption architecture* [Bro86], having three levels of control.

Robots are assumed in *ALLIANCE* to be able to sense, with some probability, the effects of their own actions and the actions of other robots, through perception and explicit broadcast of information. With this purpose, the highest level, denoted as *motivational* *behaviors*, models the impatience and the acquiescence of the robot [Par94, Par98]. *Impatience* measures the attitude of a robot towards the other robots in the team, increasing when the performance of the other robots is such that the task assigned to the system is not being accomplished. *Acquiescence* measures the attitude of the robot towards itself, increasing when the robot is taking too much time to accomplish its own task and recognizes that it may fail.

The intermediate level of *ALLIANCE* contains groups of behavior sets, with mutually exclusive behaviors placed in different sets [Par94, Par98]. At each time, the motivational behaviors level only activates a single behavior set. The lowest level is composed by basic behaviors, or competences, that must be always active (*e.g.* avoiding obstacles). When higher-level behaviors need to take control of the actuators, they subsume the roles of lower level behaviors [Bro86].

An extension was developed — *L-ALLIANCE* [Par94, Par98] — providing reinforcement learning mechanisms that allow robots to dynamically learn to update their parameters, controlling the behavior sets' activation (*e.g.* how fast a robot becomes impatient), based upon previous experiences. The architecture was successfully validated for some cooperative tasks, including box-pushing, puck-gathering, moving in formation, cleaning an initially unknown room [Par94] and multi-target observation [Par02].

2.3 Communication and information sharing

Communication is crucial for group architectures, because it influences the possible modes of inter-agent interaction, the ability of agents to model successfully other agents' mental states, *i.e.* the agents' awareness, and the agent's ability to share information and successfully build a world model — a basis to reason and coherently act towards a global system goal [DJM02].

Furthermore, in order to maximize the team's performance, explicit cooperation usually requires coordination, which in turn requires some form of communication to provide robots with the sufficient level of awareness.

2.3.1 Taxonomies

Cao *et al.* took inspiration in biological systems and characterized three major types of interactions that can be supported in multi-agent and multi-robot systems [CFK97]: interaction via environment, interaction via sensing and interaction via communications.

In the first two types of interaction, communication is implicit, whereas in the latter one it is explicit.

Interaction via environment is the simplest and the most limited type of interaction. Parker denoted it as *cooperation through the world* [Par95]. It occurs when the environment itself is the communication medium (a kind of shared memory or broadcast implicit communication) and there is no explicit communication or interaction between agents. In this case, agents simply sense the effects of other teammates' actions on the world. Some authors denote this approach as *stigmergy*.

Stigmergy stores state in the environment, so that specialized sensors can easily retrieve it (*e.g.* pheromones in nature and obstacle detection in multi-robot systems) [BHD94]. This is an appealing, reliable and robust approach because of its simplicity and its lack of dependence upon explicit communication channels and protocols, which may be fallible and have limited bandwidth. However, it is limited by the extent to which the agent's sensation of the environment reflects the salient states of the mission that the team must accomplish and the effects produced on the environment by the other teammates.

Interaction via sensing, also denoted by Parker as *passive action recognition* [Par95], occurs when an agent knowingly uses its sensing capabilities to observe the actions of its teammates that are within its sensing range. The sensory information is then used to recognize its teammates' actions, goals and plans, through appropriate modeling of other agents and perception. As the agent can only observe the nearby agents, this is a kind of local communication mechanism.

Like interaction via environment, interaction via sensing is appealing because of its lack of dependence upon explicit communication channels and protocols, which may be fallible and have limited bandwidth. However, it is limited by the degree to which an agent can successfully interpret its sensory information, as well as the difficulty of analyzing the actions of other agents and use that information to infer their mental state.

Interaction via communications, *i.e.* using explicit communication, is appealing because of its directness and the ease with which agents can become aware of the actions and/or goals of the other agents, giving access to both local and global information. Nevertheless, it has poorer fault tolerance and reliability than implicit communication mechanisms, in that it can be highly dependent upon the presence of a reliable communication channel for the successful accomplishment of a cooperative mission. Moreover, as it also depends on the communication channel bandwidth, it has worst scalability and extensibility than implicit interactions, because it is not possible to scale to more agents without additional communications overhead.

The information conveyed through explicit communication can be classified along the following classes:

- State — improves the robots' awareness; for instance, specific control data to support some coordination mechanism;
- Goal-oriented — sharing sensory information among robots, which virtually magnifies robots' own sensory and perception capabilities.

Accordingly with [DJM02], there are many possible configurations in the design of explicit communication:

- Bandwidth — high (not restrictive), medium (cost of the same order of magnitude of the cost of moving the robot), low (very restrictive).
- Range — all to all, only with nearest.
- Topology — broadcast, by address, hierarchical (*e.g.* tree-like or other graph).

Communication is either inexpensive in terms of the robots' processing time (high bandwidth or not restrictive), or it may be expensive if communication and the inherent processing time significantly prevents the robot from doing other useful work. The communication bandwidth is related with the questions "What to communicate?" and "When to communicate?"

The communication range is a function both of the communications medium and the robot distribution. Tree-like or address-based communication topologies are likely to be highly sensitive to failure of particular robots in the collective: the failure of a particular robot will isolate robots on either side of the failed node in the hierarchy; addressing implies distinctive roles for individuals, resulting in reduced interchangeability, unless the robots' roles change dynamically based on actions or failures of other members of the collective. Both communication range and topology are related with the question "With whom to communicate?"

2.3.2 State of the art of multi-robot communications

As it was shown in previous sections, implicit mechanisms are common in some biologic systems, such as bacteria and insect societies, whereas explicit communication appears in more complex animals, especially primates and humans. Implicit interaction mechanisms are mainly suited to local rules based control, whereas explicit communication is more suited to global control approaches, eventually based on planning and negotiation. Parker discussed principles for determining the proper balance between local and global control [Par93, Par94], which determines the communication mechanism's requirements. In practice, there is a continuum between strictly global and strictly local control laws.

Global control laws utilize the global goals of the cooperative team or global knowledge about the team's current or upcoming actions to direct individual agent's actions. The global goals of a team indicate the overall mission that the team is required to accomplish, which can be imposed by a central controller (*e.g.* a human), by one of the members of the team, or through planning and negotiation among the team members. Global knowledge refers to additional information that is normally not available to individual agents through their local sensors, but that may be necessary for the cooperative team to achieve the global goals.

Local control laws guide an agent's actions based on the nearby environment of that agent. Such information is usually obtained through implicit communication, using the agent's sensory capabilities, reflecting the state of the world in the agent's neighborhood.

The use of global goals and information enables the implementation of explicit models of cooperative teamwork, eventually more efficient, but it also has some shortcomings. Adequate global information may not be available to achieve the desired global goal. Even with comprehensive global knowledge, an agent may still not exhibit optimal global behavior unless it utilizes all of the available knowledge. Moreover, besides the cost of maintaining this global knowledge across the members of the team, processing it requires time and resources, which are typically scarce in real applications. If the global goals and knowledge change often enough, the agent may not be able to act upon the global knowledge before it becomes out-of-date, or simply may not be possible or viable to maintain real-time global information, due to the limited bandwidth of communication channel [Par93, Par94].

On the other hand, local control laws allow agents to react to dynamic changes in their environment, without relying on preconceived plans or expectations of the world. If local rules of individual agents are carefully designed, global functionality may emerge from their interaction. However, certain global goals cannot be attained through the use of local control laws alone, because those aspects of global goals that have no physical manifestation in the world cannot be acted upon by local control laws [Par93, Par94].

Parker described the simulation of several control strategies along the local versus global spectrum in a formation control experiment, which yielded some general principles and guidelines [Par93, Par94]. If the global goals are known *a priori* at design time, and all information required for an agent to act consistently with the global goals can be sensed locally by the agent at run-time, these goals can be designed into the agents. The more static, reliable, complete known, and easy to use the global knowledge is, the more practical its use in a global control law.

Conversely, the more unknown the global information, the more dependence the team will have on local control, combined with behavioral and environmental analysis to approximate global knowledge. Behavioral analysis may provide a suitable approximation to global knowledge, being particularly useful when the agents possess a fixed set of discernible or communicable actions. Global knowledge should be used to provide general guidance for the longer-term actions of an agent, whereas local knowledge indicates the more short-term, reactive actions that the agent should perform within the scope of longer-term goals [Par93, Par94].

Arkin demonstrated that sometimes cooperation between robotic agents is possible even in the absence of communication, however this is a weak form of cooperation and it may me very inefficient [Ark92]. Arkin *et al.* described a research work whose main goal was to specify reliable, efficient and robust means of interaction between robots [ABN93]. They found that inter-robot communication of state in a multi-robot foraging task improved performance [ABN93].

Balch *et al.* presented the continuation of the previous work [BA94], by describing a number of simulation experiments of three tasks: forage task, wherein an agent wandered about the environment looking for items and then attached and returned them to a specified home position; consume task, wherein an agent wandered about the environment to find items, attached them and then performed work there; and graze task, wherein agents must completely cover or visit the environment. Simulations were constructed using different levels of communication, including no communication, state communication (information concerning the internal state of agents) and goal communication (specific goal-oriented information). These three levels corresponded to the three basic forms of interaction: via environment, via sensing and via communication, respectively.

In these particular experiments [BA94], goal communication was considered more complex than state communication, because the former type was implemented as an interaction via sensing, while the latter one was implemented via explicit communication. The general findings of this work were: communication improved performance significantly in tasks with little implicit communication (*e.g.* forage task); communication appeared unnecessary in tasks for which implicit communication existed (*e.g.* consume and graze tasks); more complex communication strategies offered little benefit over basic communications.

Communication schemes for multi-robot systems are usually proactive when dealing with failure-prone communications, *i.e.* they try to take control actions so as to avoid the occurrence of communication failures. However, this is not possible in scenarios that require the robots to operate in environments wherein they have little or no *a priori* knowledge.

Ulam and Arkin proposed a set of reactive communications recovery behaviors [UA04] based on the schema-based reactive control paradigm [Ark89]. They compared general-purpose behaviors — attraction to waypoints stored during navigation, or attraction to the last known position of the nearest teammate — with context-specific behaviors — attraction to a nearby open space, or attraction to nearby inclines pointing to higher ground — in the context of a surveillance mission in an urban environment. They concluded that general-purpose behaviors yielded faster communication recovery times, especially moving to the nearest teammate.

Within CEBOT framework, Fukuda *et al.* [FS94] studied methods that sought to reduce (explicit) communication requirements by enabling them to model the behavior of other cells (implicit communication).

Tambe presented STEAM [Tam97], a general teamwork model, which included a heuristic that attempted to follow the most cost-effective method of attaining mutual belief in joint intentions [CL91], by managing a tradeoff between communication and team incoherence costs.

Stone and Veloso proposed a method for inter-agent communication, which assumed that agents alternated between periods of limited and "unlimited" communication [SV99] to attain mutual beliefs.

Gerkey *et al.* [GM02] proposed *MURDOCH* — a publish/subscribe mechanism — whereby messages are addressed by content rather than by destination: a data producer tags a message with a subject (or a set of subjects) describing its content and publishes it onto the network; any data consumers who have subscribed that subject (or any subject in the set) automatically receives the message. Whereas unicast communication, *i.e.* point to point requires to send the same message to every recipients, *MURDOCH* allows to send once a message and multiple recipients will receive it.

2.3.3 Discussion

Although previous work on communication structures for multi-robot systems (MRS) has led to some useful conclusions and design guidelines [Par93, Ark92, ABN93, BA94, FS94, Tam97, SV99, GM02, UA04], there is no a principled formalism that can be systematically used to share efficiently sensory data based on information utility assessment, in order to support the efficient use of communication in MRS.

As communication is always limited, either in resources applied to perceive the world or in bandwidth of a communication channel, using efficiently those resources is crucial to scale up cooperative architectures for teams of many robots, without limiting them to simple reactive and loosely-cooperative systems, with very limited or no awareness. Current architectures extensively use explicit communication, not taking care [Par98, Jun98, Yam98, Seq99, MS01], giving low emphasis [KFO+04], or using no principled heuristics to avoid the communication of redundant information.

The work of Grocholsky *et al.* [GMDW03] is an exception to this trend, because they use entropy to define theoretical information measures for predicting the expected information outcome associated with control actions. Although it seems to be a rigorous method to model the information flow within a team of robots, it is not clear how it can be used to share efficiently sensory data within mapping missions, and it is mainly focused on coordination.

Like in biological systems, the cooperation complexity in multi-robot systems scales with that of communication [Jun98] and explicit cooperation requires usually explicit communication, *i.e.* interaction via communication. This thesis studies how a set of mobile robots should interact via communication when building a volumetric map in an unknown environment, so as to attain a desirable cooperative behavior and maximize the team's overall performance.

Current explicit communication taxonomies for multi-robot systems try to encompass thoroughly all the design axes related with the *communication structure* but nothing is said about the *communication content*. Besides the design questions implicit in those taxonomies, it would be worth to answer questions such as: "What is useful to be communicated?" or "What is task-relevant to be communicated?" These questions underlie the main motivation for this thesis, which is to avoid communicating redundant information and to use efficiently communication resources. In chapters 5 and 6, a distributed group architecture is proposed, whose main novelty is to endow robots with a cooperation scheme whereby explicit communication is efficiently used to increase the robots' individual awareness, based on a criteria of information utility.

2.4 Robotic mapping

Robotic mapping addresses the problem of acquiring spatial models of physical environments with mobile robots, which might be used to safely navigate within the environment and perform other useful tasks (*e.g.* surveillance).

Some examples of sensors used for building maps are stereo-vision [MM96, EHBBB97, MMO+98, TK04], range finders using sonars [ME85, Yam98, SB03, TK04], laser or infrared rays [Yam98, BMW+02, THF+03, NSH03], radars and induction [BVS02, HV03], *etc.*

Building cooperatively maps of unknown environments is one of the application fields of multi-robot systems.

2.4.1 Motivation

Robots can be used for building fastidious maps of indoor environments [ME85, MM96, EHBBB97, Yam98, BMF+00, BMW+02, NSH03, SB03, TK04]. They can also be used in construction for building maps of buried utilities (*e.g.* gas pipes, power or communication lines, *etc.*) in order to avoid getting close to buried utilities when placing new utilities underground with excavators, drills or plows [BVS02].

Nevertheless, they are particularly useful on mapping missions of hazardous environments for human beings, such as: underground mines [MDDW98, HV03, THF+03], where updated maps are required to prevent future accidents related with inundations or collapses, but where humans access is too risky or even impossible due to difficult access routes; or nuclear facilities [MMO+98], where monitoring the state of the sarcophagus interior is required by maintenance procedures, but where humans exposure to radiation must be avoided.

Even if the map is not itself the mission's goal, when robots perform other useful missions within unknown or dynamic environments, they usually need to collect sensory data and build physical models of the surrounding environment. This spatial model is then taken as a basis to path planning, efficient navigation and other mission-specific tasks (*e.g.* maximizing the workspace coverage with a team of robots in a surveillance mission, or cleaning a room with a team of vacuum cleaners).

As sensors have always limited range, are subject to occlusions and yield measurements with noise, mobile robots performing mapping missions have to navigate through the environment and build the map iteratively. Some key challenges inherent to map building are the representation problem, the sensor modeling problem, the registration problem and the exploration problem [Thr02].

2.4.2 Representation model

The main problem inherent to the map's representation model is to deal with the high dimensionality of the entities being mapped [Thr02]. While a very detailed 3-D geometric map (whether geometric or grid-based) may require a huge amount of memory, a description based on topological entities, such as corridors, intersections, rooms and doors, may require much less memory to model the same environment, but obviously yields a map with much less detail. There are basically three types of representation models of a map: metric [CL85, ME85], feature-based [LDWC92, LVH05] and topological [KB91].

Metric maps capture the geometric properties of the environment. Chatila and Laumond proposed *geometric maps* as a representation model using sets of polyhedra to describe the geometry of environments [CL85]. The most popular metric representation are grid-based maps, also known as occupancy grids or certainty grids [ME85, PNDW98].

Grid-based maps are a metric representation model, which are widely used [ME85, MM96, BK91, GMDW03, SB03, HJSL04, RDC05d, RDC05a] to intuitively represent distributed spatial information, such as occupancy or, closely related, traversability. They discretise the workspace being mapped in cells with a given resolution. For each cell, it is maintained a probabilistic belief about its state (*e.g.* free or occupied).

Moravec *et al.* built 2-D occupancy grids in their seminal work by using a robot with sonars [ME85]. Later, they extended the occupancy grid technique for environment mapping of 3-D grids, using stereo-vision as primary sensor [MM96].

Borenstein *et al.* developed the vector field histogram [BK91], which is a popular obstacle avoidance method based on 2-D occupancy grids. Vidal *et al.* used 2-D occupancy grids to represent multi-robot team knowledge about obstacles and evaders within a pursuit-evasion game [VSK$^+$02], in order to develop probabilistic pursuit policies for that specific game theoretical framework. Grocholsky *et al.* proposed the integration of a decentralized architecture — Decentralized Data Fusion — with occupancy grids, as a means to combine observations from multiple robots with communication capabilities [GMDW03]. Hygounenc *et al.* reported a blimp project [HJSL04] wherein 3-D space is represented through digital elevation maps, which are 2-D grids associating height with each cell.

The notion of occupancy grid was refined by Stachniss *et al.* to avoid the binary representation of the cell's occupancy and to model it as a continuous value between 0 and 1 [SB03]. They used 2-D coverage maps to perform indoor exploration tasks with a robot equipped with sonars.

Being a map representation that is very intuitive for humans, yielding a detailed and rigorous model of the environment and helping to solve the registration problem are the main strengths of metric maps. Their main shortcoming, especially with grid-based maps, is not to scale well with the map's dimension: generally, the required memory to store the map increases exponentially with the map's dimension if the map's resolution stays unchanged. This dimensionality problem can be attenuated if the map's dimension increase is balanced with a decrease in the map's resolution.

Feature-based maps maintain a collection of landmarks locations and correlate the uncertainty of their mutual localization [LDWC92, LVH05]. The main idea is to extract features from the environment (*e.g.* lines, corners, doors) and then to parameterize them by, for instance, color, length, width, position, *etc.* Detected features are then registered in the map, which models their localization uncertainty.

Detecting features in a structured environment, *i.e.* in an indoor environment, is a quite easy and reliable procedure. But in natural environments, it is quite difficult to find reliably suitable candidates for the selected natural landmarks (*e.g.* bushes, shrubs, trees, hills, cliffs, *etc.*). Feature-based maps' detail is somewhere between metric maps and topological maps, yielding a representation that scales well with the environment's dimension. However, since features detection is dependent on the specific environment, robots' capabilities tend to be biased to a particular environment. Moreover, usually the map cannot be used solely to plan safe trajectories with obstacle avoidance within cluttered environments, since feature-based maps are especially useful to model sparse environments, *i.e.* having a lot of free space.

Topological maps represent environments as a list of significant places (nodes) that are connected via arcs (edges) [KB91]. Arcs are usually annotated with information on how to navigate from one place to another. Thus topological maps are mainly graphs of connectivity and are especially useful for path planning, though arcs' annotation may have also some metric information (*e.g.* the distance between the places connected by the arc). They tackles branched environments and supports the development of loop closing algorithms [SHB04, SGB05].

Topological maps scale well to large environments, since the amount of information that is stored is limited to the description of the places. As in feature-based maps, places are also detectable landmarks. Thus, topological maps suffer from the same shortcomings than feature-based maps, though they provide connectivity information between landmarks. Moreover, topological maps are difficult to be graphically represented and yield ambiguities related with representing the same place more than once, which are difficult to overcome. These problems are even harder if the environment changes very often.

Any of the aforementioned types of representation models have strengths and weaknesses and cannot be used in all situations. When dealing with large-scale space, the way is to combine hierarchically those models in order to achieve the maximum expressive power in a model that is robust to cope with uncertain, dynamic environments.

Kuipers *et al.* proposed the *Spatial Semantic Hierarchy* [KB91] which integrates both metric and topological models in a hierarchy of representations. Local metric information is used to locally perform control laws and to solve ambiguities in an hierarchy of topological representations, modeling connections between places and regions (sets of places). Both metric and topological representations are then integrated in a global metric map, covering all the environment.

Stachniss *et al.* proposed a solution to close loops when building maps of indoor environments [SHB04, SGB05]. They use a grid-based FastSLAM algortithm to build a metric map of the environment. This metric information is abstracted in a topological representation, wherein a place is added when the robot's distance to the nearest place exceeds a given threshold. Both representations are then used to detect and close loops during exploration.

In chapter 4, a probabilistic framework is proposed to represent and update volumetric maps. The main motivation of this thesis is to foster cooperation among intelligent robots, based on sharing useful information and proper coordination. In this context, building maps was just the chosen application domain for those cooperative mobile robots and it was not itself the main research purpose. Also because the proposed approaches were validated in relatively confined environments, a metric approach, more specifically a grid-based approach, was chosen.

Using other representation alternatives is out of the scope of this thesis, though an interesting future extension would be to integrate both metric and topological representations, so as to be able to represent fine detail and, simultaneously, to scale up with larger environments.

2.4.3 Sensor modeling and sensor fusion

An important challenge posed by robotic mapping arises from the nature of the measurement noise. The problem is significantly easier if the noise in different measurements is assumed to be statistically independent. This is a typical assumption on most of the known sensor modeling techniques. It means that if a robot takes more and more measurements, the effects of noise can be incrementally reduced. But this is generally a rough approximation of the sensor modeling, since errors in control accumulate over time and affect the way future sensor measurements are interpreted. For this reason, coping with the noise's statistical dependency is still a key problem in robotic mapping [Thr02]. This issue is however out of the scope of this thesis and is not addressed herein.

As sensor measurements are characterized by uncertainty and noise, most of the robotic mapping algorithms in the literature are probabilistic and rely on probabilistic inference for turning sensor measurements into maps. They usually model explicitly different sources of noise and their effects on the measurements, using the Bayes rule of inference ([9]). Using a Bayes filter, some successful methods for integrating measurements taken at different instant times in a probabilistic model have been proposed, such as Extended-Kalman filters or Expectation Maximization algorithms.

Kalman filters applied to robotic mapping represent noise, the robot's pose and the position of features (*e.g.* landmarks, shapes, *etc.*) through Gaussian probability density functions, and assume that transition functions are linear ([10]). *Extended Kalman filters* [DNC+01, HJSL04, HBFT03, FJC05] accommodate nonlinear state functions through first order Taylor series expansions.

Expectation Maximization algorithms are derived from maximum likelihood estimation and attempt to find the most likely map given the sensor's data, but cannot generally be used in real time, since they require multiple passes through the entire data set [TFB98]. More recent versions allow to perform on-line mapping by finding the most likely continuation of the previous maps under the most recent sensor measurement [Thr01].

Although these methods have received a significant attention for the past few years, since they are out of scope of the research reported herein, they will not be further detailed. In chapter 4, a straightforward Bayesian method is proposed to fuse into a volumetric map measurements taken at different instant times and at different locations, which assumes the statistical independence of the noise of different measurements.

2.4.4 Registration, localization and SLAM

The registration problem deals with registering measurements on a common coordinate space. Robot's autonomous localization is tightly related with mapping, because accurate

[9] Given two random variables X and Y with probability distribution $p(X = x)$ and $p(Y = y)$, the Bayes rule states that $p(x \mid y) = \eta p(y \mid x)p(x)$, being η a normalization factor to ensure that the left hand side is indeed a valid probability distribution [Pap91].

[10] These transition functions models, for instance, the next robot's pose given its current pose.

mapping depends on localization, which in turn relies on tracking the robot's position to distinguishable landmarks identified in the current map, if a global localization scheme is not available. While building a map, the robot has to register measurements obtained from different locations, which requires its ability to localize itself accurately in the map. Moreover, in a multi-robot solution, estimating accurately the robots' relative position is required to register measurements from different robots on a common coordinate space.

There is some recent work on building volumetric maps with a single robot, focusing mainly on the registration problem [EHBBB97, MMO+98, BVS02, NSH03, HV03]. El-Hakim *et al.* presented a mobile mapping system for generating a 3-D model of an indoor environment [EHBBB97], so as to minimize the error propagation due to the registration in a global reference frame of measurements relative to the mobile platform.

Maimone *et al.* developed a robot equipped with a trinocular stereoscopic mapping system for use in post-nuclear accident operations [MMO+98] and focused on the processing of surface meshes provided by the vision sensor and data registration in a global reference frame.

Huber and Vandapel described the problem of sensing and generating 3-D models of an underground mine with data gathered from a laser range finder, focusing on the problem of registering 3-D data sets from different views in a common coordinate system [HV03]. Their registration approach is analogous to assembling a 3-D jigsaw puzzle, with each view being a piece of the puzzle.

The problem of building a map and simultaneously tracking the robot's position on that map is known as SLAM — *Simultaneous Localization And Mapping*. Extensive research has been devoted to SLAM for the past few years and important progress has been achieved [DNC+01, THF+03, HJSL04, HBFT03, FJC05, NMS05, BR05, SGB05]. Most of the proposed solutions are based on the implementation of an Extended Kalman filter (EKF), which correlates localization estimates relative to different landmarks. SLAM is an important research topic, because it provides an integrated solution of localization and mapping for applications where a global positioning system is not available and the robot is subject to accumulation of pose errors during mapping.

Thrun *et al.* [THF+03] approached mine mapping as a SLAM problem but, due to cyclic structure of mines, it yielded difficult correspondence problems. To solve this problem, they used an iterative closest point algorithm, generating 3-D maps by applying scan matching to 3-D measurements after a 2-D occupancy grid map of the mine was obtained.

More recently, the variant FastSLAM [HBFT03] has been proposed, which combines a particle filter for sampling robot paths and an EKF for representing the map. The particle filter implements a Monte Carlo localization algorithm [TFBD01], which is more robust to data association problems than algorithms based on maximum likelihood data association [Thr01].

There are also some efforts to develop multi-robot localization algorithms [FBKT00, RB02, RR04, MPS05] and SLAM extensions to multi-robot systems [Thr01, MR05, How05]. Within these methods, when a robot determines the location of another robot relative to its own, both robots can refine their internal beliefs based on the other robot's estimate and improve localization accuracy.

Fox *et al.* introduced a probabilistic approach based on Markov localization, which has been validated through real experiments showing a drastic improvement in localization speed and accuracy, when compared to single robot localization [FBKT00].

Roumeliotis *et al.* addressed the determination of upper bounds on the position uncertainty accumulation for a group of robots, by using an Extended Kalman filter [RB02, RR04]. Martinelli *et al.* extended this approach, by considering the most general relative observation between two robots [MPS05].

Sujan *et al.* proposed a SLAM-based architecture [SDH+04] to a cliff surface exploration mission with a robot team. Each robot repositioned its sensors using an information-theoretical approach so as to fill uncertain regions of the environment map, based on maximizing the expected new obtained information.

The localization and registration problems are out of the scope and are not addressed in this thesis, by assuming that robots are externally localized through some absolute localization scheme. Therefore, the work presented herein does not fall in the heading of registration, localization and SLAM, because it intends mainly to explore the cooperation between robots through sharing useful sensory information and proper coordination.

2.4.5 Exploration and active sensing

When a robot or a team of robots explore an unknown environment and build a map, the objective is to acquire as much new information as possible with every sensing cycle, so as to minimize the time needed to completely explore it. The goal of any exploration strategy is to answer the question "Where to move the robot(s)?", *i.e.* robotic exploration is the task of generating robots' motion in the pursuit of building a map as fast as possible.

Exploration is a challenging problem because robots have to cope with partial and incomplete models and, in the case of multi-robot exploration, robots have to coordinate their motion in order to reduce their mutual interference and take advantage of simultaneous operation, by sensing simultaneously different regions of the map. Some quantities that are usually traded off are the expected gain of information, the required time and energy to acquire new information, the possible loss of pose information along the way, *etc.* [Thr02]. The problem is significantly harder if the environment dynamically changes very often. In this case, it is difficult to ensure that the exploration process converges on a consistent map.

Bourgault *et al.* used occupancy grids to address the single robot exploration problem as a balance of alternative motion actions, from the point of view of information gain (in terms of entropy), localization quality and navigation cost [BMW+02]. Although they included information gain in their strategy, their formulation was computationally heavy and they were only able to use it off-line for a limited number of proposed destinations.

Yamauchi *et al.* proposed *frontier-based exploration* [Yam98] whereby robots were driven towards boundaries between open space and unexplored regions. They also proposed a decentralized scheme whereby robots shared local 2-D occupancy grids, which were fused with their own local maps in order to obtain a global grid. Each robot explored the environment by selecting the closest frontier cell in its neighborhood.

Burgard *et al.* developed a technique for coordinating a team of robots while exploring the environment to build a 2-D occupancy grid [BMF+00]. Their approach used the frontier-cell concept proposed by Yamauchi [Yam98] and considered a balance between travel cost and utility of unexplored regions, so that robots simultaneously explored different regions. The utility of a region was reduced when a robot selected a target viewpoint whose visibility range covered it. They do not defined an architecture for the team and neither it was clear how robots should interact nor what should be communicated to accomplish the proposed coordination. In a seminal work [BFT97], they used entropy minimization to actively localize a robot by minimizing the expected future uncertainty.

Zlot *et al.* proposed a market-based approach to coordinate the exploration with multiple robots [ZSDT02]. In their approach, each robot generates autonomously a list of target points using some non-optimal heuristic (*e.g.* closest unexplored region), which aims at maximizing the amount of new information. Then, it uses that list to maintain an exploration tour across the points contained on it, which minimizes the cost (*e.g.* traveled distance). When the robot is able to communicate with its teammates, it acts as an auctioneer by sending the positions of target points and their costs (base prices). Each robot also bids on the target points sold by other robots. The auctioning mechanism decides which robot wins the auction, by selecting the bid with the lowest cost. Moreover, when a bidder has explored the target point, it informs the auctioneer to cancel the bid, in order to avoid already explored regions. The proposed method does not formally quantify neither the information gain yielded by each selected target point nor the interference among robots and the mutual information acquired by their sensors.

Ko *et al.* addressed the problem of merging local maps from different robots, with unknown start locations [KSF$^+$03]. Robots that could communicate with each other were arranged in exploration clusters. The robots within each cluster shared a common map and coordinated their exploration using an algorithm similar to the one proposed by Burgard *et al.* [BMF$^+$00]. Before two robots merged their local maps, they actively verified their relative locations, through the implementation of a particle filter and a rendezvous strategy. The solution has been applied within the Centibots project [KFO$^+$04], which deploys 100 robots in unexplored areas to build a map, search for valuable objects and protect the environment from intruders. A similar project by Howard *et al.* has been reporting experiments with a team of 80 heterogeneous robots [HPS04].

Stachniss *et al.* has been developing a 2-D grid-based version of FastSLAM algorithm [SHB04, SGB05], which generates trajectories to actively close loops during SLAM and takes into account the uncertainty about the pose of the robot during the exploration. Whenever this uncertainty becomes too large, the robot re-visits portions of the previously explored area. When the localization uncertainty is low and no loop can be closed, a frontier-based exploration strategy [Yam98] is used.

The exploration and active sensing problem is addressed in chapters 4 and 6 of this thesis. The proposed approach reformulates the frontier-based exploration concept [Yam98] and the coordination concept proposed in [BMF$^+$00], by using entropy to formally measure uncertainty, information gain and mutual information. Results obtained with its implementation in mobile robots demonstrate its ability to converge nicely to maps with lower uncertainty. In chapter 6, a coordination mechanism is also proposed, whose goal is to avoid that a robot senses the same map's regions than other robots, as a means to take maximum advantage from the parallelism provided by multiple robots.

2.5 Summary and discussion

This chapter started by giving some insight about the cooperation concept itself in sociological and biological systems and covered a significant body of research related with cooperative multi-robot systems (MRS), communication and information sharing in MRS, and robotic mapping. The goal was not to thoroughly describe the details of the many

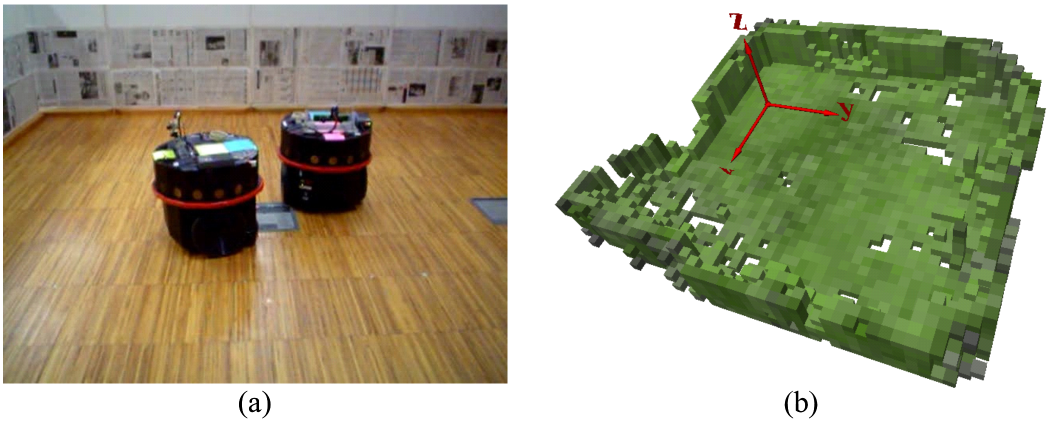


Figure 2.3: Two cooperative mobile robots building a volumetric map at Mobile Robotics Lab., ISR Coimbra [RDC05d, RDC05b, RDC05a]: (a) two Scout robots equipped with stereo-vision; (b) example of a volumetric map.

studies referred throughout the chapter, but to sufficiently present to the reader related work and put properly on context the contributions reported in this thesis.

Following the research question raised in chapter 1, the research presented in this thesis takes a specific application domain of MRS — building volumetric maps (see Fig. 2.3) — to address four topics:

- Representing a volumetric map and its uncertainty;
- Using the partial or uncertain information contained in the map to effectively explore the environment;
- A formal method to assess the information utility, so as to ensure the utility of the information communicated among robots.
- Devising a distributed and cooperative group architecture to share *useful* information among the robots and properly coordinate their actions.

In chapter 4, a grid-based probabilistic model of a volumetric map is proposed, which stores for each cell (voxel) a coverage belief. Concerning this type of metric maps, *i.e.* grid-based maps or occupancy grids [ME85, PNDW98, SB03], the main contribution is a more compact representation of this belief than using histograms [SB03], and a straightforward and efficient Bayesian update procedure. A method to easily update the map upon new data yielded by range sensors was also developed.

The problem of fusing sensory data in a common reference coordinates frame — the registration problem — will not be addressed, by assuming that robots are externally localized. Therefore, the work presented herein does not fall in the heading of registration, localization and SLAM, because the emphasis is to study the aforementioned issues related with efficient information sharing and cooperation between mobile robots.

While most of the research on communication for MRS has been devoted to design axes that are mostly related with the communication structure, another important question mostly related with the *communication content* arises: "What is useful to be communicated?" or "What is task-relevant to be communicated?", which in turn requires an answer to the question "How to assess information utility?" [RDC03, RDC05b, RDC05a].

The goal behind these questions is to avoid communicating redundant information so as to use efficiently communication resources. This is the research question that is addressed in chapter 5, wherein a distributed architecture for building volumetric maps with a team of robots is proposed. Its main features are: supporting distributed control and distributed data; enabling to share efficiently sensory data in a team of cooperative mobile robots, using a measure of information utility; taking advantage from the cooperation among homogeneous robots to build a map in less time than a single robot or than a team with less robots, especially after refining the architecture with the coordination mechanism presented in chapter 6. The main advantage of distributing control and data is to easily scale up the robotic mapping system to an arbitrary number of robots, while maintaining the system's reliability and robustness.

The approach to multi-robot exploration proposed in this thesis is closely related with frontier-based exploration [Yam98] and the coordination concept proposed in [BMF+00], with three important improvements:

- Firstly, the proposed distributed architecture model restricts the communication among robots to the minimum necessary to share useful sensory data among robots (chapter 5) and to coordinate the exploration (chapter 6).

- Secondly, entropy is used to explicitly represent uncertainty in the grid-based probabilistic map, as a means to define a formal information-theoretical background to reason about the mapping and exploration process (chapter 4).

- Thirdly, a coordination mechanism is devised whereby each robot uses the frontier-based criteria while simultaneously minimizes the interference with other robots and avoids to sense the same map's regions than other robots, so as to take maximum advantage from the parallelism provided by multiple robots (chapter 6).

The next chapter provides the reader with the basics of information theory, by presenting the definitions of entropy, differential entropy and mutual information. The goal is to contribute to the readability of chapters 4 to 6, wherein those relations are used to formally define important quantities, such as map's uncertainty, information utility and mutual information associated with two different exploration viewpoints.

Chapter 3

Information theory fundamentals

The mathematical concept of entropy plays a fundamental role in the contributions of this thesis. Although it was developed in 1948 by Claude Shannon as an information measure in the context of computer networks [Sha49], its scope of application is much more wider from a mathematical point of view.

In chapter 4, it is used to measure the uncertainty of a probabilistic belief, *i.e.* the uncertainty associated to probability distributions. In chapter 5, it is used to devise an information-theoretical measure of information utility, which supports efficient sharing of sensory information within teams of robots. In chapter 6, mutual information — a concept closely related with entropy — is used to measure the sensing overlapping associated with two different exploration viewpoints, when different robots explore the same environment to build a volumetric map.

Although the well-known theoretical definitions presented throughout this chapter may be found on text books about information theory [Sha49, CT91, vdL97], this chapter provides the reader with the basics of information theory, so as to contribute to the readability of chapters 4 to 6, wherein knowledge about entropy-related concepts is required and crucial to understand most of the contributions. This is especially worth for readers who are less familiarized with information theory, but this chapter is also important to produce a self-contained document.

3.1 Basics

Information theory is the science which deals with the concept *information*, its measurement and its applications. It aims at characterizing sources of information and quantifying information and uncertainty.

Accordingly with [vdL97], there are two dominant traditions: English and American. The English tradition studies *semantic* aspects of information, which is related to the meaning of messages and their referential aspect. Thus, it is closely related to philosophy, psychology and biology. The American tradition, denoted as *mathematical theory of information*, deals with the *syntactic* aspects of information and its fundamental theorems, being the meaning aspects of information fully abstracted. In 1948, Claude Shannon stated [Sha49]: limits on the amount of information which can be transmitted; limits on the compression of information which can be achieved; and how to build information systems approaching those limits.

Shannon is generally considered to be the founder of the American's tradition in information theory. Nevertheless, there were forerunners who attempted to formalize the efficient use of communication systems [vdL97]. In 1924, H. Nyquist published an article wherein he raised the matter of how messages could be sent over a telegraph channel with maximum speed and without distortion, though the term *information* was not used by him as such. In 1928, R. Hartley tried to define a measure of information, considering messages of l symbols and a choice of s possibilities for every symbols, it was defined as the logarithm of the number of distinguishable messages s^l:

$$H_H(s^l) = \log(s^l) = l \log(s). \tag{3.1}$$

The Hartley's definition accounts with our intuitive ideas: a message consisting of l symbols contains l times as much information as a message consisting of just one symbol; and the amount of information increases with as the number of symbols s increases. The main criticism is the fact of all the symbols having equal chances of occurring. The main achievement of Shannon was to extend the theories of Nyquist and Hartley by proposing a measure — *entropy* — wherein symbols have unequal probability of occurring [Sha49]. This measure associates information with uncertainty using the concept of probability.

3.2 Shannon's entropy

In 1948, Claude Shannon proposed *entropy* as an information measure [Sha49]. Being X a discrete random variable over a sample space $\mathcal{S}$, with a probability distribution $p(x) = P(X = x)$ the entropy was defined as

$$H(X) = -\sum_{x\in\mathcal{S}} p(x)\log p(x) = E\left[\log \frac{1}{p(X)}\right], \tag{3.2}$$

wherein the second form means that entropy can be defined as the expected value of $\log \frac{1}{p(X)}$.

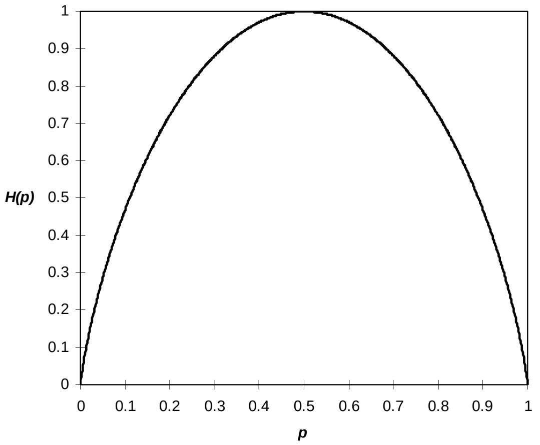


Figure 3.1: Entropy $H(p) = H(p, 1-p)$ of a binary random variable having two outcomes with probabilities $p_1 = p$, $p_2 = 1-p$, $0 \leq p \leq 1$, as a function of p.

In equation 3.2, the logarithm's base determines in what unit entropy is measured. If it has base 2, entropy is expressed in *bits*. Hereafter, the logarithm's base will be omitted and it will be assumed that entropy is always computed by taking all logarithms to base 2. It will be also assumed the convention $0 \log 0 = 0$, since $x \log x \rightarrow 0$ as $x \rightarrow 0$, which means that adding terms with zero probability does not change the entropy.

It is interesting to notice that the Hartley's definition of information measure given by equation (3.1) is equivalent to the Shannon's definition if $p(X)$ is an uniform distribution.

3.2.1 Properties

The Shannon's definition of entropy is related with the definition of entropy in thermodynamics and it has the same mathematical expression. Shannon derived it axiomatically by defining certain properties that an information measure should have [CT91].

Entropy is a continuous function. Being $p_i = P(X = x_i)$ the probability of each possible value of the discrete random variable X, the entropy function H should be continuous in the p_i.

Fig. 3.1 shows the graph of the entropy of a binary random variable, having outcomes x_1 and x_2 with probabilities $p_1 = p$ and $p_2 = 1-p$, respectively, with $0 \leq p \leq 1$. The graph shows that the probability distribution has maximum entropy if it is uniform, *i.e.* $p = \frac{1}{2}$ and $p_1 = p_2 = p$. Moreover, the entropy function is a concave function of the probability distribution, being null when $p = 0$ or $p = 1$. This means that *entropy can be viewed as a measure of how much random a variable is, i.e. a measure of its uncertainty.* Fig. 3.2

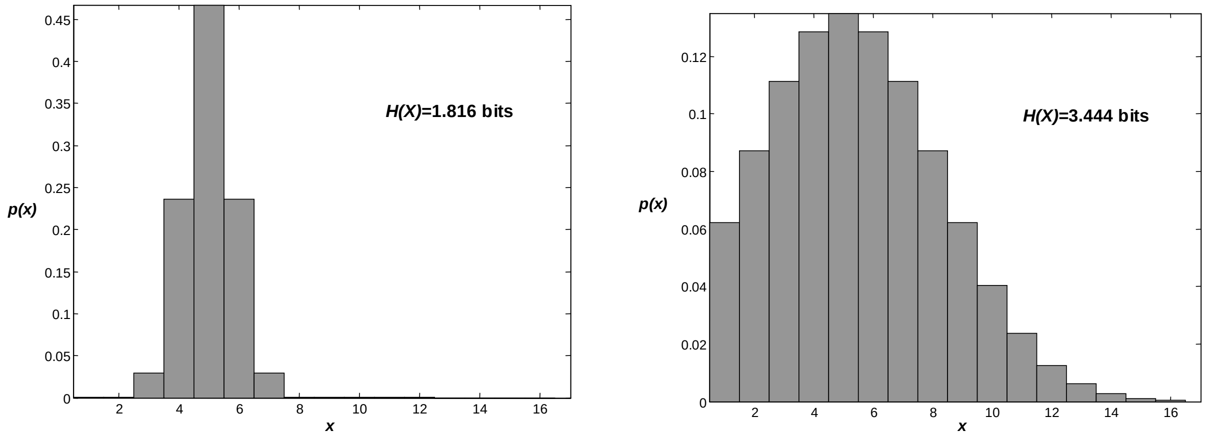


Figure 3.2: Entropy is an absolute measure of uncertainty.

depicts two different probability distributions and their entropy values. The distribution on the right has higher entropy because it presents higher uncertainty (dispersion).

Entropy is a monotonic function and a formal measure of uncertainty. If all samples of a random experiment have the same probability, *i.e.* if $p_i = 1/n$, wherein n is the number of possibilities (the cardinality of the sample space $\mathcal{S}$), H is a monotonic increasing function of n. It can be easily proven from equation (3.2) that the entropy of a random variable X_u uniformly distributed is given by

$$H(X_u) = \log n. \tag{3.3}$$

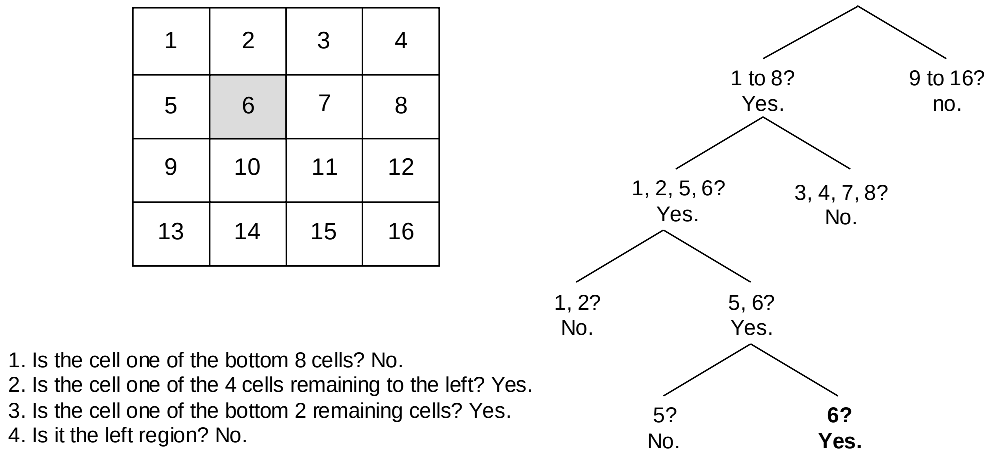
It also can be proven that the entropy of a random variable with n possible outcomes verifies the condition

$$0 \leq H(X) \leq \log n. \tag{3.4}$$

It can be easily proven that entropy is always non-negative, because in each of the terms $-p(x)\log p(x)$ in equation (3.2) the condition $0 \leq p(x) \leq 1$ is always satisfied.

This means that *entropy can be viewed as the average length of the shortest description of a random variable.* For example, in a fair coin toss experiment it is equal to 1 *bit*. As Fig. 3.1 shows, if the binary random variable is not evenly distributed, that shortest description is less than 1 *bit*, being the uniform distribution the maximum entropy case. The minimum entropy $H(X) = 0$ is achieved if $\exists_{i \in \{1,\dots,n\}}$, $P(X = x_i) = 1$, *i.e.* if the variable X is deterministic.

In Fig. 3.2, the entropy of both probability distributions falls somewhere between 0 and 4, since both random variables have $n = 16$ possible outcomes. The maximum entropy



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

1. Is the cell one of the bottom 8 cells? No.
2. Is the cell one of the 4 cells remaining to the left? Yes.
3. Is the cell one of the bottom 2 remaining cells? Yes.
4. Is it the left region? No.

Figure 3.3: Question and answer game: finding the shaded region. Example reproduced from [vdL97].

value $\log 16 = 4$ would be achieved for an uniform distribution. This example makes also clear that the entropy of a discrete random variable is a bounded, formal measure of the *uncertainty* contained on it: it may evaluate to a value between 0 — null uncertainty, deterministic case — and $\log n$ — maximum uncertainty, uniform distribution.

Fig. 3.3 depicts another illustrative example of this qualitative interpretation of entropy [vdL97]. Suppose we have a graphical field consisting of 16 regions, wherein one of them is shaded. The shaded region has to be determined by asking questions which can only answered with a "yes" or "no".

An alternative would be guessing, but then we take the risk of having to ask 16 questions before finally finding the shaded region. Fig. 3.3 suggests that a better strategy would be to ask four selective questions. It can be easily seen that those four questions are always sufficient to determine the position of the shaded region. If we assume that the position of the shaded region is uniformly distributed over the 16 possible positions, its entropy is 4 *bits*. Thus, in this case, entropy corresponds to the minimum number of questions that one must ask to determine which outcome has occurred.

Consider another example with a sample space $X = \{x_1, x_2, x_3\}$ and a probability distribution $P = \{\frac{1}{2}, \frac{1}{4}, \frac{1}{4}\}$. Playing again the answer game with questions "yes" or "no" it seems reasonable to ask for x_1 first, since this outcome has the greatest probability. If the answer is "yes", then the outcome has been found with just one question. On the other hand, if the answer is "no", the the outcome is x_2 or x_3 and another question is needed to find it, *i.e.* two questions. The entropy is in this case given by

$$H(X) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{4}\log\frac{1}{4} - \frac{1}{4}\log\frac{1}{4} = 1.5 \text{ bits},$$

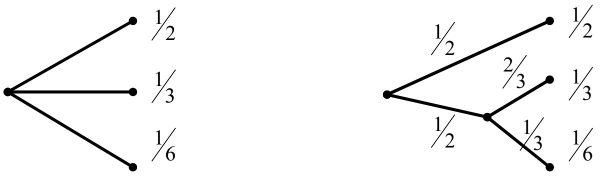


Figure 3.4: When breaking down a choice into two successive choices, the original entropy is the weighted sum of the individual values of entropy.

which is equal to the average number of questions required to find the outcome of the experiment. Thus, the previously given qualitative interpretation is also valid with unequal probabilities.

Entropy is a recursive function. If a choice is broken down into two successive choices, the original H is the weighted sum of the individual values of H.

In the example depicted in Fig. 3.4, both trees present the same probabilities in their leaves. However, on the right, there is a first choice between two possibilities with probability $\frac{1}{2}$ and, when the second occurs, another choice is made with probabilities $\frac{2}{3}$ and $\frac{1}{3}$. It can be easily proven that

$$H\left(\frac{1}{2},\frac{1}{3},\frac{1}{6}\right) = H\left(\frac{1}{2},\frac{1}{2}\right) + \frac{1}{2}H\left(\frac{2}{3},\frac{1}{3}\right).$$

Shannon proved a theorem [Sha49] which states that equation (3.2) is the only function satisfying the three aforementioned properties ([1]).

3.3 Joint entropy and conditional entropy

The definition of entropy for a single random variable, given by equation (3.2), can be extended to a pair of random variables and to conditioned variables, through the definitions of joint entropy and conditional entropy, respectively [CT91].

Let X and Y be two discrete random variables with sample spaces $\mathcal{S}_x$ and $\mathcal{S}_y$ and probability distributions $p(x) = P(X = x)$ and $p(y) = P(Y = y)$. Let also $p(x, y) =$

[1] More rigorously, any function multiplied by a positive constant having the form of equation (3.2) satisfies the three properties. In the entropy definition, that positive constant is usually assumed to be equal to 1 and it is just a scale factor.

$P(X = x, Y = y)$ be their joint probability distribution. The *joint entropy* is defined as

$$H(X, Y) = -\sum_{x \in \mathcal{S}_x} \sum_{y \in \mathcal{S}_y} p(x, y) \log p(x, y). \tag{3.5}$$

Let $p(x \mid y)$ and $p(y \mid x)$ be the conditional probability distributions representing the distribution of X when Y is given and the distribution of Y when X is given, respectively. These conditional probabilities are related with the joint probability distribution through the Bayes rule

$$p(x, y) = p(x \mid y)p(y) = p(y \mid x)p(x). \tag{3.6}$$

The *conditional entropy* $H(X \mid Y)$ measures the entropy (uncertainty) of X when Y is known and is given by

$$\begin{aligned} H(X \mid Y) = \sum_{y \in \mathcal{S}_y} p(y) H(X \mid Y = y) &= -\sum_{y \in \mathcal{S}_y} p(y) \sum_{x \in \mathcal{S}_x} p(x \mid y) \log p(x \mid y) \\ &= -\sum_{x \in \mathcal{S}_x} \sum_{y \in \mathcal{S}_y} p(x, y) \log p(x \mid y). \end{aligned} \tag{3.7}$$

Similarly, it can be proven that

$$H(Y \mid X) = -\sum_{x \in \mathcal{S}_x} \sum_{y \in \mathcal{S}_y} p(x, y) \log p(y \mid x). \tag{3.8}$$

3.3.1 Chain rule

An important result that relates joint entropy with conditional entropy is the *chain rule theorem* [CT91], which can be derived as

$$\begin{aligned} H(X, Y) &= -\sum_{x \in \mathcal{S}_x} \sum_{y \in \mathcal{S}_y} p(x, y) \log p(x, y) && (3.9) \\ &= -\sum_{x \in \mathcal{S}_x} \sum_{y \in \mathcal{S}_y} p(x, y) \log p(x) p(y \mid x) && (3.10) \\ &= -\sum_{x \in \mathcal{S}_x} \sum_{y \in \mathcal{S}_y} p(x, y) \log p(x) - \sum_{x \in \mathcal{S}_x} \sum_{y \in \mathcal{S}_y} p(x, y) \log p(y \mid x) \\ &= H(X) + H(Y \mid X), && (3.11) \end{aligned}$$

wherein equation (3.10) is obtained from equation (3.9) by using equation (3.6).

Similarly, it also can be proven the relations

$$\begin{aligned} H(X, Y) &= H(Y) + H(X \mid Y), && (3.12) \\ H(X) - H(X \mid Y) &= H(Y) - H(Y \mid X). && (3.13) \end{aligned}$$

Equations (3.11) to (3.13) mean that joint entropy is the entropy of one variable plus the conditional entropy of the other. Note that, in general, $H(X \mid Y) \neq H(Y \mid X)$.

The following inequalities may also be proven:

$$H(X,Y) \leq H(X) + H(Y), \tag{3.14}$$
$$H(X \mid Y) \leq H(X), \tag{3.15}$$
$$H(Y \mid X) \leq H(Y). \tag{3.16}$$

For example, the inequality (3.15) shows that the conditional amount of information $H(X \mid Y)$ is generally less than or equal to the marginal amount of information $H(X)$. Given that X and Y are statistically independent random variables if $p(x,y) = p(x)p(y)$, the equalities (3.14) to (3.16) occur when X and Y are statistically independent variables.

3.3.2 Joint entropy of sets of discrete random variables

The joint entropy chain rule theorem given by equation (3.11) can be extended to a set of more than two discrete random variables [CT91].

Consider a set of discrete random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ with a joint probability distribution $p(X_1 = x_1, \ldots, X_n = x_n) = p(x_1, \ldots, x_n)$. The Bayes rule given by equation (3.6) can be recursively used to decompose it as

$$\begin{aligned} p(X_1 = x_1, \ldots, X_n = x_n) &= p(x_1, \ldots, x_n) = p(x_n \mid x_1, \ldots, x_{n-1}) p(x_1, \ldots, x_{n-1}) \\ &= p(x_n \mid x_1, \ldots, x_{n-1}) p(x_{n-1} \mid x_1, \ldots, x_{n-2}) p(x_1, \ldots, x_{n-2}) \\ &= \ldots \\ &= \prod_{i=1}^{n} p(x_i \mid x_1, \ldots, x_{i-1}). \end{aligned} \tag{3.17}$$

Note that for $n = 2$, equations (3.17) and (3.6) are the same equation.

Using equation (3.17), the *joint entropy of a set of discrete random variables* $\mathcal{X} = \{X_1, \ldots, X_n\}$ with a joint probability distribution $p(X_1 = x_1, \ldots, X_n = x_n) = p(x_1, \ldots, x_n)$

can be derived as

$$
\begin{aligned}
H(\mathcal{X}) = H(X_1, \ldots, X_n) &= - \sum_{x_1,\ldots,x_n} p(x_1, \ldots, x_n) \log p(x_1, \ldots, x_n) \\
&= - \sum_{x_1,\ldots,x_n} p(x_1, \ldots, x_n) \log \prod_{i=1}^{n} p(x_i \mid x_1, \ldots, x_{i-1}) \\
&= - \sum_{x_1,\ldots,x_n} \sum_{i=1}^{n} p(x_1, \ldots, x_n) \log p(x_i \mid x_1, \ldots, x_{i-1}) \\
&= - \sum_{i=1}^{n} \sum_{x_1,\ldots,x_n} p(x_1, \ldots, x_n) \log p(x_i \mid x_1, \ldots, x_{i-1}) \\
&= \sum_{i=1}^{n} H(X_i \mid X_1, \ldots, X_{i-1}). \qquad (3.18)
\end{aligned}
$$

Using equation (3.15), it can be proven by induction the inequality

$$H(\mathcal{X}) \leq H(X_1) + H(X_2) + \ldots + H(X_n) = \sum_{i=1}^{n} H(X_i). \qquad (3.19)$$

It is important to notice that the equality occurs when all variables in the set $\mathcal{X}$ are statistically independent. This is an important result that is used in chapter 4 to define the entropy of a probabilistic volumetric map.

3.4 Mutual information

The Kullback Leibler distance, which is also denoted as *relative entropy*, is defined as

$$D(p\|q) = \sum_{x \in \mathcal{S}_x} p(x) \log \frac{p(x)}{q(x)} = E_p \log \frac{p(X)}{q(X)}. \qquad (3.20)$$

It is a measure of the inefficiency of assuming that the distribution is $q(x)$ when the true distribution is $p(x)$, *i.e.* an asymmetric measure of the difference between the two distributions. It can be proven that it is always non-negative, reaching zero if and only if the distributions are identical. The definition given by equation (3.20) assumes the conventions $0 \log \frac{0}{q} \rightarrow 0$ and $p \log \frac{p}{0} \rightarrow \infty$ and that both distributions $p(x)$ and $q(x)$ has the same support set $\mathcal{S}_x$.

Mutual information is a measure of the amount of information that one random variable contains about another random variable or, equivalently, the reduction of the variable's uncertainty due to the knowledge of the other [CT91]. The mutual information

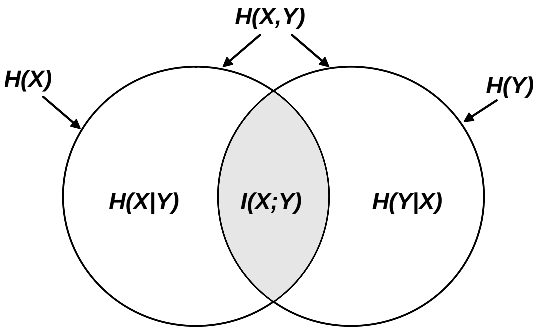


Figure 3.5: Relationship between mutual information and entropy. Figure reproduced from [CT91].

between two random variables X and Y, having probability distributions $p(x)$ and $p(y)$, is defined as

$$I(X;Y) = D(p(x,y)\|p(x)p(y)). \tag{3.21}$$

Thus, it can be interpreted as a measure of the statistical dependence between two random variables.

Mutual information is related with entropy through the relations

$$\begin{aligned} I(X;Y) &= H(X) - H(X \mid Y), && (3.22)\\ &= H(Y) - H(Y \mid X), && (3.23)\\ &= H(X) + H(Y) - H(X,Y). && (3.24) \end{aligned}$$

Equations (3.22) and (3.23) state that mutual information is the information of a variable minus its information if the other is given (see Fig. 3.5). Equation (3.24) can be derived by combining, for example, equations (3.22) and (3.12). Equation (3.22) can be easily derived as follows:

$$\begin{aligned} I(X;Y) &= \sum_{x\in\mathcal{S}_x}\sum_{y\in\mathcal{S}_y} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} = \sum_{x\in\mathcal{S}_x}\sum_{y\in\mathcal{S}_y} p(x,y)\log\frac{p(x\mid y)}{p(x)}\\ &= -\sum_{x\in\mathcal{S}_x}\sum_{y\in\mathcal{S}_y} p(x,y)\log p(x) + \sum_{x\in\mathcal{S}_x}\sum_{y\in\mathcal{S}_y} p(x,y)\log p(x\mid y)\\ &= -\sum_{x\in\mathcal{S}_x} p(x)\log p(x) - \left(-\sum_{x\in\mathcal{S}_x}\sum_{y\in\mathcal{S}_y} p(x,y)\log p(x\mid y)\right)\\ &= H(X) - H(X\mid Y). \end{aligned}$$

Equation (3.23) could be derived in a similar way.

Note that mutual information is symmetric, *i.e.*

$$I(X;Y) = I(Y;X) \tag{3.25}$$

Accordingly with [CT91], given the Jensen's inequality, which states that $E[f(X)] \leq f[E(X)]$ if f is concave, it can be proven that

$$I(X;Y) \geq 0, \tag{3.26}$$

wherein the equality occurs if X and Y are statistically independent.

Note also that

$$I(X;X) = H(X) - H(X \mid X) = H(X). \tag{3.27}$$

Because of equation (3.27), entropy is sometimes referred to as *self-information*.

The *conditional mutual information* of two random variables X and Y given another variable Z is defined as

$$\begin{aligned} I(X;Y \mid Z) &= H(X \mid Z) - H(X \mid Y, Z) && (3.28)\\ &= \sum_{x,y,z} p(x,y,z) \log \frac{p(x,y \mid z)}{p(x \mid z)p(y \mid z)}, && (3.29) \end{aligned}$$

which is a generalization of equation (3.22) to conditional distributions.

3.5 Differential entropy

The entropy's classical definition applies only to discrete random variables, as it was developed by Shannon as a measure of information for computer networks [Sha49]. However, its definition may be generalized for continuous random variables, being denoted as *differential entropy* [CT91]. Being $p(x)$ the probability density function (pdf) of a given continuous random variable X, with a continuous domain $\mathcal{S}$, its differential entropy is defined as

$$h(X) = -\int_S p(x) \log p(x) dx. \tag{3.30}$$

Because probability density functions (pdf) are mathematical functions that may evaluate to values greater than one ([2]), differential entropy cannot be taken as an absolute measure of information or uncertainty because it may be negative and it is unbounded, *i.e.* $-\infty < h(X) < +\infty$. Moreover, $h(X) \to -\infty$ as the continuous random variable X becomes less uncertain, whereas discrete entropy tends to 0 in that situation.

However, *differential entropy provides a relative measure of information and uncertainty*. Fig. 3.6 depicts two different probability density functions (Gaussians) and

[2] A probability density function is a mathematical function that must verify the condition $\int_S p(x) = 1$. Nevertheless, even verifying that condition, the proposition $\exists_{x \in S},\ p(x) > 1$ may be true.

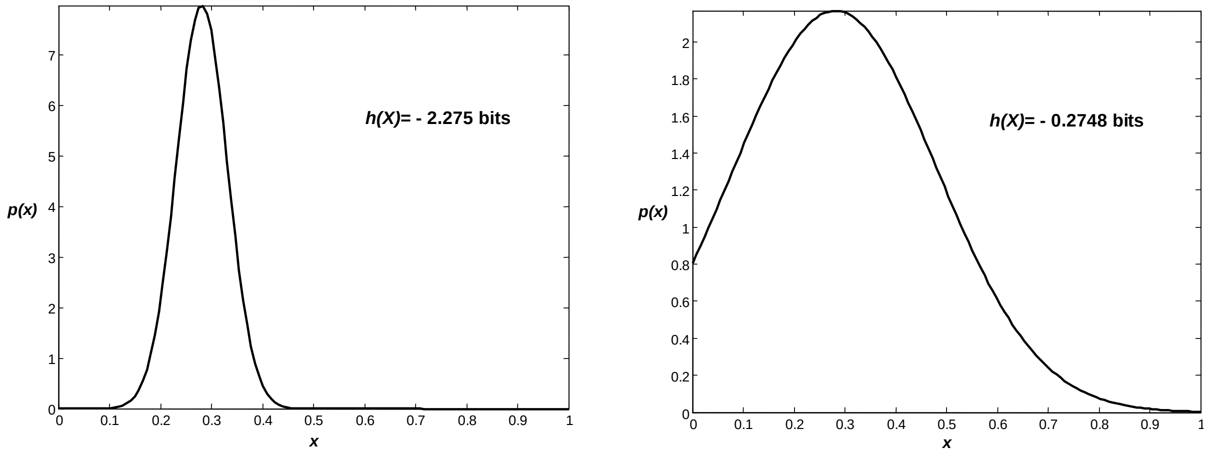


Figure 3.6: Differential entropy is a relative measure of information and uncertainty, though it cannot be taken as an absolute measure.

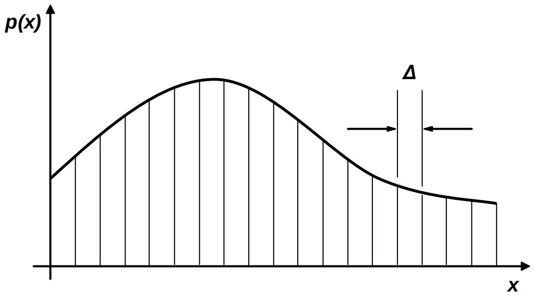


Figure 3.7: Quantization of a continuous random variable.

their differential entropy values. The pdf on the right has higher differential entropy because it presents higher uncertainty (dispersion). Since differential entropy is unbounded, $h(X) \to -\infty$ as $\sigma \to 0$ and $h(X) \to +\infty$ as $\sigma \to +\infty$.

3.5.1 Relation of differential entropy to discrete entropy

Consider a continuous random variable X with the probability density function (pdf) depicted in Fig. 3.7. Suppose that the domain of X is divided into bins of length Δ. Assuming that the pdf is continuous within the bins, and using the mean value theorem [CT91], there exists a value x_i within each bin such that

$$p(x_i)\Delta = \int_{\Delta}^{(i+1)\Delta} p(x)dx. \tag{3.31}$$

Now consider a quantized (discrete) random variable X^{Δ} defined upon the continuous random variable X, which is defined by

$$X^\Delta = x_i, \quad \text{if} \quad i\Delta \leq X \leq (i+1)\Delta. \tag{3.32}$$

Then, by equation (3.31), the probability $p_i = P(X^\Delta = x_i)$ is

$$p_i = \int_{i\Delta}^{(i+1)\Delta} p(x)dx = p(x_i)\Delta. \tag{3.33}$$

The (discrete) entropy of the quantized version is

$$\begin{aligned} H(X^\Delta) &= -\sum_{-\infty}^{+\infty} p_i \log p_i = -\sum_{-\infty}^{+\infty} p(x_i)\Delta \log(p(x_i)\Delta) \\ &= -\sum_{-\infty}^{+\infty} p(x_i)\Delta \log p(x_i) - \sum_{-\infty}^{+\infty} p(x_i)\Delta \log \Delta \end{aligned} \tag{3.34}$$

Given that

$$\sum_{-\infty}^{+\infty} p(x_i)\Delta = \sum_{-\infty}^{+\infty} p_i = 1,$$

equation (3.34) can be written as

$$H(X^\Delta) = -\sum_{-\infty}^{+\infty} p(x_i) \log p(x_i)\Delta - \log \Delta. \tag{3.35}$$

The limit of equation (3.35) when $\Delta \to 0$ is

$$\begin{aligned} \lim_{\Delta\to 0} H(X^\Delta) &= -\lim_{\Delta\to 0} \left(\sum_{-\infty}^{+\infty} p(x_i) \log p(x_i)\Delta - \log \Delta \right) \\ &= -\lim_{\Delta\to 0} \left(\sum_{-\infty}^{+\infty} p(x_i) \log p(x_i)\Delta \right) - \lim_{\Delta\to 0} \log \Delta \\ &= -\int_{-\infty}^{+\infty} p(x) \log p(x)dx - \lim_{\Delta\to 0} \log \Delta \\ &= h(X) - \lim_{\Delta\to 0} \log \Delta, \end{aligned} \tag{3.36}$$

From equation (3.36), we have

$$h(X) = \lim_{\Delta\to 0} H(X^\Delta) + \lim_{\Delta\to 0} \log \Delta, \tag{3.37}$$

which relates the differential entropy of the pdf with the discrete entropy of its quantized version. Clearly, the differential entropy is unbounded due to the second term on the right-hand of equation (3.37), because

$$\lim_{\Delta\to 0} \log \Delta = -\infty$$

and thus $h(X) \to -\infty$.

3.5.2 Differential entropy of a Gaussian

Consider a continuous random variable X following a Gaussian probability density function

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right], \tag{3.38}$$

with mean μ and standard deviation σ. Fig. 3.6 depicts examples of Gaussian probability density functions.

Using equation (3.30), the differential entropy of a Gaussian is

$$\begin{aligned} h_N(X) &= -\int_{-\infty}^{+\infty} \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right] \cdot \log \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right] dx \\ &= \log \sigma\sqrt{2\pi} \int_{-\infty}^{+\infty} \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right] dx \\ &\quad + \int_{-\infty}^{+\infty} \left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2 \cdot \frac{\log e}{\sigma\sqrt{2\pi}} \exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right] dx \\ &= \log \sigma\sqrt{2\pi} + \frac{\log e}{2\sigma^2} Var(X) = \log \sigma\sqrt{2\pi} + \frac{1}{2}\log e \\ &= \log \sqrt{2\pi e} + \log \sigma. \end{aligned} \tag{3.39}$$

Note that $Var(X)$ is the variance of X. Thus, the differential entropy of a Gaussian is proportional to $\log \sigma$ and may evaluate to negative values:

$$h_N(X) < 0 \Longleftrightarrow \sigma < \frac{1}{\sqrt{2\pi e}} = 0.242. \tag{3.40}$$

It can be proven that, from all probability density distributions with the same variance, the Gaussian distribution is the one that maximizes the differential entropy [CT91].

3.5.3 Properties

Most of the properties and theorems of the entropy definition for discrete random variables are also valid for differential entropy. However, while the latter tends to $-\infty$ when a random variable has no uncertainty/information, the former evaluates to zero.

The definitions of joint entropy, conditional entropy and mutual information can be extended to continuous random variables. The following relations are always valid for any pair of continuous random variables X and Y:

$$h(X,Y) = -\int\int p(x,y)\log p(x,y)dxdy, \tag{3.41}$$

$$h(X \mid Y) = -\int\int p(x,y)\log p(x \mid y)dxdy, \tag{3.42}$$

$$h(Y \mid X) = -\int\int p(x,y)\log p(y \mid x)dxdy, \tag{3.43}$$

$$h(X \mid Y) \leq h(X), \tag{3.44}$$

$$h(Y \mid X) \leq h(Y), \tag{3.45}$$

$$h(X,Y) = h(X) + h(Y \mid X) = h(Y) + h(X \mid Y), \tag{3.46}$$

$$h(X,Y) \leq h(X) + h(Y), \tag{3.47}$$

$$I(X;Y) = \int\int p(x,y)\log \frac{p(x,y)}{p(x)p(y)}dxdy, \tag{3.48}$$

$$I(X;Y) = h(X) - h(X \mid Y) = h(Y) - h(Y \mid X),$$

$$I(X;Y) \geq 0, \tag{3.49}$$

$$= h(X) + h(Y) - h(X,Y). \tag{3.50}$$

See [CT91] for more details about the proof of these properties.

3.6 Application examples

As it was already mentioned, entropy was developed by Shannon as an information measure in the context of computer networks [Sha49]. Thus, his entropy definition was originally used in that context to define limits on the amount of information which can be transmitted over a communication channel; limits on the compression of information which can be achieved; and how to build computer networks approaching those limits. Nevertheless, the entropy's scope of application is much wider, since its mathematical formulation of uncertainty and information associated with random variables can be used in many other domains.

Entropy can be used to develop entropy-based optimization methods, such as the *MaxEnt* and the *MinxEnt* methods [KK92]. While the former method uses the Jaynes' maximum entropy principle to minimize the distance to the uniform distribution, the latter method uses the Kullback's principle of minimum joint entropy to minimize the statistical distance to a given *a priori* distribution. In both cases, the essential idea is to use all the available information and maximize the uncertainty (entropy) of unknown information. Both methods have been used to develop optimization algorithms in different application domains, such as statistics, geography, geophysics, pattern recognition, queuing theory, parameter estimation, *etc.* The entropy concept has also been used in robotics for quantifying uncertainty and mutual information.

Stachniss *et al.* [SB03] and Bourgault *et al.* [BMW$^+$02] used it to predict the expected information outcome of different control actions while exploring an unknown environment with a robot. Tarapore *et al.* used entropy and mutual information for quantitatively correlate what sort of robot's actions lead to what sort of information (acquired sensory data), *i.e.* for understanding the agent-environment interaction [TMG04]. Sujan *et al.* developed an information-theoretical approach to fill uncertain regions of the environment map in cliff surface exploration missions with multiple robots, based on maximizing the expected new obtained information [SDH$^+$04]. Burgard *et al.* proposed a method to actively localize a robot, by choosing actions so as to minimize the expected future uncertainty, wherein entropy was used to measure the uncertainty of future belief distributions. Entropy has also been used in SLAM approaches to formulate the minimization of localization uncertainty [BMW$^+$02].

Saéz *et al.* proposed a solution to registration in the SLAM problem, using stereo-vision, which is based on the minimization of the entropy of a 2-D distribution induced by the projection of the 3-D point cloud [SE05]. The approach assumed a plane-parallel environment wherein all planes were either parallel or orthogonal. The entropy function was a measure of the quality of the alignment of map's views taken at different times and from different viewpoints. In order to maintain the map's global consistency, the different views were aligned so as to minimize the entropy function.

Viola proposed an information-theoretical approach to find the pose of an object in an image, which align the object so as to maximize the mutual information between the image and the object [Vio95].

Arbel *et al.* developed an entropy-based gaze planning algorithm to addresses the recognition of known objects in unstructured environments [AF01]. For each object being tracked, it was gathered a set of optical signatures, which were then used to compute a probability distribution for each tracked object along the set of known objects. A measure of ambiguity of the distribution based on entropy was defined to plan gaze and motion through entropy maps. Entropy maps were built off-line, during training, to relate object ambiguity to viewing position. The entropy maps of the known objects were used to navigate the sensor to the most informative gaze, *i.e.* the gaze with less entropy, in order to reduce the recognition time.

Solving the path planning problem requires the examination of different robot's configurations in its configuration space. Its complexity grows exponentially in the degrees of freedom that the robot possesses. Burns *et al.* addressed the problem by proposing an entropy-guided approach, wherein the information gain obtained associated with each examination of the configuration space was formulated as entropy reduction [BB05]. The strategy was to maximize the expected entropy reduction concerning a particular path between two given initial and final configurations, so as to achieve maximal progress toward a solution to the required path.

3.7 Summary and discussion

This chapter provided the reader with the basics of information theory, so as to contribute to the readability of the following chapters of the thesis. After an historical revision of the foundations of information theory, the basic definitions of entropy, joint entropy, conditional entropy and mutual information for discrete random variables were presented. These definitions were further extended to continuous random variables through the definition of differential entropy. The main differences between the discrete case and the continuous case were clarified, and the relationship between discrete entropy and differential entropy was presented. The chapter ended with a brief description of several application examples of those information-theoretical definitions to other application domains than computer networks, with a particular emphasis on robotics' applications.

The entropy concept plays a fundamental role in the contributions of this thesis, mainly because it is a formal measure of uncertainty. Because differential entropy is not an absolute measure of uncertainty, discrete entropy will be preferred in the approach herein proposed. In chapter 4, discrete entropy is used to measure the uncertainty of probabilistic beliefs related with space occupancy and to formally define the uncertainty of a grid-based map. In chapter 5, mutual information is used to devise an information-theoretical measure of information utility, which supports efficient sharing of sensory information within teams of robots. In chapter 6, mutual information is used to measure the sensing overlapping associated with two different exploration viewpoints, when different robots explore the same environment to build a volumetric map.

The next chapter presents a grid-based volumetric model of a map, which comprises the model itself, using entropy to quantify the map's uncertainty, converting range measurements into beliefs related with space occupancy, a probabilistic model of a range sensor, a Bayes filter for updating the map upon new sensor's measurements, and an entropy gradient-based exploration method [RDC05d, RDC05a].

Chapter 4

Probabilistic volumetric maps

An important resource for robotic mapping and exploration is obviously a map representing the robots' knowledge about the environment. As sensors have limited range, are subject to occlusions and yield noisy measurements, robots have to navigate through the environment and build the map iteratively, in order to reduce the map's uncertainty. Thus, the map's representation model should be probabilistic and represent explicitly uncertainty.

In this chapter, a *grid-based probabilistic model of a volumetric map* is proposed, which stores for each cell (voxel) a coverage belief and enables to model explicitly uncertainty [RDC05d, RDC05a]. The main contribution is a more compact representation of this belief than using histograms, and a straightforward and efficient Bayes filter to update the map upon measurements taken at different instant times and from different locations. Although the proposed framework might be used to model any phenomena spatially distributed, since it was validated through experiments with mobile robots equipped with stereo-vision range sensors providing distance measurements, hereafter a map is denoted as a coverage map, which is a 3-D representation of the environment occupancy with obstacles.

After justifying the use of grid-based maps and presenting the grid-based volumetric map and the voxel's coverage concept, an entropy-based measure of the map's uncertainty is presented. Then, a probabilistic model of a range sensor is proposed, which enables to convert range measurements into voxels' coverage estimates. A Bayes filter to easily update the map upon new data yielded by range sensors is also presented. In order to survey the environment and build the map iteratively, an entropy gradient-based exploration method is proposed, which directs the robot's sensor to frontier voxels between more explored and less explored regions. The chapter ends with examples of real volumetric maps and a brief discussion of the proposed framework for robotic mapping.

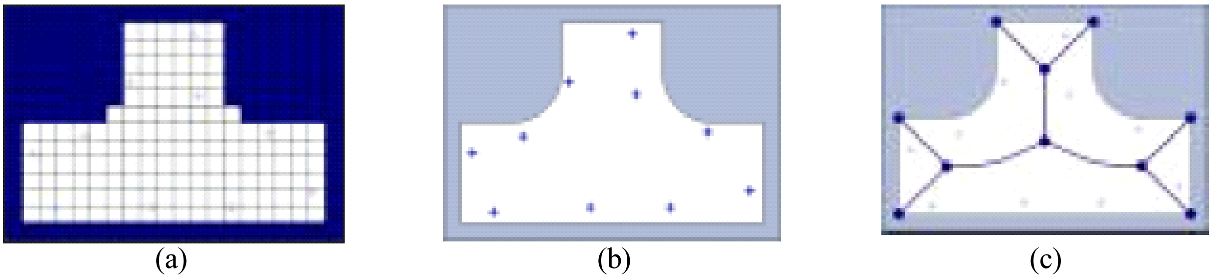

(a) (b) (c)

Figure 4.1: Basic map types: (a) geometric maps (grid-based maps); (b) feature-based maps; (c) topological maps. Figure reproduced from [CF04].

4.1 Why using grid-based maps?

There are basically three different types of representation models of a map (see Fig. 4.1): *metric*, *feature-based* and *topological* ([1]).

Metric maps are very intuitive, yield a rigorous model of the environment and help to register measurements taken from different locations. Grid-based maps — the most popular metric maps — have been widely used to represent the environments' geometry through sets of polyhedra. The metric maps' main shortcoming is not to scale well with the map's dimension.

Feature-based maps maintain a collection of distinguishable landmarks and correlate their mutual localization. Since they are based on extracting specific features in the environment, they tend to be biased to a particular environment, though they scale better with the map's dimension than metric maps, at the expense of less detail. Moreover, they are especially useful to model environments with a lot of free space and cannot be used solely to navigate through the environment and avoid obstacles.

Topological maps represent the environment as a connected graph comprising places and edges. They are mainly logical representations and store information related with connectivity. Although they scale well with the map's dimension, they suffer from the same shortcomings than feature-based maps and, moreover, they are difficult to be graphically (intuitively) represented and might yield ambiguities related with representing the same place more than once, which are difficult to overcome.

All the three aforementioned representation models present strengths and weaknesses and none of them is well suited to all situations. Choosing one of them for a specific situation means to decide about a tradeoff between detail and scalability. While metric

[1] For more details, see section 2.4.2, page 70.

maps offer much detail but usually do not scale well with large environments (*e.g.* grid-based maps), feature-based and topological maps scale better at the expense of less detail.

For large scale environments, the best solution to cope with detail and scalability simultaneously would be to combine hierarchically all of those models (*e.g.* [KB91]). For instance, in [LVH05], a computationally efficient method is presented to extract and update features via component principal analysis, so as to compress the metric data required to represent detailed 3-D maps.

As it was already pointed out in chapter 1, the main research question of the work presented herein is how to foster cooperation among intelligent robots, based on sharing useful information and proper coordination. Therefore, in the context of this thesis, building maps is the chosen application domain for those cooperative mobile robots and it is not itself the main research purpose. Moreover, as the proposed approach has been validated through indoor experiments in the laboratory, the map's dimension has been relatively confined.

Therefore, since the maps' dimension has not been very restrictive, grid-based metric maps have been used to represent the maps due to their fine detail and intuitive representation. Nevertheless, the approach can still scale up to larger environments, if the map's resolution can be decreased as a means of attenuating the dimensionality problem posed by the map's dimension increase. One of the future directions of the research presented herein will be to extend the current grid-based mapping framework with a hierarchy of representations. This hierarchy will comprise both metric and topological representations, in order to be used in outdoor large-scale environments, while being capable of representing fine detail.

4.2 Volumetric model based on a 3-D grid

One of the most popular space representation models of a map is an *occupancy grid*, which is a discretised random field wherein the probability of occupancy of each independent cell is maintained [ME85, PNDW98]. This geometric model has been extensively used in robotics mainly due to their simplicity and suitability for decision-theoretic approaches. Some recent examples of their application are [BMF$^+$00, BMW$^+$02, GMDW03, HJSL04].

The definition of probabilistic map proposed herein was first introduced in [SB03], wherein the notion of *occupancy grid* was refined in order to avoid a strictly binary representation of each cell's occupancy (free or occupied), through the notions of *coverage* and *coverage map*.

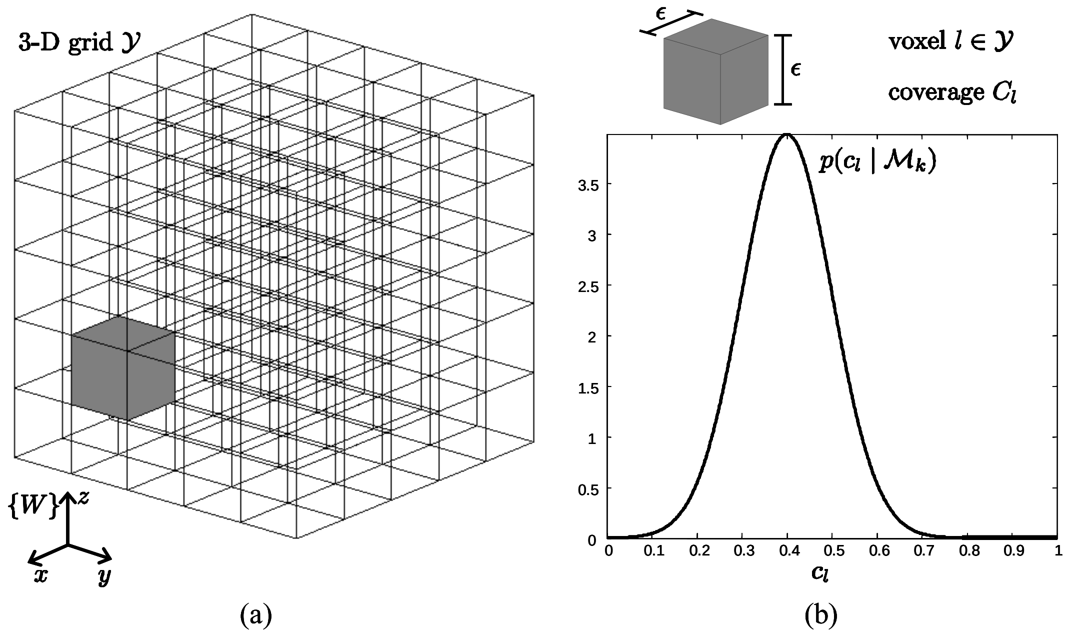


Figure 4.2: Volumetric discrete grid: (a) the grid divides the workspace into equally sized voxels (cubes), whose edges are aligned with one of the axes of the world coordinates reference frame $\{W\}$; (b) the coverage C_l of each voxel $l \in \mathcal{Y}$ with edge ϵ, given the sequence $\mathcal{M}_k$ of k batches of measurements, is modeled through a probability density function (pdf) $p(c_l \mid \mathcal{M}_k)$ (the example is a normal pdf $N(\mu_l = 0.4, \sigma_l = 0.1)$).

> **Coverage of a cell**: The coverage of a cell is the portion of the the cell that is covered or occupied by obstacles. It is represented through a continuous random variable C taking values between 0 and 1.
>
> **Coverage map**: A coverage map is a grid-based map that stores a probabilistic belief about the coverage of each cell contained in the grid.

These definitions of *coverage* and *coverage map* are used herein to define a volumetric model as a 3-D rectangular grid (lattice) comprised of a set of cells $\mathcal{Y}$, denoted as *voxels*, wherein each voxel is a cube with edge $\epsilon \in \mathbb{R}$ [RDC05d, RDC05a]. The voxels divide the workspace into equally sized cubes with volume ϵ^3. Fig. 4.2 shows a geometric representation of the representation model. Any edge of any voxel is assumed to be aligned with one of the axes of a global coordinates reference frame $\{W\}$.

The small capital letter l will be used hereafter to denote a given voxel of the grid $\mathcal{Y}$. The portion of the volume of a given voxel $l \in \mathcal{Y}$ that is covered or occupied by obstacles is modeled through a continuous random variable C_l, taking values $c_l \in [0, 1]$. This random variable is statistically described through a probability density function

$p(c_l)$. The objective of building such a map is to obtain for each voxel $l \in \mathcal{Y}$ an estimate as "accurate" ([2]) as possible about its coverage C_l.

A range sensor typically provides batches of measurements at discrete instant times. Let $\mathcal{T}$ be the discrete set of time instants $\{t_k : t_k \in \mathbb{R},\ k \in \mathbb{N}_0,\ t_{k-1} \leq t_k,\ \forall_{k\in\mathbb{N}}\}$ and let denote as $t_k \in \mathcal{T}$ the instant time when the k-th batch of measurements is obtained. Let also $t_0 \in \mathcal{T}$ denote the initial time instant before any measurements, with $t_0 \leq t_k,\ \forall_{k\in\mathbb{N}}$.

The k-th batch of measurements is

$$M_k = (\mathbf{x}_k, \mathcal{V}_k) : k \in \mathbb{N}, \tag{4.1}$$

being $\mathbf{x}_k$ the sensor's position from where measurements are obtained and $\mathcal{V}_k$ the set of measurements belonging to the batch, provided by the robot's sensor at $t = t_k,\ t_k \in \mathcal{T}$.

The sequence of k batches of measurements, corresponding to the period of time $t_0 \leq t \leq t_k$ is

$$\mathcal{M}_k = \{M_i : i \in \mathbb{N},\ i \leq k\}. \tag{4.2}$$

Before any batch of measurements, i.e. for $k = 0$, the sequence of batches is the empty set $\mathcal{M}_0 = \emptyset$.

The knowledge about a given voxel's coverage C_l, after k batches of measurements, is modeled through the pdf

$$p(c_l \mid \mathcal{M}_k),\ 0 \leq c_l \leq 1. \tag{4.3}$$

The *volumetric probabilistic map*, after k batches of measurements, is the set of random variables

$$\mathcal{C} = \{C_l : l \in \mathcal{Y}\}, \tag{4.4}$$

containing a coverage random variable for each voxel l contained in the discrete grid $\mathcal{Y}$. These random variables are statistically described through the set of coverage probability density functions:

$$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k) = \{p(c_l \mid \mathcal{M}_k) :\ l \in \mathcal{Y}\}. \tag{4.5}$$

For the sake of simplicity, the coverage of each individual voxel is assumed to be independent from the other voxels' coverage and, thus, $\mathcal{C}$ is a set of statistically independent random variables. Recall that if two continuous random variables X and Y, with probability density functions $p(x)$ and $p(y)$, respectively, are statistically independent, their joint probability density function (pdf) is just $p(x, y) = p(x)p(y)$. Therefore, given the aforementioned assumption, the *map's joint pdf* $p(\mathcal{C} \mid \mathcal{M}_k)$, after k batches of measurements,

[2] A formal measure of how *accurate* is the voxel's coverage belief will be given in this chapter later on.

can be simply written as

$$p(\mathcal{C} \mid \mathcal{M}_k) = \prod_{l \in \mathcal{Y}} p(c_l \mid \mathcal{M}_k). \tag{4.6}$$

4.3 Entropy-based measure of the map's quality

The objective of building a probabilistic grid-based volumetric map $\mathcal{C}$ is to obtain for each voxel of the grid $l \in \mathcal{Y}$ an estimate as "accurate" as possible about its coverage $C_l \in \mathcal{C}$. Chapter 3 presented entropy as formal mathematical measure for the uncertainty of a probabilistic belief, which is used in this section to quantify the voxel's coverage uncertainty and the map's uncertainty. Hereafter, these quantities will be referred to as the voxel's entropy and the map's entropy, respectively.

Although the Shannon's original definition of entropy for discrete random variables can be extended to continuous random variables through the definition of differential entropy, this cannot be taken as an absolute measure of uncertainty ([3]). Furthermore, discrete entropy is more convenient for modeling mutual information between sets of random variables, because it is always non-negative. This will be especially relevant in chapter 6, wherein mutual information is used to measure the sensing overlapping associated with two different exploration viewpoints, when different robots explore the same environment to build a volumetric map. For these reasons, computing discrete entropy is generally preferable to differential entropy.

Since the knowledge about the voxel's coverage is modeled through a continuous random variable, a quantized version of the voxel's coverage pdf is used to compute discrete entropy (Fig. 4.3). The coverage continuous random variable C_l, $l \in \mathcal{Y}$ is discretised through a discrete random variable $C_l^{\triangle}$ having b possible outcomes $c_l^{\triangle} \in \{1, \ldots, b\}$. This discrete variable models an approximation of the voxels' coverage pdf $p(c_l)$ through a relative frequency histogram $p(c_l^{\triangle})$ with b bins, such as:

$$p(c_l^{\triangle} = i) = \int_{\frac{i-1}{b}}^{\frac{i}{b}} p(c_l) dc_l, \; i \in \{1, \ldots, b\}. \tag{4.7}$$

Using the definition of discrete entropy, given by equation (3.2), the *voxel's entropy* is

$$H(C_l) \equiv \sum_{i=1}^{b} p(c_l^{\triangle} = i) \log p(c_l^{\triangle} = i). \tag{4.8}$$

[3] For more details, see section 3.5, page 91.

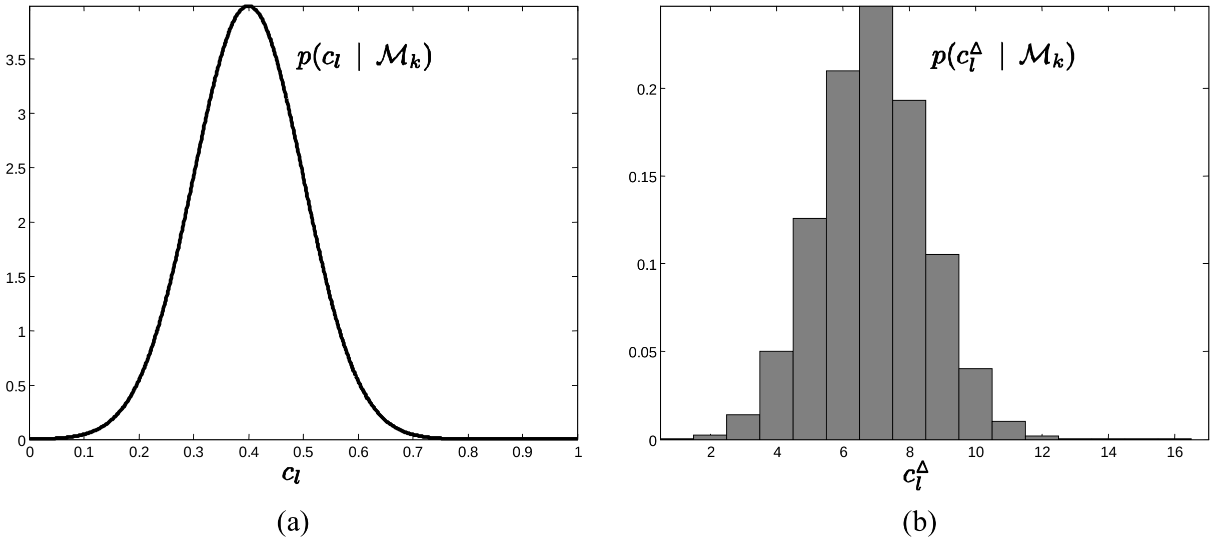


Figure 4.3: Quantization of the voxel's coverage probability density function (pdf): (a) example of a Gaussian for the voxel's pdf $p(c_l \mid \mathcal{M}_k)$; (b) quantized version $p(c_l^\Delta \mid \mathcal{M}_k)$ of the voxel's pdf $p(c_l \mid \mathcal{M}_k)$, using a 16 bins histogram. In this example, the voxel's discrete entropy is $H(C_l) = 2.749$ bits.

Hereafter, when nothing is said, always assume that $b = 128$ bins are used in the computation of equation (4.8), which means that the voxel's entropy is bounded to the interval $0 \leq H(C_l) < 7$ bits.

As the coverage random variables of different voxels are assumed to be statistically independent, accordingly with equations (3.18) and (3.19) ([4]), page 89, and the map's joint probability density function given by equation (4.6), the *map's joint entropy* is

$$H(\mathcal{C}) \equiv \sum_{l \in \mathcal{Y}} H(C_l), \tag{4.9}$$

i.e. it is just the sum of the voxels' individual entropy. Equation (4.9) is a formal measure of how much uncertainty the map contains. As the map's uncertainty is directly related with the map's quality, equation (4.9) is also a measure of the map's quality: the map's quality increases as the map's entropy decreases [RDC05d, RDC05a].

If the knowledge about voxels' coverage is conditioned to the k previous batches of measurements $\mathcal{M}_k$, equations (4.7), (4.8) and (4.9) can obviously also be used to compute the voxel's coverage entropy $H(C_l \mid \mathcal{M}_k)$ and the map's joint entropy $H(\mathcal{C} \mid \mathcal{M}_k)$ conditioned to that knowledge, by using $p(c_l \mid \mathcal{M}_k)$ and $p(c_l^\Delta \mid \mathcal{M}_k)$ instead of using $p(c_l)$

[4] Although equations (3.18) and (3.19) were written for sets of discrete random variables, it can be easily proven that they are also valid for continuous random variables.

and $p(c_l^{\triangle})$. In order to simplify the notation, the map's joint entropy $H(\mathcal{C} \mid \mathcal{M}_k)$ after k batches of measurements will be denoted hereafter as $H(t_k)$.

4.3.1 Mission execution time

Since discrete entropy is an absolute measure of uncertainty, the map's entropy given by equation (4.9) inherits that property and is an absolute measure of the map's uncertainty or quality. This property can be used to define an important performance measure, which is the mission execution time.

Consider a given environment to be mapped and its associated discrete grid $\mathcal{Y}$. If different mapping missions are performed in this environment at different time periods and, perhaps, by different teams of robots, the robots' performance can be easily compared if a given map's entropy threshold H_{th} is defined. This entropy value is the minimum map's quality that robots must accomplish at the end of the mapping mission.

The *mission execution time* $t_{k_{max}} \in \mathcal{T}$, which is associated with the k_{max}-th batch of measurements, *i.e.* the last batch of measurements acquired by the robot with the lowest entropy at the end of the mission, can be defined as the time instant that verifies the proposition

$$H(t_{kmax}) \leq H_{th} \wedge \forall_{k<k_{max},\ k\in\mathbb{N}_0},\ H(t_k) > H_{th}. \tag{4.10}$$

The mission execution time is thus the first instant time when the map's entropy is reduced below the pre-defined map's entropy threshold H_{th} [RDC05d, RDC05b, RDC05a]. It can be used as a performance benchmark to compare the performance of different mapping missions in the same environment.

4.4 Converting range measurements into coverage estimates

A range sensor typically provides batches of range measurements from each point where it is located. Consider a batch of measurements $M_k = (\mathbf{x}_k, \mathcal{V}_k)$, being $\mathbf{x}_k \in \mathbb{R}^3$ the sensor's position from where measurements are obtained (shared by all measurements in the batch), and a set

$$\mathcal{V}_k = \{\overrightarrow{\mathbf{v}}_{k,i} \in \mathbb{R}^3 : i \in \mathbb{N},\ i \leq m_k\} \tag{4.11}$$

of m_k applied vectors (measurements) connecting $\mathbf{x}_k$ to the set of points $\{\mathbf{x}_k + \overrightarrow{\mathbf{v}}_{k,i} : i \in \mathbb{N},\ i \leq m_k\}$ where obstacles are detected. Each measurement $\overrightarrow{\mathbf{v}}_{k,i} \in \mathcal{V}_k$, obtained from the sensor's location $\mathbf{x}_k$, only influences the coverage belief of a subset of voxels $\mathcal{Z}_{k,i}$ belonging to the volumetric discrete grid $\mathcal{Y}$ (see Fig. 4.4-b). Thus, a method to convert range measurements into voxels' coverage estimates is needed [RDC05d, RDC05a].

4.4.1 Voxels traversed by a vector

Each range measurement is an applied vector in the current robot's sensor position, which influences the coverage belief of those voxels that are traversed by the vector. An useful operator to determine this set of voxels is the one that computes the set of voxels traversed by a vector [RDC05d, RDC05a].

Consider an applied vector $\overrightarrow{\mathbf{u}} \in \mathbb{R}^3$ connecting point $\mathbf{a}$ to point $\mathbf{b}$ (see Fig. 4.4-a). The set of voxels traversed by $\overrightarrow{\mathbf{u}}$ can be determined by sampling it so that at least one sample per traversed voxel is gathered in a set of w 3-D points

$$\mathcal{Q}(\overrightarrow{\mathbf{u}}, \mathbf{a}) = \{\mathbf{q}_i : i \in \mathbb{N},\ i \leq w\}. \tag{4.12}$$

To guarantee this minimum sampling, vector $\overrightarrow{\mathbf{u}}$ is divided into segments with maximum length equal to the voxel's edge ϵ, wherein the coordinates of each sampling point are given by

$$\mathbf{q}_i = \mathbf{a} + (i-1) \cdot \epsilon \cdot \frac{\overrightarrow{\mathbf{u}}}{\|\overrightarrow{\mathbf{u}}\|},\ \ i \in \mathbb{N},\ i \leq w. \tag{4.13}$$

The number of sampled points is

$$w = \operatorname{trunc}\left(\frac{\|\overrightarrow{\mathbf{u}}\|}{\epsilon}\right) + 1. \tag{4.14}$$

Let

$$v : \mathbb{R}^3 \to \mathcal{Y} \tag{4.15}$$

be a function which determines what grid's voxel a given point belongs to. The set of voxels traversed by vector $\overrightarrow{\mathbf{u}}$ when applied in point $\mathbf{a}$ is

$$\mathcal{Z}(\overrightarrow{\mathbf{u}}, \mathbf{a}) = \{v(\mathbf{q}_i) : \mathbf{q}_i \in \mathcal{Q}(\overrightarrow{\mathbf{u}}, \mathbf{a})\} \subset \mathcal{Y}. \tag{4.16}$$

4.4.2 Voxels influenced by a measurement

Consider again a batch of measurements $M_k = (\mathbf{x}_k, \mathcal{V}_k)$and a given measurement $\overrightarrow{\mathbf{v}}_{k,i} \in \mathcal{V}_k$, obtained from the sensor's location $\mathbf{x}_k$. Fig. 4.4-b suggests that the set of voxels

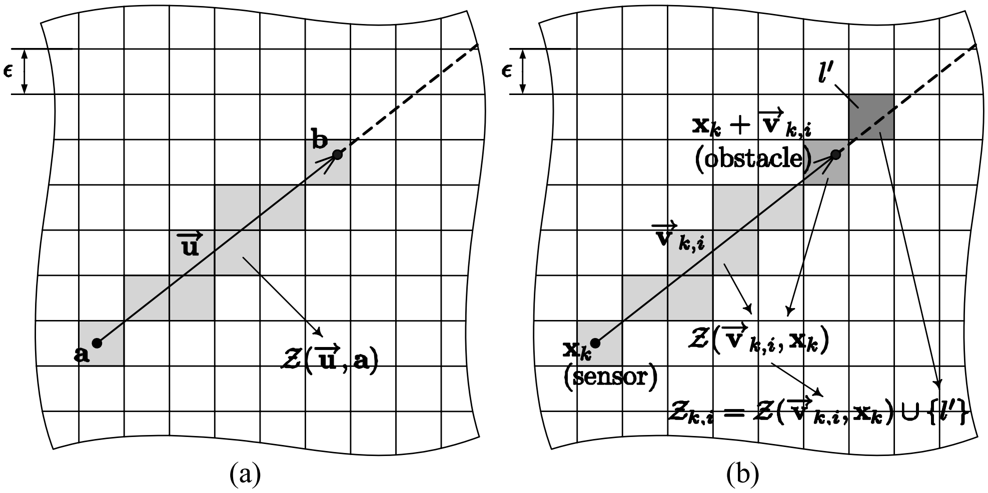

ϵ
b
$\vec{\mathbf{u}}$
a
$\mathcal{Z}(\vec{\mathbf{u}}, \mathbf{a})$
(a)
ϵ
l'
$\mathbf{x}_k + \vec{\mathbf{v}}_{k,i}$
(obstacle)
$\vec{\mathbf{v}}_{k,i}$
$\mathbf{x}_k$
(sensor)
$\mathcal{Z}(\vec{\mathbf{v}}_{k,i}, \mathbf{x}_k)$
$\mathcal{Z}_{k,i} = \mathcal{Z}(\vec{\mathbf{v}}_{k,i}, \mathbf{x}_k) \cup \{l'\}$
(b)

Figure 4.4: Set of voxels traversed by a vector in a 2-D grid: (a) the set of traversed voxels $\mathcal{Z}(\vec{\mathbf{u}}, \mathbf{a})$ contains the shaded voxels traversed by the vector $\vec{\mathbf{u}}$ when applied in point $\mathbf{a}$; (b) given a measurement (vector) $\vec{\mathbf{v}}_{k,i}$, when the sensor is located in the point $\mathbf{x}_k$, the set of influenced voxels $\mathcal{Z}_{k,i}$ contains the shaded voxels; light grey voxels between the sensor and the obstacle are more likely to be fully empty, dark grey voxels near to the detected obstacle have coverage values between 0 and 1, and the black voxel l', located immediately behind the detected obstacle, is more likely to be fully occupied.

$\mathcal{Z}_{k,i} \subset \mathcal{Y}$, whose coverage belief is influenced by the range measurement, includes the set of voxels $\mathcal{Z}(\overrightarrow{\mathbf{v}}_{k,i}, \mathbf{x}_k)$ traversed by $\overrightarrow{\mathbf{v}}_{k,i}$, plus the voxel l' which is immediately behind the obstacle [RDC05d, RDC05a]. This can be written as

$$\mathcal{Z}_{k,i} = \mathcal{Z}(\overrightarrow{\mathbf{v}}_{k,i}, \mathbf{x}_k) \cup \{l'\}. \tag{4.17}$$

The voxels belonging to the set of voxels $\mathcal{Z}(\overrightarrow{\mathbf{v}}_{k,i}, \mathbf{x}_k)$ traversed by the vector $\overrightarrow{\mathbf{v}}_{k,i}$, applied in $\mathbf{x}_k$, and located between the sensor and the obstacle, are more likely to be fully empty ([5]). The voxel l', located immediately behind the obstacle, can be computed as

$$l' = v\left(\mathbf{x}_k + (\text{trunc}(\|\overrightarrow{\mathbf{v}}_{k,i}\|/\epsilon) + 1) \cdot \epsilon \cdot \frac{\overrightarrow{\mathbf{v}}_{k,i}}{\|\overrightarrow{\mathbf{v}}_{k,i}\|}\right). \tag{4.18}$$

and is more likely to be fully occupied.

4.4.3 Measurements influencing the voxel's coverage

As it was mentioned previously, the coverage belief of each voxel *is not* influenced by *every* range measurements yielded by the range sensors. In order to maintain the probabilistic

[5] This statement is just to give a first insight into the problem and is stated more rigorously in this chapter later on.

belief about the coverage of each voxel $l \in \mathcal{Y}$, the set of measurements that *really* influence the voxel's coverage belief need to be maintained.

Consider a sensor's position $\mathbf{x}_k$ and a sensor's measurement $\overrightarrow{\mathbf{v}}_{k,i}$ obtained from that position. If a given voxel $l \in \mathcal{Y}$ belongs to $\mathcal{Z}_{k,i}$, this means that the range measurement influences its coverage belief.

Let

$$\mathbf{w} : \ \mathcal{Y} \rightarrow \mathbb{R}^3 \tag{4.19}$$

be a function that computes the center coordinates $[x_l, y_l, z_l]^T \in \mathbb{R}^3$ of a voxel $l \in \mathcal{Y}$. Let the tuple

$$D_j^l = (d_j, d_j^l) \tag{4.20}$$

be an individual measurement influencing the coverage belief of a voxel $l \in \mathcal{Y}$, being

$$d_j = \|\overrightarrow{\mathbf{v}}_{k,i}\| \tag{4.21}$$

the measured distance (distance between the sensor and the detected obstacle) and

$$d_j^l = \|(\mathbf{w}(l) - \mathbf{x}_k)\| \tag{4.22}$$

the distance between the sensor and the voxel's center.

The set of $n_k(l)$ measurements influencing the coverage belief of a voxel $l \in \mathcal{Y}$, after k batches of measurements, is

$$\mathcal{D}_k^l = \{D_j^l : j \in \mathbb{N},\ j \leq n_k(l)\} = \{D_1^l, \ldots, D_{n_k(l)}^l\}, \tag{4.23}$$

having cardinality

$$n_k(l) < \sum_{a=1}^{k} m_a,\ n_k(l) \in \mathbb{N}_0, \tag{4.24}$$

because not all measurements yielded by the sensor necessarily influence the voxel's coverage.

Given the initial empty set of influencing measurements $\mathcal{D}_0^l$, the set of influencing measurements is recursively built upon $\mathcal{M}_k$ as

$$\mathcal{D}_k^l = \mathcal{D}_{k-1}^l \cup \left[\bigcup_{i=1}^{m_k} \left\{ \begin{array}{ll} \{\|\overrightarrow{\mathbf{v}}_{k,i}\|, \|(\mathbf{w}(l) - \mathbf{x}_k)\|)\}, & l \in \mathcal{Z}_{k,i}, \\ \emptyset, & \text{otherwise.} \end{array} \right\} \right]. \tag{4.25}$$

Note that, accordingly with this equation, if $l \notin \mathcal{Z}_{k,i}, \forall_{i \in \{1,\ldots,m_k\}}$ we have $\mathcal{D}_k^l = \mathcal{D}_{k-1}^l$, *i.e.* in that case there is no any measurement influencing the coverage belief of voxel l in the k-th batch of measurements.

The set $\mathcal{M}_k$ given by equation (4.2) contains all the measurements yielded by the sensor until the k-th batch of measurements, but the measurements which really influence the coverage of voxel $l \in \mathcal{Y}$ are those measurements contained in the set $\mathcal{D}_k^l$. For this reason, we have the important equality

$$p(c_l \mid \mathcal{M}_k) = p(c_l \mid \mathcal{D}_k^l), \ \forall_{l \in \mathcal{Y},\ k \in \mathbb{N}_0}. \tag{4.26}$$

4.4.4 Probabilistic model of a range sensor

Consider a given voxel $l \in \mathcal{Y}$ and a measurement $D_j^l = (d_j, d_j^l)$ influencing its coverage belief. In order to convert the range measurement into coverage estimates $C_l = c_l$, a probabilistic model of the range sensor is needed, which is modeled through the probability density function (pdf) $p(c_l \mid D_j^l)$.

The exact model of the distribution $p(c_l \mid D_j^l)$ is generally unknown. However, as localization errors and sensor errors can be usually assumed to follow the Gaussian model given by equation (3.38), page 94, the voxel's coverage belief can be modeled through a Gaussian model

$$p(c_l \mid D_j^l) = N(\mu(d_j, d_j^l), \sigma(d_j, d_j^l), c_l), \tag{4.27}$$

wherein, accordingly with the previously defined notation, $d_j \in \mathbb{R}$ is the distance between the sensor and the detected obstacle and $d_j^l \in \mathbb{R}$ the distance between the sensor and the voxel's center [RDC05d, RDC05a].

Being ϵ the voxel's edge, the mean of the Gaussian sensor model is defined as

$$\mu(d_j, d_j^l) = \begin{cases} 0, & (d_j^l - d_j) \leq -\frac{\epsilon}{2} \\ \frac{1}{2} + \frac{d_j^l - d_j}{\epsilon}, & |d_j^l - d_j| < \frac{\epsilon}{2} \\ 1, & (d_j^l - d_j) \geq \frac{\epsilon}{2} \end{cases} . \tag{4.28}$$

This equation distinguishes three cases, which are depicted in Fig. 4.5. In the first case (Fig. 4.5-a), the measured distance does not end in the voxel l, with $d_j^l < d_j$, and thus it is more likely that the voxel is fully empty (coverage equal to 0). In the second case (Fig. 4.5-b), the measured distance ends in l and the mean of its coverage is inverse proportional to the amount of the voxel covered by d_j (a value between 0 and 1). In the third case (Fig. 4.5-c), which is only applicable to the voxel l' in equation (4.17), located immediately behind the obstacle, the measured distance does not end in the voxel l, with $d_j^l > d_j$, and thus it is more likely that the voxel is fully occupied (coverage equal to 1).

The range sensors' accuracy typically decreases with distance d. A reasonable model for the standard deviation $\sigma_s(d)$ of the range sensor's error as a function of distance is

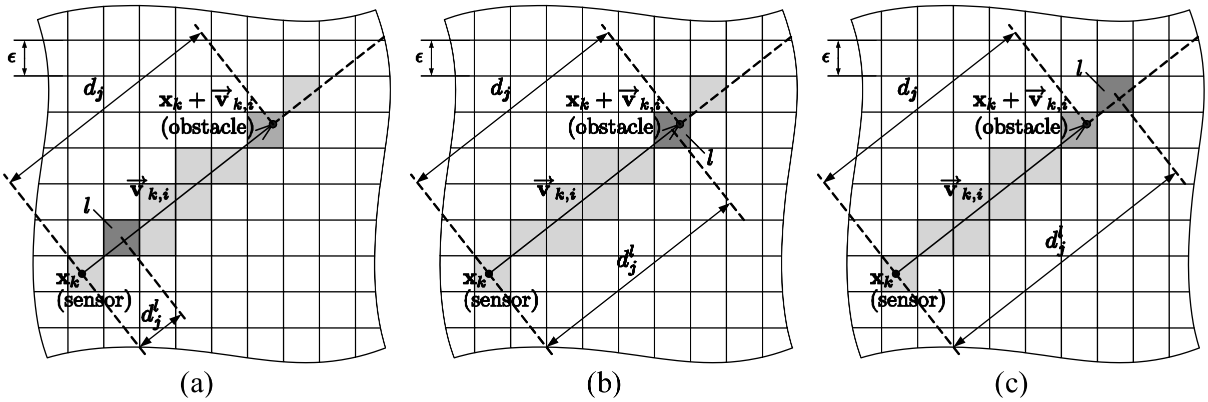


Figure 4.5: Converting a range measurement into a coverage estimate of a given voxel $l \in \mathcal{Y}$ (darkest shaded voxel): (a) the voxel is likely to be fully empty ($c_l = 0$) if it is located between the sensor and the obstacle; (b) the voxel's coverage takes values $0 \leq c_l \leq 1$ if the voxel contains the point where the obstacle is detected, depending on the difference $-\frac{\epsilon}{2} \leq (d_j^l - d_j) \leq \frac{\epsilon}{2}$; (c) the voxel is likely to be fully occupied ($c_l = 1$) if it is located immediately behind the detected obstacle.

the linear model

$$\sigma_s(d) = \sigma_{min} + \zeta.d. \tag{4.29}$$

This model states that $\sigma_s(d)$ is minimum and equal to σ_{min} near to the sensor and increases linearly with distance d with a derivative ζ.

Concerning the standard deviation of the Gaussian model, Fig. 4.5 suggests that the voxel's coverage estimate provided by a given range measurement has higher uncertainty if the measured distance does end in the voxel (see Fig. 4.5-b), *i.e.* if $|d_j^l - d_j| \leq \frac{\epsilon}{2}$. On the other hand, the uncertainty decays with $|d_j^l - d_j|$ for voxels farther from the detected voxel (see Fig. 4.5-a,c). A reasonable model for this modulation of the uncertainty, associated with the voxel's coverage estimate provided by a given influencing measurement D_j^l, is depicted in Fig. 4.6, wherein the Gaussian's standard deviation $\sigma(d_j, d_j^l)$ decays exponentially with $|d_j^l - d_j|$, given the damping ratio τ.

Thus, the standard deviation of the Gaussian sensor model is modeled as

$$\sigma(d_j, d_j^l) = \begin{cases} \frac{\sigma_s(d_j)}{\epsilon}, & |d_j^l - d_j| \leq \frac{\epsilon}{2} \\ \\ \frac{\sigma_s(d_j)}{\epsilon} \exp\left(-\frac{|d_j^l - d_j| - \frac{\epsilon}{2}}{\tau}\right), & \text{otherwise} \end{cases}. \tag{4.30}$$

This equation states that $\sigma(d_j, d_j^l)$ is equal to $\sigma_s(d_j)/\epsilon$ if the measured distance does end in the voxel (Fig. 4.5-b) and, given the damping ratio τ, decays exponentially with $|d_j^l - d_j|$

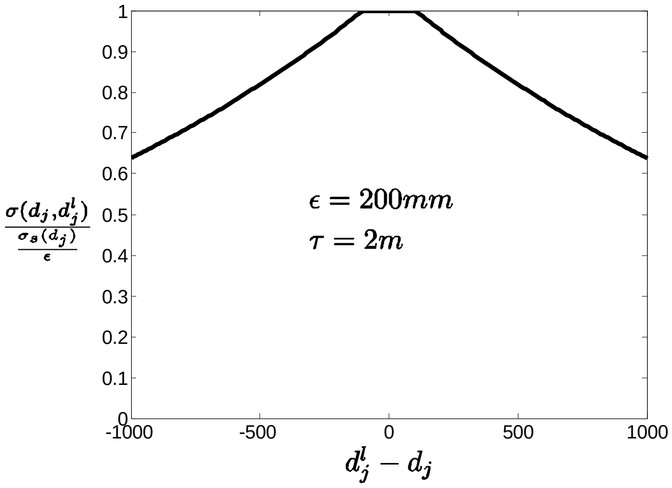


Figure 4.6: Example of the coverage's standard deviation damping with the distance $|d_j^l - d_j|$ between the voxel's center and the detected obstacle.

for voxels farther from the detected obstacle (Fig. 4.5-a,c) which, intuitively, have less uncertain coverage estimates.

Accordingly with the definition of the coverage's probability density function (pdf) given by equation (4.3), page 4.3, the sensor model's Gaussian must be truncated so that the cumulative probability over the coverage domain sums up to one, *i.e.* in order to verify the condition $P(0 \leq C_l \leq 1) = 1$.

Stachniss *et al.* proposed a sensor model for a range sensor, which is based on a mixture of a Gaussian and an uniform distribution [SB03]. This latter distribution adds some white noise to ensure a correct normalization when truncating the Gaussian to the range $[0, 1]$. Since adding white noise is a very questionable normalization method, a new method of normalizing the Gaussian distribution, truncated to the interval $0 \leq C_l \leq 1$, is proposed herein: the idea is to multiply the pdf by a normalization factor

$$\gamma(\mu, \sigma) = \left(\int_0^1 N(\mu, \sigma, x).dx \right)^{-1}, \tag{4.31}$$

which preserves the normal distribution instead of summing white noise [RDC05d, RDC05a].

As it will be seen in the following section, preserving the normal distribution makes the coverage update upon new measurements quite simple. The definition of the Gaussian's mean given by equation (4.28) is also slightly different from the Gaussian definition proposed by Stachniss *et al.* for the second case of the equation [SB03], due to their different normalization method.

Fig. 4.7 shows an example of the sensor model for a detected obstacle at a distance

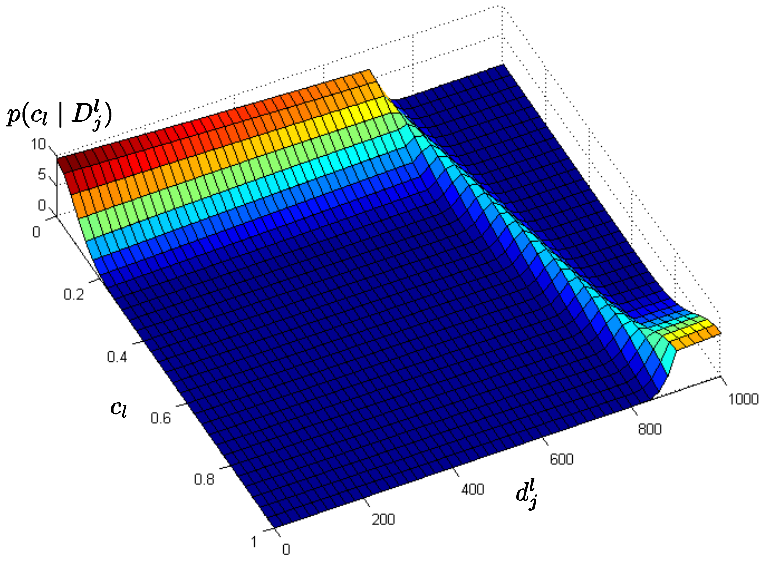


Figure 4.7: Example of a sensor model for the range sensor: $d_j = 800\ mm$, $\sigma_{min} = 16\ mm$, $\zeta = 1 \times 10^{-2}$, $\tau = 2\ m$, $\epsilon = 200\ mm$.

$d_j = 800\ mm$ and $d_j^l \in [0, 1000]\ mm$. In order to use the proposed sensor model with a given range sensor, an appropriate calibration procedure has to be performed in order to estimate its parameters σ_{min}, ζ and τ. This topic is addressed in section 4.7.3, page 127, after presenting the mobile robots that were used to validate the framework proposed herein (section 4.7).

4.5 Updating the map upon new measurements

Consider the current robot's map $p(\mathcal{C} \mid \mathcal{M}_{k-1})$ conditioned to the past $k-1$ batches of measurements, *i.e.* the measurements obtained for $t_0 \leq t \leq t_{k-1}$. Consider also a new batch of measurements $M_k = (\mathbf{x}_k, \mathcal{V}_k)$, being $\mathbf{x}_k$ the sensor's position and $\mathcal{V}_k = \{\overrightarrow{\mathbf{v}}_{k,i} \in \mathbb{R}^3 : i \in \mathbb{N},\ i \leq m_k\}$ a set of m_k measurements. The tuple M_k is the k-th batch of measurements yielded by the range sensor at $t = t_k$. This section presents a straightforward and efficient Bayes filter to update the map $p(\mathcal{C} \mid \mathcal{M}_{k-1})$ upon a new batch of measurements M_k, so as to obtain a new probabilistic map $p(\mathcal{C} \mid \mathcal{M}_k)$, with $\mathcal{M}_k = \mathcal{M}_{k-1} \cup \{M_k\}$ [RDC05d, RDC05a].

Recall that: $\mathcal{Z}_{k,i} \subset \mathcal{Y}$ denotes the set of influenced voxels by a measurement $\overrightarrow{\mathbf{v}}_{k,i} \in \mathcal{V}_k$; $\mathcal{D}_k^l$ denotes the set of influencing measurements of a voxel $l \in \mathcal{Y}$ after the k-th batch of measurements; $\mathcal{D}_k^l$ has cardinality $n_k(l)$; and $\mathcal{D}_n^l$ denotes a set of n measurements

$\{D^l_1, \ldots, D^l_n\}$ influencing the coverage belief of a voxel $l \in \mathcal{Y}$. Note that $\mathcal{D}^l_{n_k(l)}$ and $\mathcal{D}^l_k$ are equivalent notations for the sequence of measurements influencing the coverage belief of a voxel $l \in \mathcal{Y}$ up to the k-th batch of measurements.

The algorithm for updating the map upon the new batch M_k can be written in pseudocode as:

```
$\mathcal{D}^l_k \leftarrow \mathcal{D}^l_{k-1},\ \ \forall_{l \in \mathcal{Y}}$
$n_k(l) \leftarrow n_{k-1}(l),\ \ \forall_{l \in \mathcal{Y}}$
for $i = 1 \ldots m_k$
  forall $l \in \mathcal{Z}_{k,i}$
    $n_k(l) \leftarrow n_k(l) + 1$
    Compute influencing measurement $D^l_{n_k(l)}$ upon $\overrightarrow{\mathbf{v}}_{k,i}$
    $\mathcal{D}^l_k \leftarrow \mathcal{D}^l_k \cup \{D^l_{n_k(l)}\}$
    Update $p(c_l \mid \mathcal{D}^l_{n_k(l)-1})$ upon $p(c_l \mid D^l_{n_k(l)})$
       and obtain $p(c_l \mid \mathcal{D}^l_{n_k(l)})$
  end_forall
end_for.
```

Equations (4.21) and (4.22), page 109, are used to compute $D^l_{n_k(l)} = (d_{n_k(l)}, d^l_{n_k(l)})$ upon a vector $\overrightarrow{\mathbf{v}}_{k,i}$ yielded by the sensor located in $\mathbf{x}_k$. The sensor model, given by equation (4.27), page 110, is used to convert sensor's measurements into coverage estimates. It has not been stated yet how to specify the initial voxel's coverage belief $p(c_l \mid \mathcal{D}^l_0) = p(c_l \mid \mathcal{M}_0)$ and how to update the map upon new coverage estimates, *i.e.* how to combine the current voxel's coverage belief $p(c_l \mid \mathcal{D}^l_{n-1})$ with a new estimate $p(c_l \mid D^l_n)$, in order to obtain an updated coverage belief $p(c_l \mid \mathcal{D}^l_n)$, being $\mathcal{D}^l_n = \mathcal{D}^l_{n-1} \cup \{D^l_n\}$.

4.5.1 Initial map

The initial voxel's coverage belief $p(c_l \mid \mathcal{D}^l_0) = p(c_l \mid \mathcal{M}_0)$ represents prior knowledge, before any measurements. Unless there is a previous map of the environment being mapped, that prior belief is usually chosen to be the less informative, *i.e.* a probability density function (pdf) having maximum uncertainty (entropy).

In section 3.5.2, page 94, the equation (3.39) was derived to compute the differential entropy of a Gaussian pdf. Therein, it was proven that the differential entropy of a Gaussian is proportional to the logarithm of its standard deviation σ. Moreover, the

Gaussian pdf is the maximum entropy pdf given the first two moments [CT91] and, thus, equation (3.39) is a maximum differential entropy bound for a pdf with variance σ^2. For these reasons, a convenient initial belief $p(c_l \mid \mathcal{D}_0^l)$ is a Gaussian distribution with $\sigma \to +\infty$, *i.e.* an uniform distribution. In practice, this means choosing a Gaussian with σ much larger (*e.g.* ten times greater) than the sensor's model standard deviation, given by equation (4.30) [RDC05d, RDC05a].

4.5.2 Bayes filter for updating the voxel's coverage belief

Consider a given voxel $l \in \mathcal{Y}$ and the set

$$\mathcal{D}_n^l = \{D_1^l, \ldots, D_n^l\} \tag{4.32}$$

containing n measurements influencing its coverage. Given the list of influencing measurements $\mathcal{D}_{n-1}^l$ before the last measurement D_n^l, a Bayes filter is here proposed to update the coverage probabilistic belief $p(c_l \mid \mathcal{D}_{n-1}^l)$ and to determine the new voxel's coverage belief $p(c_l \mid \mathcal{D}_n^l)$ upon its new influencing measurement D_n^l [RDC05d, RDC05a]. This measurement is converted into a new coverage estimate through the sensor model $p(c_l \mid D_n^l)$, given by equation (4.27).

Being X and Y a pair of random variables with marginal probability density functions $p(x)$ and $p(y)$, respectively, the Bayes' law [Pap91] states that

$$p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)} = \eta p(y \mid x)p(x), \tag{4.33}$$

wherein $p(x)$ is called the prior and η is a normalizer that is necessary to ensure that the integral of the left-hand side is indeed a valid probability density function (pdf). Bayes' law is especially useful in statistical inference because it inverts conditional probabilities. If the pdf of X need to be estimated upon measurements of Y, the Bayes' law allows to estimate $p(x \mid y)$ upon the prior knowledge $p(x)$, when $p(y \mid x)$ is known. The denominator $p(y)$ is not very important in practical situations; it is just a scale factor to properly normalize the function $p(x \mid y)$, so that its integral sums up to one over the sample space of X.

The belief about the voxel's coverage can be computed as

$$p(c_l \mid \mathcal{D}_n^l) = \frac{p(\mathcal{D}_n^l \mid c_l).p(c_l)}{p(\mathcal{D}_n^l)} = \beta_1.p(c_l).p(\mathcal{D}_n^l \mid c_l) \tag{4.34}$$

$$= \beta_1.p(c_l).\prod_{j=1}^{n} p(D_j^l \mid c_l) \tag{4.35}$$

$$= \beta_1.p(c_l).\prod_{j=1}^{n} \frac{p(c_l \mid D_j^l).p(D_j^l)}{p(c_l)} \tag{4.36}$$

$$= \beta_1.\beta_2.\prod_{j=1}^{n} p(c_l \mid D_j^l) = \beta_1.\beta_2.p(c_l \mid D_n^l).p(c_l \mid \mathcal{D}_{n-1}^l). \tag{4.37}$$

Applying the Bayes' law, equation (4.34) is obtained. Then, assuming that consecutive measurements are independent given the voxel's coverage, we have equation (4.35). Applying again the Bayes' law, we have equation (4.36). Assuming that $p(D_j^l)$ is constant with j, equation (4.37) is finally obtained. The constants β_1 and β_2 are normalization constants ensuring that the left-hand side sums up to one over all c_l.

Note that equation (4.37) is a Bayes filter that can be used recursively to update the belief $p(c_l \mid \mathcal{D}_n^l)$ whenever a new influencing distance D_n^l is obtained. This is done by multiplying the current coverage belief $p(c_l \mid \mathcal{D}_{n-1}^l)$ and the coverage estimate $p(c_l \mid D_n^l)$, provided by the sensor model and the new influencing measurement D_n^l, and applying the normalization factor $\beta_1.\beta_2$. Note also that for $n = 1$, *i.e.* for the first influencing measurement, this recursive procedure uses the initial belief $p(c_l \mid \mathcal{D}_0^l)$, with $D_0^l = \emptyset$.

Consider equation (4.23) giving the set of influencing measurements of a voxel $l \in \mathcal{Y}$ up to the k-th batch of measurements. Recall that $\mathcal{D}_{k-1}^l = \{D_1^l, \ldots, D_{n_{k-1}(l)}^l\}$ is the set of $n_{k-1}(l)$ measurements influencing the coverage estimate of that voxel until the $(k-1)$-th batch of measurements, and that $p(c_l \mid \mathcal{D}_{k-1}^l)$ is the associated voxel's coverage belief. When the range sensor provides the k-th batch of measurements M_k at $t = t_k$, some measurements are eventually appended to the set $\mathcal{D}_{k-1}^l$, which yields a new set of measurements

$$\mathcal{D}_k^l = \{D_1^l, \ldots, D_{n_{k-1}(l)+1}^l, \ldots, D_{n_k(l)}^l\}, \tag{4.38}$$

having cardinality $n_k(l) > n_{k-1}(l)$. Thus, the set of measurements that have just been appended is

$$\mathcal{D}_{k-1}^l \cap \mathcal{D}_k^l = \{D_j^l : j \in \mathbb{N},\ n_{k-1}(l) + 1 \leq j \leq n_k(l)\}. \tag{4.39}$$

The voxel's coverage belief after the k-th batch of measurements can be computed by using recursively equation (4.37) for all these new influencing measurements, as

$$p(c_l \mid \mathcal{D}^l_k) = \beta_3 . \Big[\prod_{j=n_{k-1}(l)+1}^{n_k(l)} p(c_l \mid D^l_j) \Big] . p(c_l \mid \mathcal{D}^l_{k-1}), \tag{4.40}$$

wherein β_3 is a normalization constant ensuring that the left-hand side sums up to one over all c_l. Note that if $n_k(l) - n_{k-1}(l) = 0$, *i.e.* if there are no new influencing measurements provided by the k-th batch of measurements, $\mathcal{D}^l_k = \mathcal{D}^l_{k-1}$ and, obviously, $p(c_l \mid \mathcal{D}^l_k) = p(c_l \mid \mathcal{D}^l_{k-1})$. Note also that, accordingly with the equality given by equation (4.26), page 110, we have $p(c_l \mid \mathcal{M}_{k-1}) = p(c_l \mid \mathcal{D}^l_{k-1})$ and $p(c_l \mid \mathcal{M}_k) = p(c_l \mid \mathcal{D}^l_k)$.

4.5.2.1 Special case of updating Gaussians

Accordingly with the sensor model proposed in section 4.4.4, page 110, new coverage estimates $p(c_l \in D^l_j)$ of any voxel $l \in \mathcal{Y}$, upon any influencing measurement D^l_j, are always Gaussian distributions. Furthermore, the initial belief $p(c_l \mid \mathcal{D}^l_0)$ of any voxel $l \in \mathcal{Y}$ (see section 4.5.1) is whether a maximum entropy (non informative) Gaussian with $\sigma \to \infty$, or a Gaussian $N(\mu, \sigma)$ with $\mu \in [0, 1]$, representing prior knowledge about the voxel's coverage. Thus, for the first influencing measurement D^l_1, and given the initial Gaussian belief $p(c_l \mid \mathcal{D}^l_0)$, equation (4.37) always involves the multiplication of two Gaussian distributions. If the resultant probability density function (pdf) is also a Gaussian, updating the voxel's coverage belief upon any measurement always involves the multiplication of two Gaussian distributions.

In fact, it can be easily shown (see appendix A.1, page 245) that the product of two Gaussians $p(c_l \mid \mathcal{D}^l_{n-1}) = N(\mu_1, \sigma_1)$ and $p(c_l \mid D^l_n) = N(\mu_2, \sigma_2)$ always yields a Gaussian multiplied by a constant [RDC05d, RDC05a]:

$$p(c_l \mid \mathcal{D}^l_{n-1}) . p(c_l \mid D^l_n) = \frac{1}{\beta} . N(\mu, \sigma), \tag{4.41}$$

$$\mu = \frac{\mu_1 \sigma_2^2 + \mu_2 \sigma_1^2}{\sigma_1^2 + \sigma_2^2}, \tag{4.42}$$

$$\sigma = \frac{\sigma_1 \sigma_2}{\sqrt{\sigma_1^2 + \sigma_2^2}}, \tag{4.43}$$

$$\beta = \sqrt{2\pi(\sigma_1^2 + \sigma_2^2)} \exp\left[\frac{(\mu_1 - \mu_2)^2}{2(\sigma_1^2 + \sigma_2^2)} \right]. \tag{4.44}$$

Comparing equations (4.37) and (4.41), we conclude that: updating the coverage belief of a voxel, between consecutive influencing measurements, is as simple as computing the parameters of a new Gaussian distribution through the closed form equations (4.42) and

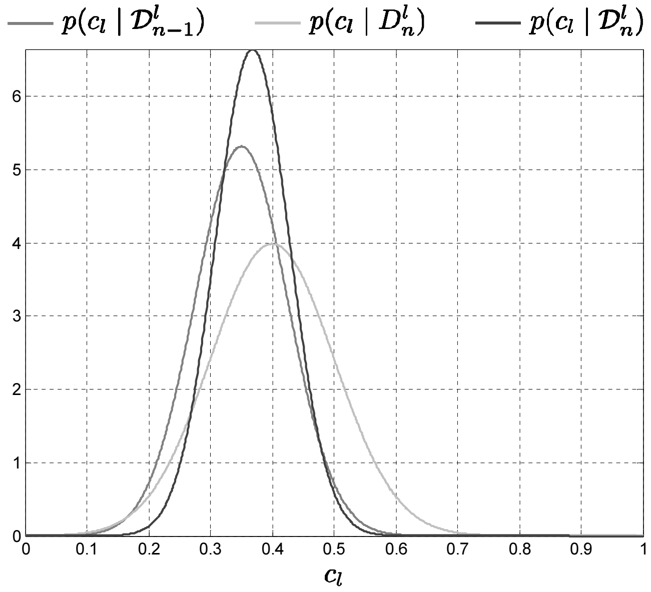


Figure 4.8: Example of the voxel's update procedure with Gaussian distributions. The following Gaussian distributions are depicted: $p(c_l \mid \mathcal{D}^l_{n-1}) = N(0.35, 0.075)$, $p(c_l \mid D^l_n) = N(0.4, 0.1)$ and $p(c_l \mid \mathcal{D}^l_n) = N(0.368, 0.06)$.

(4.43); and the normalization constant is just $\beta_1.\beta_2 = \beta$, being β given by equation (4.44). This simplicity of computation is a consequence of the Gaussian nature of the proposed sensor model and the careful choice of an initial coverage belief.

It is easy to conclude from equation (4.43) that the standard deviation σ of the Gaussian yielded by the product of two Gaussians with standard deviation σ_1 and σ_2, respectively, always verifies the condition $\sigma < \sigma_1 \wedge \sigma < \sigma_2$, *i.e.* the new voxel's coverage belief has always lower standard deviation and, accordingly with equation (3.39), page 94, lower differential entropy and uncertainty than the previous coverage belief [RDC05d, RDC05a].

Fig. 4.8 shows an example of the aforementioned update procedure. The differential entropy value for the depicted Gaussian probability density functions are $h(C_l \mid \mathcal{D}^l_{n-1}) = -1.690$ bits, $h(C_l \mid D^l_n) = -1.275$ bits and $h(C_l \mid \mathcal{D}^l_n) = -2.012$ bits. The entropy value of the associated quantized versions are $H(C_l \mid \mathcal{D}^l_{n-1}) = 5.312$ bits, $H(C_l \mid D^l_n) = 5.726$ bits and $H(C_l \mid \mathcal{D}^l_n) = 4.991$ bits.

Although the domain of a Gaussian distribution is not restricted to the interval $[0, 1]$, accordingly with equations (4.28) and (4.42), we can conclude that $0 \leq \mu_l \leq 1$. In practice, truncating the Gaussian to that interval is not strictily required to update the coverage belief (we have just to compute the new parameters μ_l and σ_l) but, if for some purpose we have to do it ([6]), we apply the normalization factor given by equation (4.31), page 112. Although the mean of the truncated Gaussian is different from μ_l, its mode is equal to μ_l and might be taken as a good estimate of the voxel's coverage, because that difference tends to zero provided that $\sigma_l \rightarrow 0$.

While Stachniss *et al.* represented the coverage belief of a cell through histograms having b bins (b is typically more than 10) [SB03], in the framework proposed herein the voxel's coverage belief is represented as a Gaussian pdf, which is fully characterized by just *two* parameters: μ_l and σ_l. Thus, in the set of probability density functions given by equation (4.5), page 103, only two values need to be stored in memory for each voxel, which is a much more compact representation than a histogram. Moreover, the aforementioned procedure for updating the voxels' coverage belief is very simple and we can still build histograms upon the pdf, with an arbitrary number of bins, through equation (4.7), page 104 [RDC05d, RDC05a].

4.6 Entropy gradient-based exploration

Previous sections present a grid-based probabilistic framework to represent a map and its associated uncertainty. Besides representing internally the map, the robot has also to be able to use its current map with the aim of navigating and exploring autonomously the environment, by selecting new exploration viewpoints. The goal is to explore less explored regions and reduce the map's uncertainty as fast as possible. This section presents a frontier-based exploration strategy, which uses the current map's entropy gradient [RDC05d, RDC05a].

In a mapping robotic mission, the objective is to acquire as much new information about the environment as possible with every sensing cycle. Yamauchi proposed *frontier-based exploration* for grid-based maps [Yam98] whereby a robot is driven towards boundaries between open space and unexplored regions, by selecting the closest frontier cell in its neighborhood. In the Yamauchi's strategy, a cell may be in one of three states: occupied, unoccupied or explored. The strategy seeks for unoccupied cells having a frontier with other unexplored cells.

The Yamauchi's exploration concept is herein reformulated through the notions of voxel's entropy and map's entropy and the key point is to realize that map's entropy gradient is maximum in frontier voxels between more explored and less explored regions. When a robot has to select a new viewpoint for acquiring data through its sensor, the

[6] For instance, the purpose may be to compute a cumulative probability, such as $P(0 \leq C_l \leq 0.3)$.

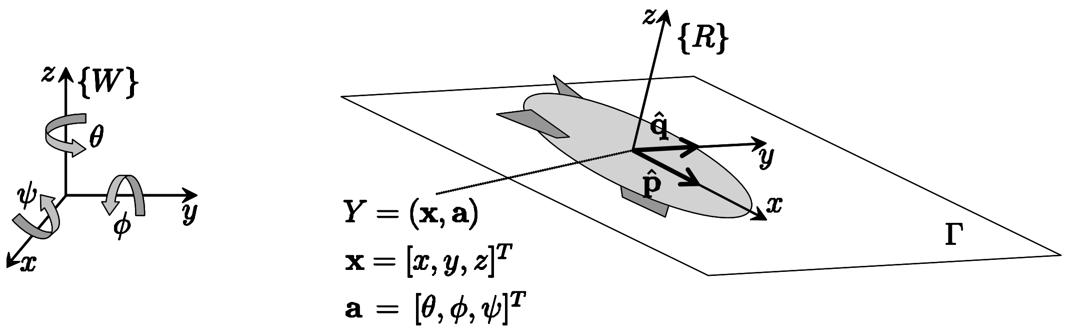


Figure 4.9: Robot's pose and associated coordinates reference frames: on the left, the global (absolute) coordinates reference frame $\{W\}$ and the three Euler angles θ, ϕ and ψ, respectively, the yaw, pitch and roll angles of the robot; on the right, the robot's pose $Y = (\mathbf{x}, \mathbf{a})$, the robot's reference frame $\{R\}$ and the two orthogonal vectors $\hat{\mathbf{p}}$ and $\hat{\mathbf{q}}$ defining the robot's sensor motion plane Γ. The robot's heading is represented through vector $\hat{\mathbf{p}}$.

robot's sensor is directed to voxels having higher magnitudes of entropy gradient and low expected coverage in the neighborhood of the robot, *i.e.* the robot seeks for frontier voxels that are more likely to be unoccupied. This strategy aims at maximizing the information gain of new acquired data, so that the exploration is more efficient [RDC05d, RDC05c, RDC05a].

4.6.1 Method for a mobile robot with 6 DOF

For the more general mobile robot moving in a 3-D workspace, the robot's sensor pose $Y = (\mathbf{x}, \mathbf{a})$ is defined through three cartesian coordinates $\mathbf{x} = [x, y, z]^T$ giving its position and three Euler angles $\mathbf{a} = [\theta, \phi, \psi]^T$ giving its attitude (see Fig. 4.9). These orientation angles are assumed to be positive in the counterclockwise direction and have the following definition: angle θ is the yaw angle, being a rotation about the zz axis; angle ϕ is the pitch angle, being a rotation about the yy axis; angle ψ is the roll angle, being a rotation about the xx axis. Hereafter, the expressions *robot' pose*, *sensor's pose* and *robot's sensor pose* are assumed top be interchangeable and any of them refer to the same quantity.

Let denote the applied vector connecting the robot's sensor position $\mathbf{x} \in \mathbb{R}^3$ to the center of a given voxel l as $\overrightarrow{\mathbf{u}}(\mathbf{x}, l) = \mathbf{w}(l) - \mathbf{x}$. Given a neighborhood around the current robot's sensor position with radius ε, the strategy selects for its new position the center of a given voxel belonging to the set of voxels

$$\mathcal{N}(\mathbf{x}, \varepsilon) = \{l : l \in \mathcal{Y},\ \|\mathbf{u}(\mathbf{x}, l)\| \leq \varepsilon\}. \tag{4.45}$$

The map's entropy $H(C_l)$ given by equation (4.9), page 105, is a measure of the map's uncertainty. On the other hand, being $\mathbf{w}(l)$ the center of a given voxel $l \in \mathcal{Y}$, the volumetric grid $\mathcal{Y}$ discretises the 3-D space $\mathbb{R}^3$ at discrete points $\mathbf{w}(l)$, $l \in \mathcal{Y}$, equally spaced by the voxel's edge ϵ. Thus, the probabilistic map allows to associate with each of these points an entropy value $H(l) = H(C_l)$ computed through equation (4.8), wherein a continuous entropy field $H : \mathbb{R}^3 \rightarrow \mathbb{R}$ is sampled along the centers of the voxels belonging to the grid $\mathcal{Y}$. The strategy seeks for neighbor regions having higher magnitudes of entropy gradient $\overrightarrow{\nabla} H$ and regions more likely unoccupied, in the neighborhood of the robot.

Let $l_{\Theta-}$ denote the contiguous voxel to l in the negative direction of axis Θ. A reasonable (first order) approximation to the entropy gradient at the center of a voxel l is

$$\overrightarrow{\nabla} H(l) \approx \frac{1}{\epsilon}\left[H(l) - H(l_{x-}), \quad H(l) - H(l_{y-}), \quad H(l) - H(l_{z-})\right]^T, \tag{4.46}$$

having magnitude $\left\|\overrightarrow{\nabla} H(l)\right\|$.

If the center of a voxel $l \in \mathcal{N}(\mathbf{x}, \varepsilon)$ is selected to be the next robot's selected position $\mathbf{x}^s$, the associated gaze direction $\mathbf{a}(l)$ is defined by the unitary vector

$$\hat{\mathbf{p}}(l) = \frac{\overrightarrow{\nabla} H(l)}{\left\|\overrightarrow{\nabla} H(l)\right\|}, \quad \overrightarrow{\nabla} H(l) \neq \overrightarrow{0}. \tag{4.47}$$

Being $E(C_l)$ the expected coverage of a voxel $l \in \mathcal{Y}$, and given the set of voxels $\mathcal{N}(\mathbf{x}, \varepsilon)$ located in the robot's neighborhood, the strategy is to direct the robot's sensor to the voxel

$$\boxed{l^s = \underset{l \in \mathcal{N}(\mathbf{x}, \varepsilon)}{\operatorname{argmax}} \left(\left\|\overrightarrow{\nabla} H(l)\right\| . [1 - E(C_l)]\right),} \tag{4.48}$$

with a gaze on arrival defined by the unitary vector $\hat{\mathbf{p}}(l^s)$. If the gradient-based criteria is not conclusive, *i.e.* if $\overrightarrow{\nabla} H(l) = \overrightarrow{0}$, $\forall_{l \in \mathcal{N}(\mathbf{x}, \varepsilon)}$, the robot should wander randomly until that condition is not verified.

4.6.2 Method for a ground mobile robot with 3 DOF

Equations (4.47) and (4.48) do not cope with robot's kinematic restrictions and assume that the robot is able to move freely in the 3-D workspace with 6 DOF. This is suitable for an unmanned aerial vehicle (UAV), but it is not an appropriate model for a ground mobile robot, having just 3 DOF. Since the proposed method was used on ground mobile robots, whose sensor's motion is instantaneously restricted to a plane Γ parallel to the robot's motion plane (*e.g.* the floor plane in an indoor workspace), this section adapts the method presented previously to a robot with 3 DOF: two position coordinates x and

y and an orientation angle θ (see Fig. 4.9). The key point of this 3 DOF version of the exploration method is that voxels which are intersected by the plane Γ are preferable to be explored [RDC05d, RDC05c, RDC05a].

Consider again the current robot's pose $Y = (\mathbf{x}, \mathbf{a})$, being $\mathbf{x}$ its current position and $\mathbf{a} = [\theta, \phi, \psi]^T$ its attitude. Given the robot's coordinates reference frame $\{R\}$, equal to the global (absolute) coordinates references frame $\{W\}$ after translation and rotation, the robot's motion plane Γ is defined by two orthogonal axes (see Fig. 4.9): a longitudinal axis $\hat{\mathbf{p}}' = [1, 0, 0]^T$, which is the unitary vector along xx axis, and a transverse axis $\hat{\mathbf{q}}' = [0, 1, 0]^T$, which is the unitary vector along yy axis; for example, for an UAV, $\hat{\mathbf{p}}$ would be an axis pointing to the front of the plane with direction between tail and head, and $\hat{\mathbf{q}}$ would be an axis connecting the plane's body to its left wing.

It can be shown that these robot's axes can be expressed in the global coordinates reference frame $\{W\}$ as [RDC05a]

$$\hat{\mathbf{p}} = \begin{bmatrix} \cos\theta.\cos\phi & \sin\theta.\cos\phi & -\sin\phi \end{bmatrix}^T, \tag{4.49}$$

$$\hat{\mathbf{q}} = \begin{bmatrix} \cos\theta.\sin\phi.\sin\psi - \sin\theta.\cos\psi \\ \sin\theta.\sin\phi.\sin\psi + \cos\theta.\cos\psi \\ \cos\phi.\sin\psi \end{bmatrix}. \tag{4.50}$$

Recall that angles θ, ϕ and ψ are the yaw angle, the pitch angle and the roll angle, respectively, and are assumed to be positive in the counterclockwise direction. Note that axis $\hat{\mathbf{p}}$ represents the robot's sensor gaze direction, *i.e.* the robot's heading.

Any vector $\vec{\mathbf{u}}$ can be projected on the robot's sensor motion plane Γ as

$$\underset{\Gamma}{\text{proj}}\ \vec{\mathbf{u}} = (\vec{\mathbf{u}} \cdot \hat{\mathbf{p}})\hat{\mathbf{p}} + (\vec{\mathbf{u}} \cdot \hat{\mathbf{q}})\hat{\mathbf{q}}, \tag{4.51}$$

wherein $(\cdot)$ denotes the internal product of two vectors.

Let denote the applied vector connecting the robot's sensor position $\mathbf{x} \in \mathbb{R}^3$ to the center of voxel l as $\vec{\mathbf{u}}(\mathbf{x}, l) = \mathbf{w}(l) - \mathbf{x}$. Given a neighborhood around the current robot's sensor position with radius ε, its new position is selected as the center of a voxel belonging to the set of voxels

$$\mathcal{N}_\Gamma(\mathbf{x}, \varepsilon) = \{l : l \in \mathcal{Y},\ \|\vec{\mathbf{u}}(\mathbf{x}, l)\| \leq \varepsilon,\ l = v(\underset{\Gamma}{\text{proj}}\ \mathbf{w}(l))\}. \tag{4.52}$$

Any voxel $l \in \mathcal{N}_\Gamma(\mathbf{x}, \varepsilon)$ satisfies two conditions: plane Γ intersects the voxel, *i.e.* it is near the robot's sensor motion plane; and the voxel's distance to the robot's current position is less or equal to ε.

Given a voxel $l \in \mathcal{Y}$ and its entropy gradient $\overrightarrow{\nabla} H(l)$ computed through equation (4.46), the projection of the entropy gradient on the robot's sensor motion plane Γ is

$$\overrightarrow{\nabla} H_\Gamma(l) = \operatorname*{proj}_{\Gamma} \overrightarrow{\nabla} H(l), \tag{4.53}$$

having magnitude $\left\| \overrightarrow{\nabla} H_\Gamma(l) \right\|$.

If the center of a given voxel $l \in \mathcal{N}_\Gamma(\mathbf{x}, \varepsilon)$ is selected to be the next robot's selected position $\mathbf{x}^s$, the robot selects the associated attitude $\mathbf{a}(l)$ defined by the unitary vector

$$\hat{\mathbf{p}}(l) = \frac{\overrightarrow{\nabla} H_\Gamma(l)}{\left\| \overrightarrow{\nabla} H_\Gamma(l) \right\|}, \quad \overrightarrow{\nabla} H_\Gamma(l) \neq \overrightarrow{0}. \tag{4.54}$$

Being $E(C_l)$ the expected coverage of a voxel $l \in \mathcal{Y}$, and given the set of voxels $\mathcal{N}_\Gamma(\mathbf{x}, \varepsilon)$ that are intersected by the robot's sensor motion plane Γ and are located in the robot's neighborhood, the robot's sensor is directed to the voxel

$$\boxed{l^s = \operatorname*{argmax}_{l \in \mathcal{N}_\Gamma(\mathbf{x}, \varepsilon)} \left(\left\| \overrightarrow{\nabla} H_\Gamma(l) \right\| . [1 - E(C_l)] \right),} \tag{4.55}$$

with a gaze on arrival defined by the unitary vector $\hat{\mathbf{p}}(l^s)$ [RDC05d, RDC05c, RDC05a]. If the gradient-based criteria is not conclusive, the robot should wander randomly until that condition is not verified.

Note the similarities between the pairs of equations (4.45) and (4.52), (4.47) and (4.54), and (4.48) and (4.55).

4.7 Implementation in mobile robots

Before presenting experimental results that validate the framework presented in previous sections, this section describes some details about the experimental setup, namely the robots, the stereo-vision sensors and their calibration, the global localization system based on a color camera, and the robots' control software. This experimental setup is comprised of Scout robots running a Linux operating system, software specifically developed to implement the framework presented previously, stereo-vision sensors and their associated software for computing range data, and a desktop PC connected to a color camera that provides the robots with global localization.

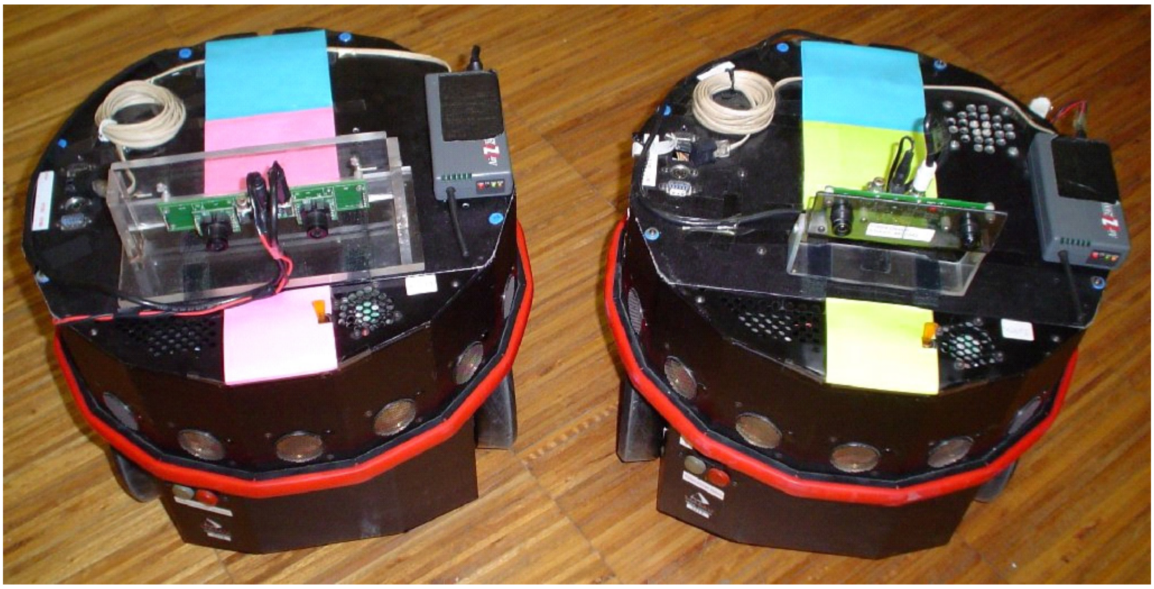


Figure 4.10: Scout mobile robots from Nomadic Technologies, Inc., equipped with sonars, stereo-vision and a modem radio providing TCP/IP wireless communication. Both robots have colored markers on the top (rectangular areas with different colors) in order to enable global localization, through color segmentation in the image provided by a global color camera covering the robots' workspace.

4.7.1 Robots

Fig. 4.10 depicts a photo of the two robots that were used in the 3-D mapping experiments performed indoor, at the Mobile Robotics Laboratory, ISR — Institute of Systems and Robotics —, University of Coimbra, Portugal.

Each robot is a Scout mobile robot from Nomadic Technologies, Inc., manufactured in 1999, having differential kinematics, odometry sensing and ultrasonic sensing [Nom99]. At a lower level, it uses a special multiprocessor control system that controls the sensing and motion. At a higher level, the robot is controlled by an embedded computer, with a Pentium 133 MHz processor (no MMX) and 32 Mb of RAM. It is powered through two 12 V, 17 A-h, chargeable lead-acid batteries, which can be either charged by plugging in a 220 V connection to the robot, or through an external charger ([7]). The robot's autonomy is highly sensible to the robot's usage. When used to carry out mapping missions, the robot's batteries lasted, on average, for 3 hours. The values of the robots' most relevant parameters are presented in Table B.3, page 250.

[7] The robot's batteries were often charged externally in order to extend the batteries' lifetime as much as possible, though being easier to use the robot's own power supply to charge the batteries.

The robot's embedded computer runs a Linux operating system — the RedHat 7.0 Linux distribution — installed in a hard disk partition with about 1 Gb, which includes a X-Windows system with resolution 640 x 480, a C/C++ compiler and most of the C/C++ libraries.

The robot has a ring of 16 Polaroid 6500 sonar ranging modules, which can measure distances from 15 *cm* to 11 *m* ([8]). Because stereo-vision range sensors cannot measure distances below a given distance threshold, which depends on the chosen parameters for the algorithm that computes range data, sonars were used for preventing the robot to acquire stereo image pairs too close to obstacles. They were also used for collision avoidance when moving the platform.

A stereo-vision sensor and a modem radio providing wireless TCP/IP communication were mounted on the top of each robot. This equipment is not an integral part of the Scout robot; it was added to fulfill the requirements of the mapping experiments. While the former one provides the robot with range measurements, the latter one allows the robot to communicate with other robots and other computers in the network. The wireless communication speed is about 100 Kbit/s.

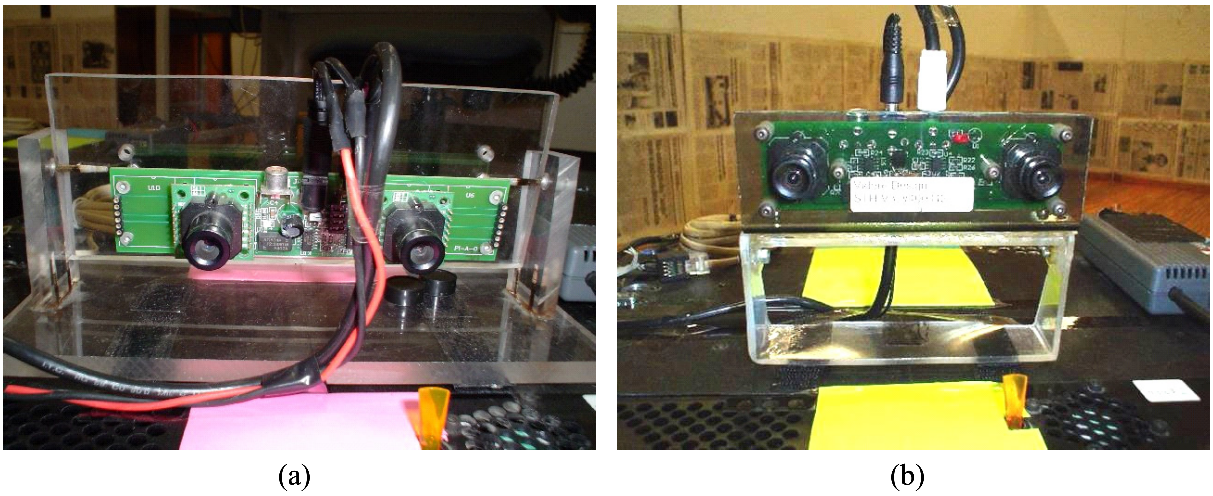
Colored markers (rectangular areas with different colors) were put on the top of the robots, in order to localize them through a global camera and color segmentation. See section D, page 263 for more details about this localization method.

Although the experiments used two mobile robots, which were available in the research lab, they could be readily extended to teams with more robots. In fact, as we shall see in chapter 6, computer simulations were used to perform experiments with teams containing more than two robots, thus extending the results obtained with the two mobile robots described above.

4.7.2 Stereo-vision sensors

The robots' ability to measure distances was implemented through stereo-vision sensors. There were several reasons on the basis of choosing these sensors and not choosing, for example, laser range finders. The main three reasons were: those sensors could be easily found in the laboratory where the experiments were carried out; they have been used to support research on computer vision by other researchers in the group, who possess important know-how about the technology [LAD02]; and they intrinsically provide large

[8] Accordingly with the manufacturer [Nom99], the robot's sonars can measure distances between 6 *in.* and 35 *ft*.



(a) (b)

Figure 4.11: Stereo-vision sensors mounted on the robots. Both stereo-vision sensors are compact, low-cost analog stereo rigs from Videre Design [Vid05], with resolution 160x120 pixels, comprising two CMOS cameras: (a) STH-V2 stereo rig; (b) STH-V3 stereo rig.

amounts of 3-D data.

The sensors that were mounted on the robots are depicted in Fig. 4.11. Their main characteristics are summarized in Table B.4, page 251. Each stereo-vision sensor is a small, compact, low-cost analog stereo rig from Videre Design with a fixed baseline ([9]), comprising two synchronized monochrome CMOS cameras modules mounted on a base-board with resolution 320 x 240 pixels [Vid05]. Since the video output provides interlaced stereo image pairs, the devices's resolution is half the resolution of each individual camera, *i.e.* 160 x 120 pixels.

A Pinnacle Studio PCTV frame grabber (Bt878-based card) was mounted on the PCI slot of each robot's embedded computer, in order to acquire the video signal from the stereo-vision sensor and convert it to a digital format.

For computing range data from stereo images, the SVS — Small-Vision System — version 2.3c, a stereo engine from SRI International, was used [KB02]. This software provides the user with a set of functions to acquire and display video from a binocular stereo-vision system, calibrate the stereo rig, compute range data from an image pair, *etc.* Further details about the SVS software are given in appendix C, page 255.

In the specific SVS software application that was programmed to carry out mapping experiments with the stereo-vision sensors mounted on the robots of Fig. 4.10, the stereo parameters were empirically set to the values presented in Table B.1, page 249. With

[9] The baseline is the distance between the optical centers of the two cameras.

these parameters, the sensors shown in Fig. 4.11, have an horopter ([10]) covering distances roughly between 0.75 m and 4 m.

If the stereo-vision sensor is used to measure an object that falls outside the horopter, the true disparity is not found and some distribution of random disparities is returned. The upper limit is not relevant for the mapping experiments that are reported herein, because measured distances are almost always below it. However, the lower limit must be carefully avoided in order to avoid outliers in the range measurements. Before a robot acquire a stereo pair, it uses its frontal sonars to check if the sensor's distance to the obstacles is greater than the lower distance that it can successfully measure. Then, the robot uses the stereo-vision to get 3-D range data only, *and if only*, the stereo-vision sensor is not too close to obstacles.

4.7.3 Calibration of the parameters of the sensor model

The sensor model of a range sensor given by equation (4.29), page 111, includes two parameters — σ_{min} and ζ — that must be properly calibrated for the stereo-vision range sensors depicted in Fig. 4.11. In order to estimate those parameters, one of the stereo-vision sensors was used to measure known distances d to a planar obstacle — a checkerboard — parallel to the cameras' image plane. Fig. 4.12 depicts the set of image pairs that were used in the calibration.

Each stereo pair was obtained with the checkerboard located at different distances d within the stereo-vision sensor's horopter limits, ranging from 79 cm to 200 cm. For each distance d, the variability of the measured distance by the sensor over 100 trials was statistically described by the measurement's standard deviation, by comparing the values yielded by the sensor against the theoretical distance d between the optical center of the right camera and the respective point in the known planar target.

A fixed set of points sparse in the image plane were used for measuring each sampled distance d. Fig. 4.13 shows the obtained results for each one of the chosen points in the image plane. Fitting the curves to the linear model represented by equation (4.29), and averaging the parameters obtained for each point in the image plane, the values $\sigma_{min} = -0.06\ mm$ and $\zeta = 3.75 \times 10^{-3}$ were found.

At first sight, it may seem strange a negative value for the parameter σ_{min}, but recall that the sensor's horopter imposes a lower limit for the range measurements, which is equal to roughly 0.75 m. For this minimum range, the sensor model yields indeed a

[10] See section C.1, page 258, for the definition of horopter.

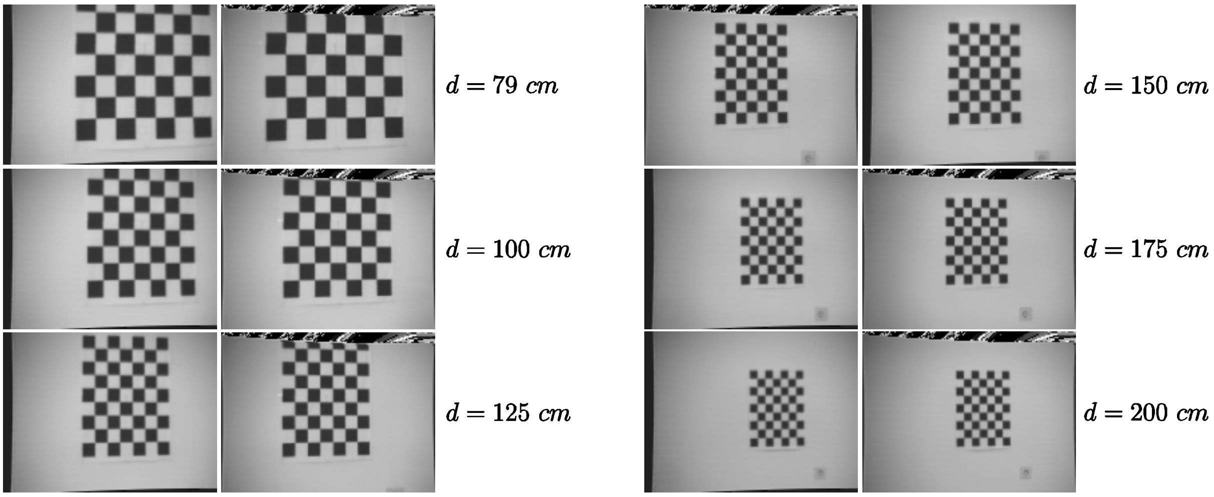


Figure 4.12: Set of image pairs used to calibrate the parameters of the sensor model. Every image pairs focus on a checkerboard — a planar obstacle — that is parallel to the image plane of both cameras of the stereo-rig. Each stereo pair was acquired with a different distance d between the stereo-rig and the checkerboard.

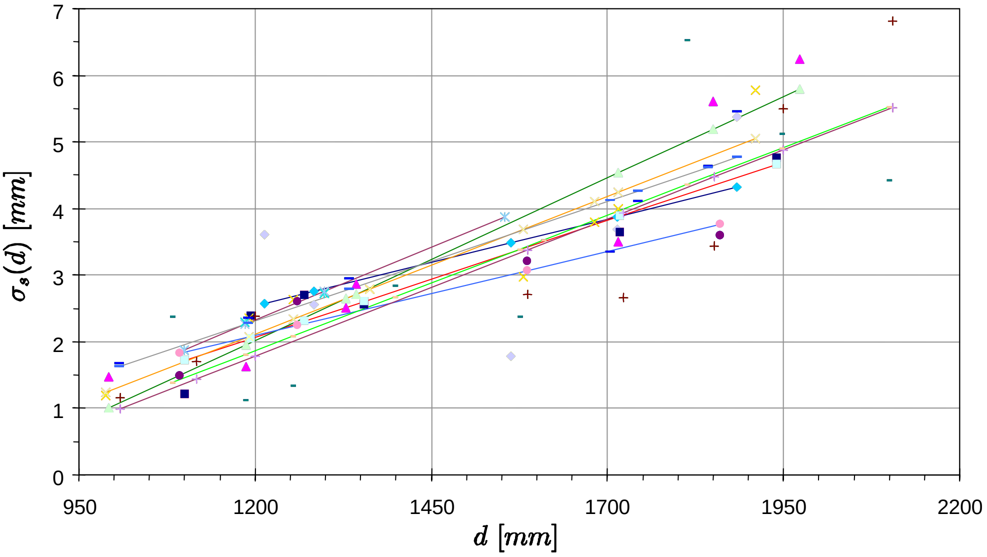


Figure 4.13: Calibration results for the parameters of the sensor model. The graph shows the sensor's standard deviation σ_s at each chosen point in the image plane as a function of the distance d to the planar obstacle, and also the curves fitting to a linear model. Averaging all those linear models yielded the parameter values $\sigma_{min} = -0.06\ mm$ and $\zeta = 3.75 \times 10^{-3}$ for the equation (4.29).

positive standard deviation equal to 2.753 mm and, obviously, it always yields a positive value within the horopter limits.

Accordingly with equation (4.30), page 111, the uncertainty of the coverage estimate of a given voxel is higher if the voxel is closer to the measured obstacle, and decays exponentially for farther distances given a damping ratio τ. The value of this parameter was empirically chosen after performing some mapping experiments, being set to $\tau = 2\ m$.

The values of the sensor model's parameters are also presented in Table B.2, page 250, for further reference.

4.7.4 Relating different coordinates reference frames

The grid-based volumetric model depicted in Fig. 4.2-a, page 102, assumes that the position of a given voxel $l \in \mathcal{Y}$ is expressed on an global coordinates reference frame $\{W\}$ — the world coordinates reference frame. However, there are other important coordinates reference frames that should be considered.

Fig. 4.14 shows that, besides $\{W\}$, it is useful to consider: the robot's reference frame $\{R\}$, centered on the platform's center, on the floor, whose xx axis is aligned with the robot's heading, pointing to the robot's motion direction, being the zz axis orthogonal to the floor; the robot's sensor reference frame $\{C\}$, centered on the right camera's optical center, wherein the zz axis is parallel to the cameras' optical axes and the xx axis is parallel to the images' lines (orthogonal to the images' columns). It is important to understand the relation between these coordinate systems and how a position expressed on a given system can be converted to other systems.

Consider a 3-D point whose position is expressed on $\{W\}$ as $\mathbf{x} = [x, y, z]^T \in \mathbb{R}^3$. Consider the coordinates of the same point expressed on $\{R\}$ as $\mathbf{x}_R = [x_R, y_R, z_R]^T$ and expressed on $\{C\}$ as $\mathbf{x}_C = [x_C, y_C, z_C]^T$. When the stereo-vision sensor yields the coordinates $\mathbf{x}_C$ of that point on $\{C\}$, they can be expressed on $\{R\}$ as

$$\mathbf{x}_R =^R \mathbf{T}_C \begin{bmatrix} \mathbf{x}_C \\ 1 \end{bmatrix} = \begin{bmatrix} {}^R\mathbf{R}_C \big| {}^R\mathbf{t}_C \end{bmatrix}_{3\times 4} \begin{bmatrix} \mathbf{x}_C \\ 1 \end{bmatrix}, \tag{4.56}$$

wherein ${}^R\mathbf{T}_C$ is a 3×4 transformation matrix that is comprised of a 3×3 square rotation matrix ${}^R\mathbf{R}_C$ and a 3-D translation vector ${}^R\mathbf{t}_C$. Then, the coordinates can also be expressed on $\{W\}$ through a similar transformation

$$\mathbf{x} =^W \mathbf{T}_R \begin{bmatrix} \mathbf{x}_R \\ 1 \end{bmatrix} = \begin{bmatrix} {}^W\mathbf{R}_R \big| {}^W\mathbf{t}_R \end{bmatrix}_{3\times 4} \begin{bmatrix} \mathbf{x}_R \\ 1 \end{bmatrix}. \tag{4.57}$$

Combining equations (4.56) and (4.57), we have the overall transformation from $\{C\}$ to $\{W\}$

$$\mathbf{x} =^W \mathbf{T}_C \begin{bmatrix} \mathbf{x}_C \\ 1 \end{bmatrix} =^W \mathbf{T}_R \cdot^R \mathbf{T}_C \begin{bmatrix} \mathbf{x}_C \\ 1 \end{bmatrix}. \tag{4.58}$$

The transformation given by matrix ${}^R\mathbf{T}_C = \left[{}^R\mathbf{R}_C \middle| {}^R\mathbf{t}_C \right]_{3\times 4}$ is a rigid transformation that can be easily calibrated.

The translation vector ${}^R\mathbf{t}_C$ represents the coordinates of the right camera's optical center on the robot's reference frame $\{R\}$. As Fig. 4.15-a shows, it can be determined using a ruler and the stereo-rig's baseline. Considering the plane wherein the top of the robot's platform is embedded, the translation about xx axis is the distance between the platform's center and the projection of the right camera's optical center on that plane, which can be measured with a ruler (see the two measurements surrounded by an ellipse in Fig. 4.15-a). Since the stereo rig is centered with the robot's heading, the translation about yy axis is equal to half the baseline (its value is presented in Table B.4, page 251). The translation about the zz axis is the distance of the stereo rig's lens to the floor and can be measured again with a ruler.

In order to determine the rotation matrix ${}^R\mathbf{R}_C$, consider the checkerboard depicted in Fig. 4.14-b and its associated coordinates reference frame $\{B\}$. It is placed on the floor, in front of the robot, with its yy axis parallel to the robot's yy axis. Observe that $\{R\}$ can be obtained from $\{C\}$ after a rotation about zz axis of π radians and thus we have

$${}^R\mathbf{R}_C = \mathbf{R}_{zz}(\pi) \cdot^B \mathbf{R}_C = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot^B \mathbf{R}_C, \tag{4.59}$$

wherein ${}^B\mathbf{R}_C$ is the rotation matrix from $\{C\}$ to $\{B\}$.

Observe also that another way of obtaining $\{R\}$ from $\{C\}$ is the combination of two pure rotations, as

$${}^R\mathbf{R}_C = \mathbf{R}_{zz}\left(-\frac{\pi}{2}\right) \cdot \mathbf{R}_{xx}\left(-\frac{\pi}{2} - \alpha\right) = \begin{bmatrix} 0 & -\sin\alpha & \cos\alpha \\ -1 & 0 & 0 \\ 0 & -\cos\alpha & -\sin\alpha \end{bmatrix}. \tag{4.60}$$

The first rotation about the xx axis includes the stereo rig's tilt angle α towards the floor (see Fig. 4.15-b) and the second one is a rotation about the zz axis of $-\frac{\pi}{2}$ radians. The rotation matrix ${}^B\mathbf{R}_C$ can be easily determined through any general purpose camera calibration software, using a set of images of a checkerboard placed in different points, having different translations and rotations relative to the camera.

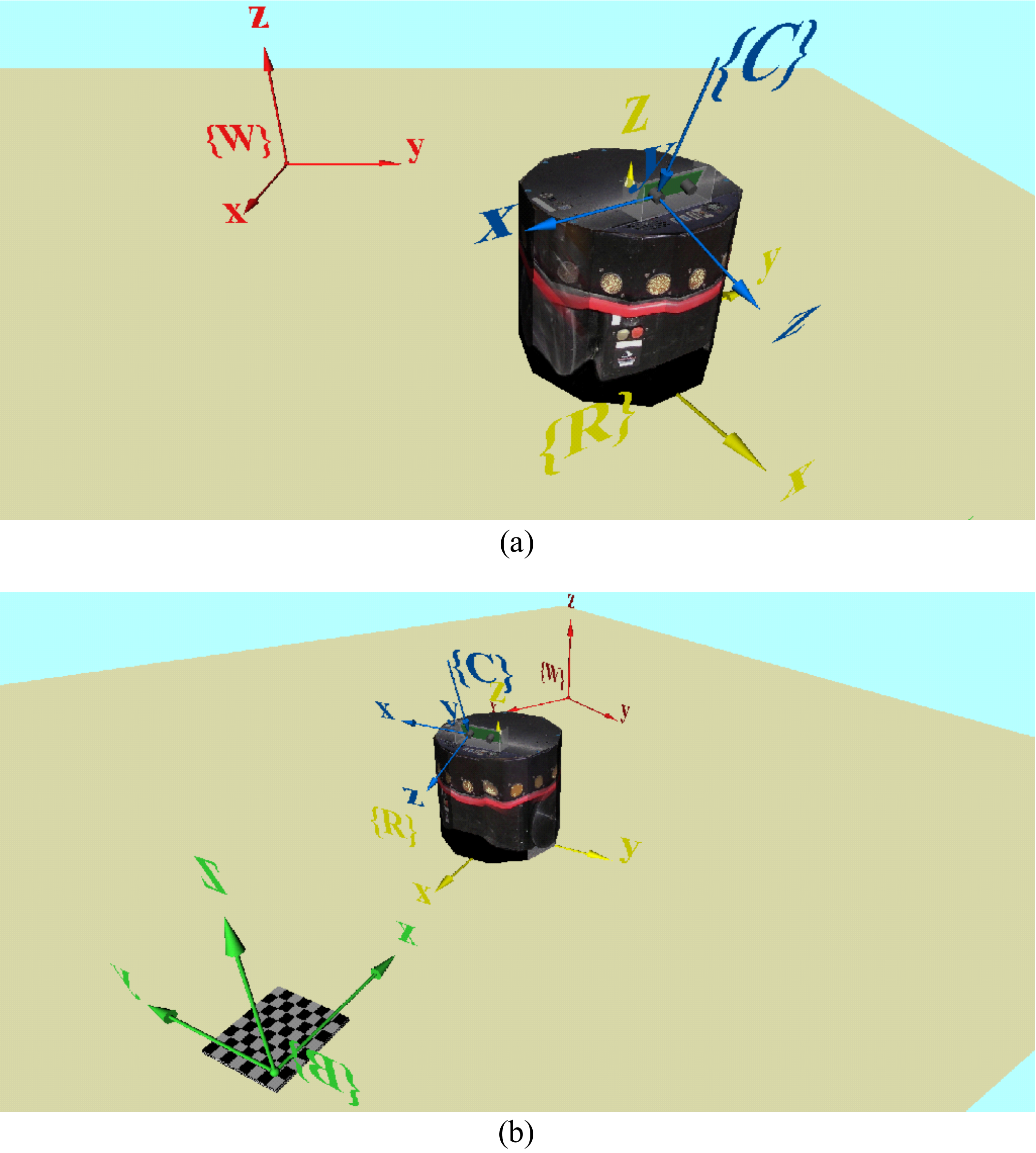


Figure 4.14: VRML model showing the different coordinates reference frames: (a) the global (world) reference frame $\{W\}$, the robot's reference frame $\{R\}$ and the robot's sensor (right camera) reference frame $\{C\}$; (b) additionally, the reference frame $\{B\}$ was used to calibrate the stereo rig's tilt angle towards the floor, through a checkerboard placed in front of the robot.

The camera calibration toolbox for Matlab© [Bou05], which is readily available, was used to determine the matrix ${}^{B}\mathbf{R}_{C}$. The calibration algorithm was provided with the image set depicted in Fig. 4.15-c, wherein the first image (top of the figure) was obtained with a checkerboard's placement that is similar to the one that is suggested by Fig. 4.14-b.

In some conditions that the user can easily control, the camera calibration software places the checkerboard reference frame for that first image as in Fig. 4.14-b, *i.e.* places it in the upper left vertex of the black square located in the upper left corner of the image. Thus, the rotation matrix (the extrinsic parameters) provided by the calibration software for that first image is just the matrix ${}^{B}\mathbf{R}_{C}$. The transformation ${}^{R}\mathbf{T}_{C}$ for the sensors depicted in Fig. 4.11, page 126, including rotation and translation, are presented in appendix B.2, page 253.

In equation (4.57), page 129, the transformation given by matrix ${}^{W}\mathbf{T}_{R} = \left[{}^{W}\mathbf{R}_{R} \middle| {}^{W}\mathbf{t}_{R} \right]_{3\times 4}$ is not rigid and must be given by some robot's localization scheme. During the mapping experiments herein reported, which were carried out in laboratory, a global localization scheme based on a global camera was used. It was chosen because cameras were readily available in the laboratory wherein the experiments took place.

This localization scheme might be extended to a network of cameras whose fields of view present some overlapping, in order to cover larger indoor areas. However, this was not required in the experiments reported herein, because, as it is shown in Fig. 4.16-a, the robots' workspace was confined to the part of a single room, wherein a single camera was able to cover it sufficiently. In more realistic environments, especially outdoor environments, this global localization scheme would be not viable and other localization schemes would have to be considered. However, this issue is out of the scope of this thesis.

The global localization scheme is based on a RGB analog color camera, which is depicted on the bottom-right of Fig. 4.16-a. The camera's video signal is acquired on a desktop PC (see the bottom-left of Fig. 4.16-a), where a localization server (see Fig. 4.16-b) uses color segmentation on the acquired images to provide the robots with localization information. See appendix D, page 263, for further details about this global localization scheme.

4.7.5 Software architecture

The organization of the software that was implemented to perform autonomous mapping missions with one of the robots of Fig. 4.10, page 124, is shown in Fig. 4.17. The software modules are distributed along the host PC depicted in Fig. 4.16-a, page 134,

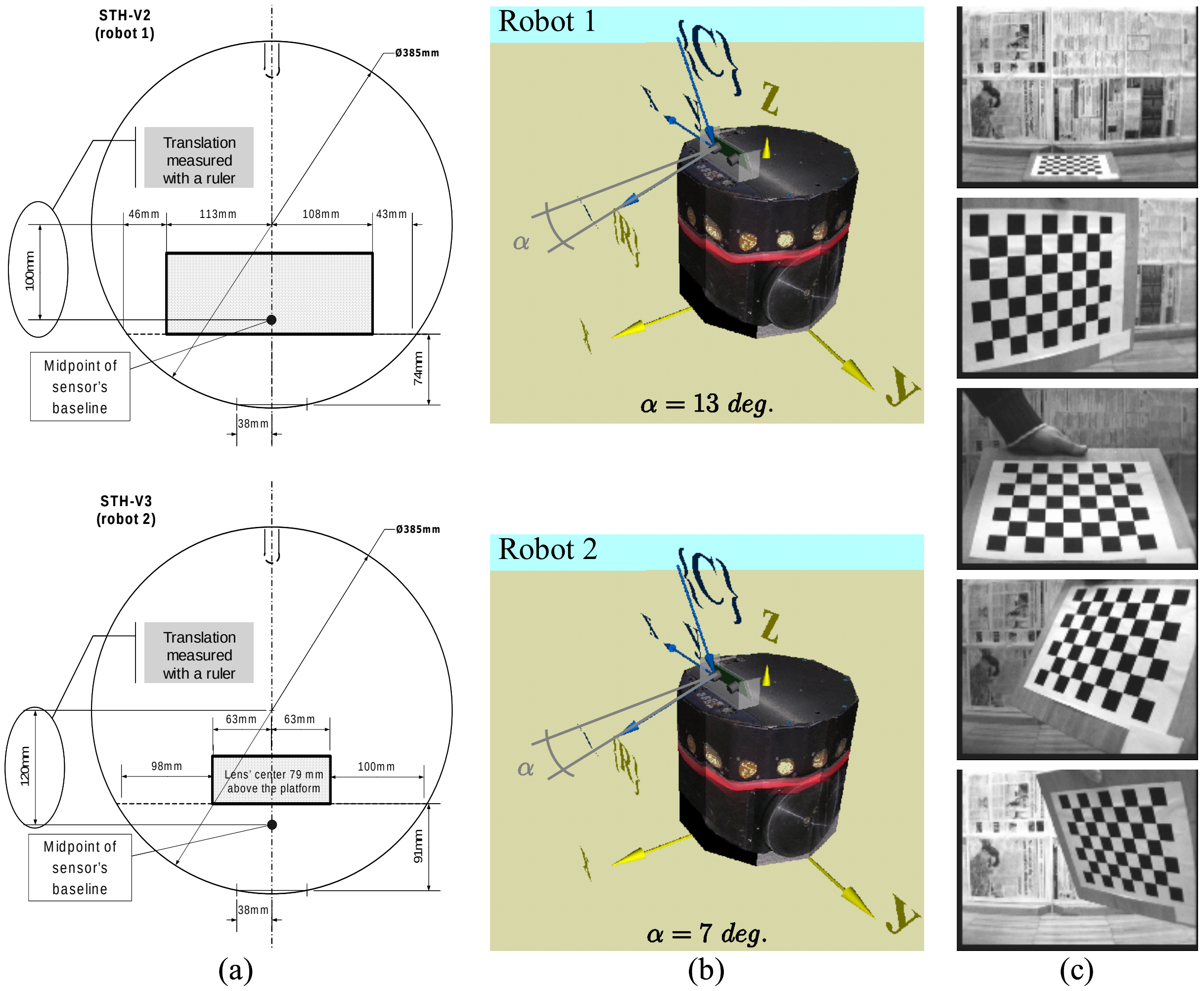
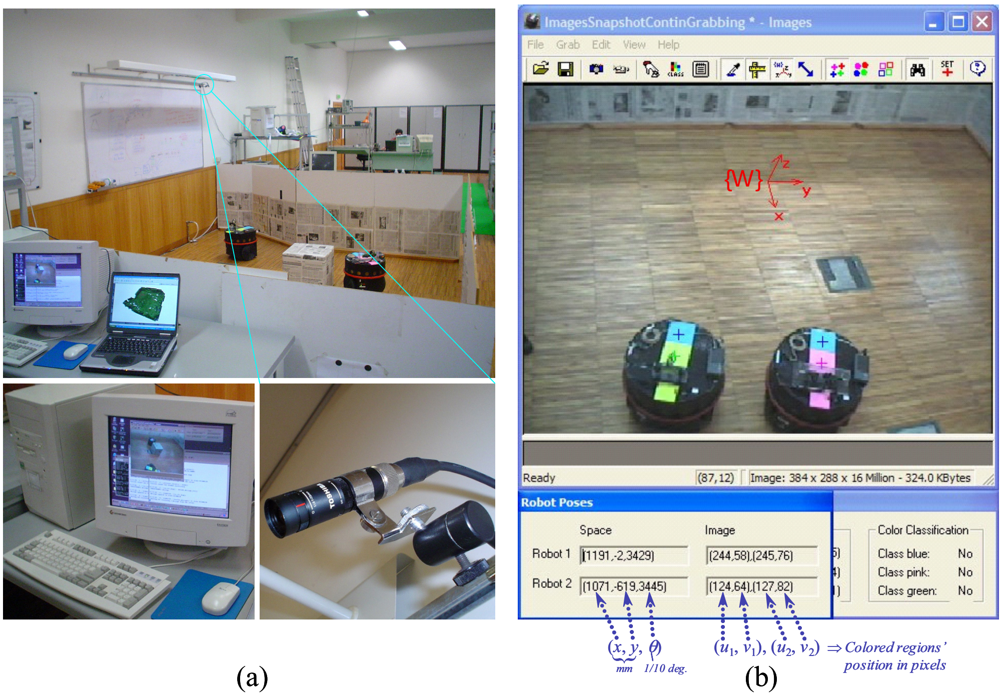

STH-V2
(robot 1)
Ø385mm
Translation measured with a ruler
46mm
113mm
108mm
43mm
100mm
Midpoint of sensor's baseline
74mm
38mm
STH-V3
(robot 2)
Ø385mm
Translation measured with a ruler
63mm
63mm
98mm
100mm
Lens' center 79 mm above the platform
120mm
Midpoint of sensor's baseline
91mm
38mm
Robot 1
Z
α
y
α = 13 deg.
Robot 2
Z
α
y
α = 7 deg.
(a)
(b)
(c)

Figure 4.15: Position and orientation of the sensor relative to the robot's platform: (a) translation measured with a ruler for both robots; (b) tilt angle towards the floor for the sensors of both robots; (c) image set used in the calibration of the stereo rig's tilt angle towards the floor of one of the robots.


(a)
ImagesSnapshotContinGrabbing * - Images
File Grab Edit View Help
{W}
z
y
x
Robot Poses
Color Classification
$\underbrace{(x, y}_{mm}, \underbrace{\theta)}_{1/10\ deg.}$
$(u_1, v_1), (u_2, v_2)$ ⇒ Colored regions' position in pixels
(b)

Ready (87,12) Image: 384 x 288 x 16 Million - 324.0 KBytes

	Space	Image
Robot 1	(1191,-2,3429)	(244,58),(245,76)
Robot 2	(1071,-619,3445)	(124,64),(127,82)

Class blue: No
Class pink: No
Class green: No

Figure 4.16: Global localization with a color camera: (a) RGB color camera from Toshiba covering the robots' workspace and the PC wherein the video signal is acquired through a frame grabber and further processed; (b) global localization server running on the PC.

and the robot itself, being interconnected through a TCP/IP computer network, wherein the robot uses wireless technology.

The host PC (see Fig. 4.17-a) runs four different software modules:

- `LOCALIZSRV` – *Localization Server*
  It is a TCP/IP server that is responsible for providing the robot with global localization through the global color camera that covers the robot's workspace (see section D, page 263, for further details).

- `MASTERCTR` – *Master Controller*
  The user interacts with this module in order to supervise and control the execution of 3-D mapping missions, by sending commands to the robot such as `start`, `end`, `pause` or `restore` a mission execution.

- `MAPCOLLECT` – *Map Collector*
  It is a TCP/IP server that is responsible for collecting incrementally data from the robot during the mission, in order to store and maintain a copy of the robot's map in the host PC's hard disk.

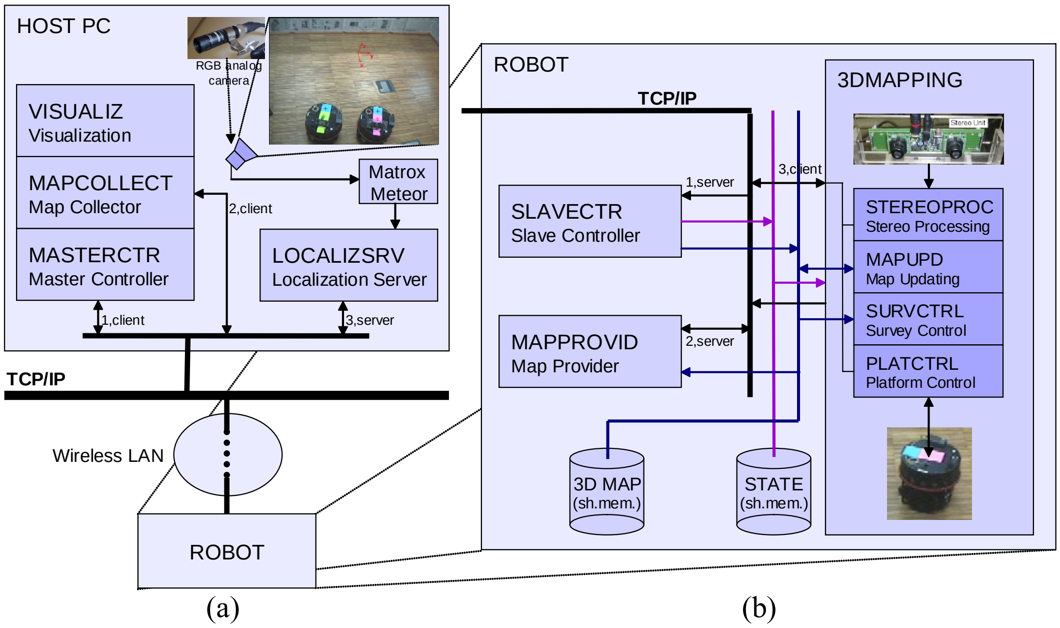


Figure 4.17: Diagram of the software architecture for one robot: (a) interaction between the robot and the host PC used for supervising 3-D mapping missions and providing global localization through a global camera; (b) software modules running locally on the robot.

- `VISUALIZ` – *Visualization*

  Builds a VRML — Virtual Reality Modeling Language — model for representing graphically and visualizing the maps collected by the module `MAPCOLLECT`.

The robot is able to build autonomously a volumetric map on its own, storing and maintaining locally the grid-based map on a shared memory area, which is shared by the different modules running concurrently in the robot (see Fig. 4.17-b). The mutual exclusive access to the map's shared memory is ruled through a semaphor provided by the Linux kernel. There is also a shared memory area for storing and sharing the robot's state among those modules. The robot's state may be `disabled`, `enabled` or `paused`, depending if whether the robot is waiting for the order to start a new mapping mission, or is currently performing a mapping mission, or is paused while performing a mapping mission, respectively.

The robot runs locally three software modules:

- `SLAVECTR` – *Slave Controller*

  It is a TCP/IP server that receives commands from the module `MASTERCTR` running remotely in the host PC; these commands determine the current robot's state. It creates the other processes running on the robot at the beginning of the mission (transition `disabled-enabled`). Conversely, it finishes those processes at the end of the mission (transition `enabled-disabled`).

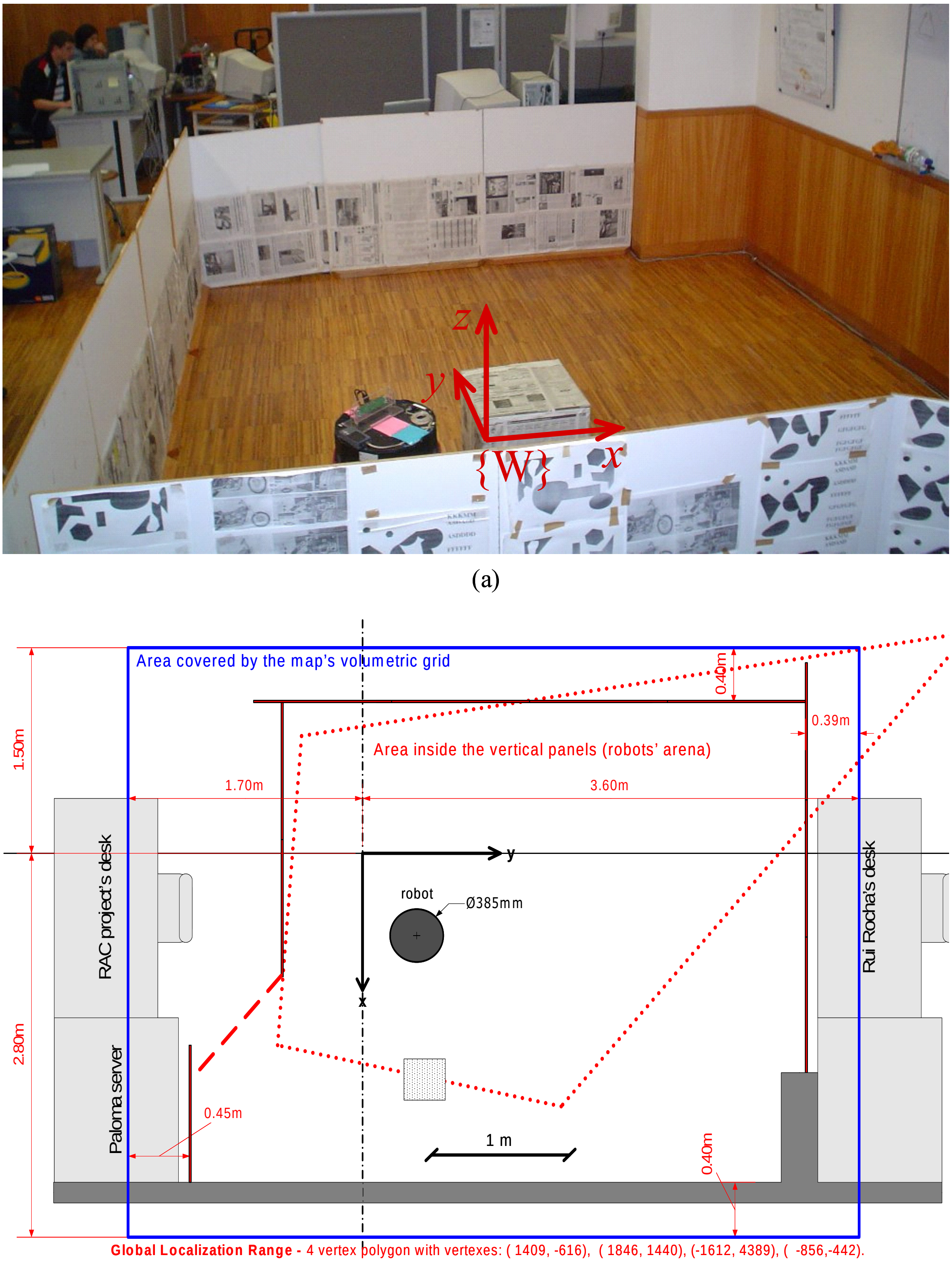
- `MAPPROVID` – *Map Provider*
  At the beginning of the mission, it connects itself as a TCP/IP client to the module `MAPCOLLECT`, which runs remotely on the host PC. Then, it sends incrementally data to it during the mission execution as long as the robot's current map is updated. The goal is to synchronize the map's copy stored in the host PC with the robot's own copy, whenever the robot updates the map upon new measurements.

- `3DMAPPING` – *3-D Mapping Main Process*
  Performs most of the robot's computation burden. When the robot's state is the state `enabled` or `paused`, it is responsible for performing sequentially a computation cycle that includes the following operations: `STEREOPROC` – acquiring a stereo image pair from the robot's stereo-vision sensor and obtaining a new batch of measurements; `MAPUPD` – updating the robot's map upon the new acquired measurements; `SURVCTRL` – using the current map and the entropy gradient exploration method to select a new viewpoint for the robot's pose, aiming at completely explore the environment; `PLATCTRL` – controlling the motion of the robot's platform, in order to move the robot's sensor from the current viewpoint to the new viewpoint, which has just been selected by `SURVCTRL`. While the robot's state is the state `paused`, the robot is able to perform all these operations but moving the platform. When in state `paused` after operation `SURVCTRL`, the robot remains stopped and waits for a transition to the state `enabled`, before starting to move to the new exploration viewpoint.

4.8 Results

This section reports experimental results obtained with one of the robots depicted in Fig. 4.10, page 124. The robot was programmed accordingly with the software architecture described in section 4.7.5, page 132, with the aim of building autonomously volumetric maps in the laboratory depicted in Fig. 4.18. The software used the volumetric mapping framework that was presented previously in this chapter.

As Fig. 4.18-b shows, the robot's workspace was roughly a rectangle having 5.3 m in width and 4.3 m in length. It was completely delimited by walls, which was either covered with wood (real walls of the laboratory) or covered with newspaper sheets (added vertical


z
y
x
{W}
Area covered by the map's volumetric grid
0.40m
0.39m
Area inside the vertical panels (robots' arena)
1.50m
1.70m
3.60m
y
RAC project's desk
Rui Rocha's desk
robot
Ø385mm
x
2.80m
Paloma server
0.45m
1 m
0.40m
Global Localization Range - 4 vertex polygon with vertexes: (1409, -616), (1846, 1440), (-1612, 4389), (-856,-442).

(a)

(b)

Figure 4.18: Workspace used in the experiments (April 2005): (a) photo showing the region of the Mobile Robotics Lab of ISR Coimbra where the experiments took place, which was delimited through vertical panels covered with newspaper sheets to imitate high-textured surfaces; (b) plan of the workspace's layout.

panels). The dotted polygon in Fig. 4.18-b, in the middle of the robot's arena, represents the area covered by the global camera, while the surrounding rectangle represents the area covered by the volumetric grid $\mathcal{Y}$ associated with the grid-based representation of the map. The value of most of the parameters used in the experiments are presented in Table B.5, page 251, wherein the exploration strategy is the uncoordinated one.

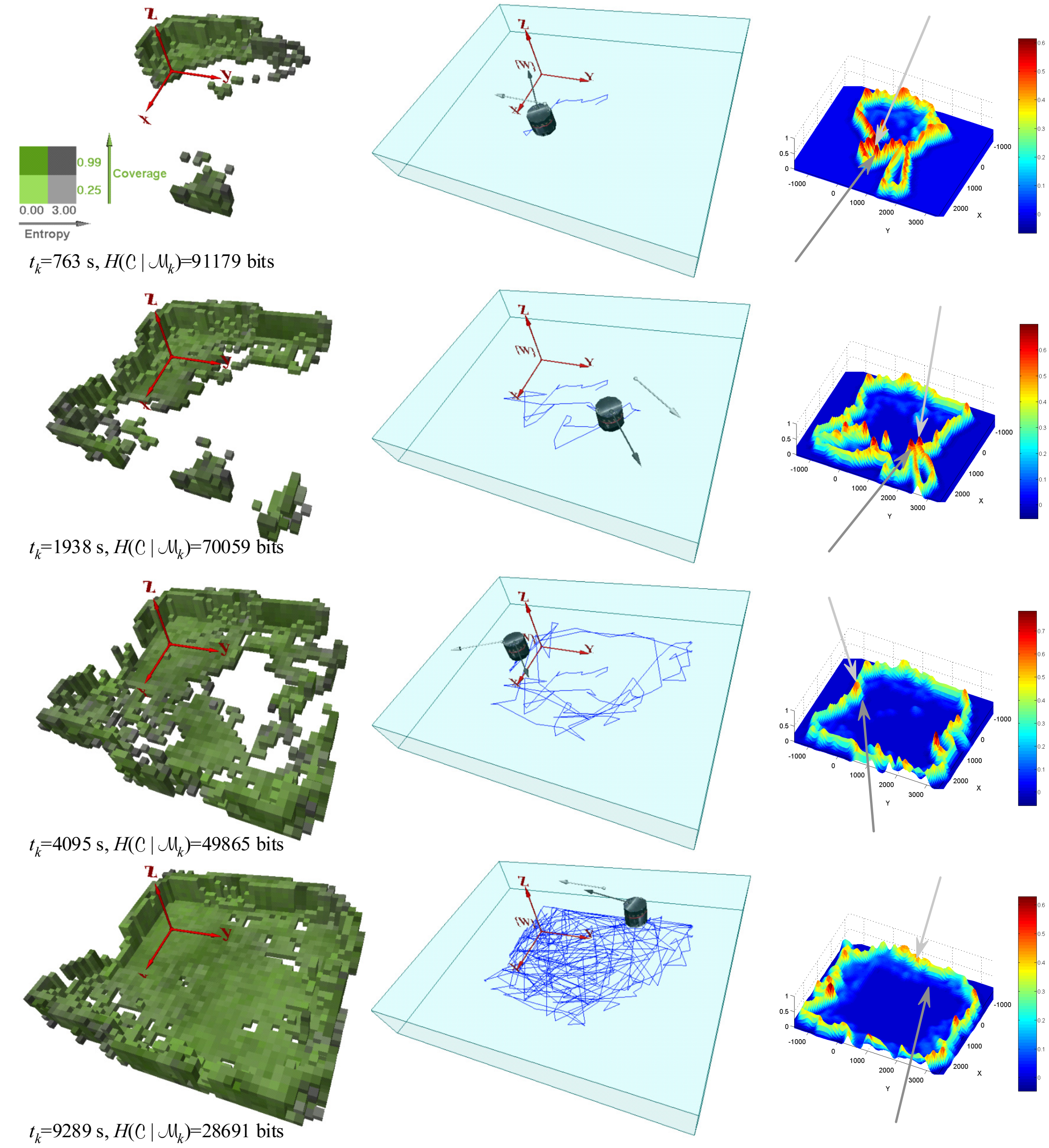
The laboratory was a highly dynamic environment with people passing by very often and much people working there. Due to their work, these researchers moved very often slightly pieces of furniture, such as chairs and tables, and sometimes modified the room's layout in order to set up new experiences and new equipment. Although this was a fortunate reality, since it demonstrated the research group's vitality, it created nevertheless a practical problem for the experiences reported herein.

As it is described in the following chapters of this thesis, the detailed map of a given volume had usually to be built several times, at different days or weeks, using one or more robots, using different programs in the robots, *etc.* Furthermore, in order to be able to compare the robots' performance and the obtained maps within different experiments, it was vital to ensure that they were performed within the *same* environment. This was accomplished by delimiting the robots' workspace with vertical panels, so as to isolate the robots' workspace from the activities taking place in the rest of the laboratory.

Moreover, since the used stereo algorithm was based on correlating patches from the two images yielded by the stereo-vision sensor (see section 4.7.2, page 125 for further details), another important practical limitation arose: the stereo-vision sensors could not measure distances when focusing on textureless areas, such as the blank vertical panels. Therefore, those "artificial" walls were covered with newspaper sheets, so as to significantly enhance the images' texture and ensure reasonable distance measurements with the stereo-visions sensors (see Fig. 4.18-a).

Fig. 4.19 shows on the left column a VRML (Virtual Reality Modeling Language) model of the obtained volumetric map for that environment at different instant times along a mission [RDC05d, RDC05a]. Each voxel is represented through a given color which depends on its coverage and its coverage belief uncertainty (entropy). The map's resolution is $\epsilon = 0.1\ m$. The robot started the mission with a maximum entropy map for which $H(0) = 11.167 \times 10^4$ bits, wherein each voxel belonging to the grid had an entropy value equal to 7 bits ($b = 128$ bins for the histogram). Then, it explored gradually the environment until the map's entropy was reduced below the threshold $H_{th} = 3 \times 10^4$ bits.

Observe on the left column of Fig. 4.19 that the map improved gradually as long as the map's entropy decreased until the final (best) volumetric map on the bottom. Fig.



t_k=763 s, $H(\mathcal{C} \mid \mathcal{M}_k)$=91179 bits

t_k=1938 s, $H(\mathcal{C} \mid \mathcal{M}_k)$=70059 bits

t_k=4095 s, $H(\mathcal{C} \mid \mathcal{M}_k)$=49865 bits

t_k=9289 s, $H(\mathcal{C} \mid \mathcal{M}_k)$=28691 bits

Figure 4.19: Volumetric map obtained with a mobile robot equipped stereo-vision. Each row represents the map at a different instant time t_k and different map's entropy level $H(\mathcal{C} \mid \mathcal{M}_k)$. The left column depicts VRML models of the 3-D map registered on the global reference frame $\{W\}$. The middle column shows the current robot's sensor pose – dark-grey arrow – and its next exploration viewpoint – light-grey arrow – within the volume being explored. The blue polyline indicates the sequence of the robot's positions since the beginning of the mission. The right column depicts graphs of the normalized entropy gradient magnitude of voxels belonging to the sensor's motion plane Γ. In the middle column, the arrows' origin indicates the robot's position and their direction indicates the robot's orientation. In the right column, the dark-grey and light-green arrows point to the current position of robot's sensor and its next exploration point, respectively. The scale of the pictures in the left and middle columns is such that each represented arrow is equivalent to a real length of 1 m.

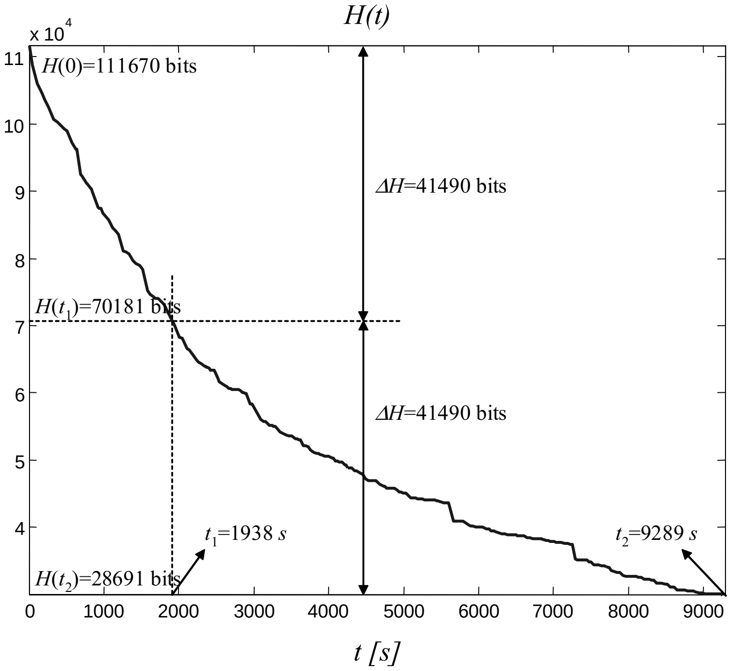


Figure 4.20: Graph of the map's entropy along a mission with one robot. A decrease of the same absolute amount of the map's entropy (41490 bits in the example) has a strong impact in the mission execution time, which, in the example, increased roughly 5 times.

4.20 shows a graph of the map's entropy along the mission and the non-linear increase of the mission execution time with a decrease of the map's entropy by the same amount.

The robot needed $t_{kmax} = 9289\ s$ to accomplish the mission without any human intervention, processed a total of 1.8×10^6 measurements, distributed along 195 batches of measurements, and had to travel a distance of 72 m. Each batch of measurements contained on average 9182 measurements.

Note that a significant part of the final map's entropy, equal to 44% of the final entropy, was due to unexplored voxels located behind the wall represented on the left of the map (the wall on the bottom of Fig. 4.18-a), which the robot would never be able to sense. Discounting this entropy bias, the final map yielded an entropy decrease of 84%, when compared with the maximum entropy initial map.

The mission execution time may seem to be exaggerated, but it is strongly influenced by the performance of the robot's embedded computer. The robot used in the experiment had indeed a quite obsolete embedded computer, which was based on a Pentium 133 MHz processor (no MMX) ([11]). Recall the software architecture depicted in Fig. 4.17, page 135. The most critical computation in the module `3DMAPPING` is updating the map upon

[11] This limitation applies to both available robots, which are shown in Fig. 4.10, page 124.

a new batch of measurements, *i.e.* the operation `MAPUPD`. On average, the robot needed 46 s to accomplish the computation cycle that included acquiring and processing a stereo image pair, updating the map, selecting the next exploration viewpoint and moving the platform to the selected target viewpoint.

The experiment was repeated with a Scout robot borrowed from another laboratory, which had an upgraded mother board with a slightly faster chipset that included a Pentium 166MHz, with MMX. This robot took 5799 s to accomplish the same mission, *i.e.* it needed only 29.7 s to execute the aforementioned computation cycle. Even better performance would be certainly achieved with an up to date processor on the robot (*e.g.* a Pentium IV processor) and, therefore, the performance yielded by the mapping framework proposed herein is better than it seems to be at first sight, when considering the absolute mission execution time with a such old processor.

The middle and right columns of Fig. 4.19 depicts the gradient entropy-based exploration criteria, which the robot used to explore gradually the environment.

The middle column represents the current robot's sensor pose, which is indicated through the dark-grey arrow, and its new selected exploration viewpoint, which is indicated through the light-grey arrow.

The right column shows a graph of the entropy gradient magnitude $\left\|\overrightarrow{\nabla} H_\Gamma(l)\right\|$, $l \in \mathcal{N}_\Gamma(\mathbf{x}, \varepsilon)$, normalized between 0 and 1, of the set of voxels $\mathcal{N}_\Gamma(\mathbf{x}, \varepsilon) \subset \mathcal{Y}$ that are intersected by the robot' sensor plane Γ. This set of voxels is the search space represented in equation (4.52), page 122. In equation (4.55), page 123, the entropy gradient is used by the robot to select a new exploration viewpoint. In each graph on the right column of Fig. 4.19, a dark-grey arrow points to the current robot's sensor position and a light-grey arrow points to the robot's sensor next exploration point. Observe that the robot usually chooses a voxel in its neighborhood having maximum (local) entropy gradient, so as to improve the information gain of new acquired sensory data.

The final map represented on the bottom-left of Fig. 4.19 is represented again in Fig. 4.21-a from different viewpoints. Fig. 4.21-b represents several viewpoints of a similar map, which was obtained using the same parameters, and within the same environment than the map of Fig. 4.21-a, but having a box within the robot's workspace. This example shows that the mapping framework proposed herein is able to cope with environments where sometimes the robots' sensors experience partial occlusions. Note that if the robot's sensor is focusing the box, it cannot sense the walls behind the box. Nevertheless, the robot is able to build gradually the map of those regions that are sometimes partially occluded, through exploring iteratively the environment from different viewpoints.

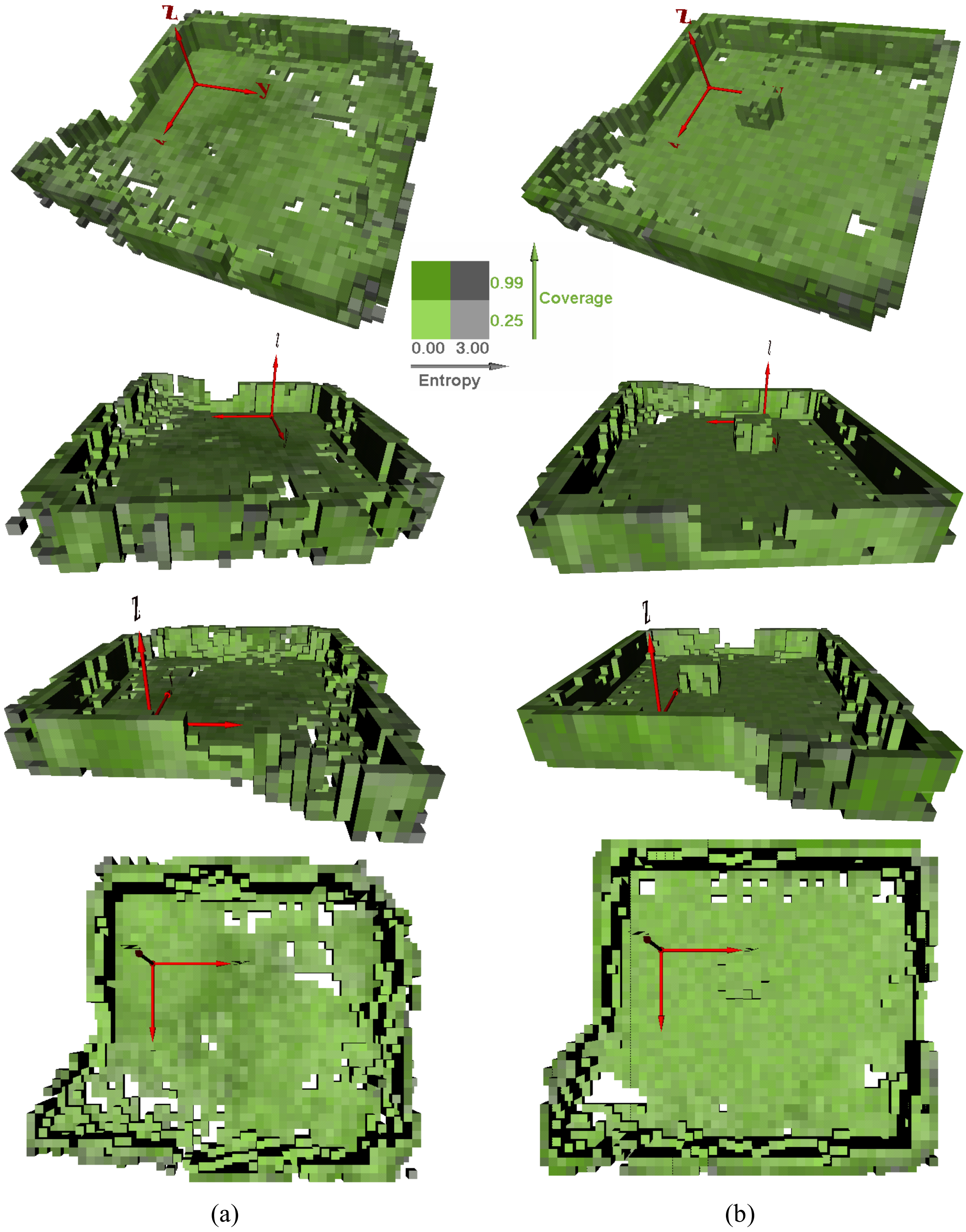


Figure 4.21: Different views of the volumetric map's VRML model at the end of two missions: (a) clean workspace; (b) workspace having a box somewhere in the middle (see Fig. 4.18-a).

4.9 Summary and discussion

This chapter proposed a grid-based probabilistic model of a volumetric map, which stores for each cell (voxel) a coverage belief and allows to model explicitly uncertainty. Entropy was used to evaluate the uncertainty associated with a voxel's belief and an entropy-based measure of the map's uncertainty was proposed. The main novelties concerning the representation model were a more compact representation of the voxel's belief than using histograms, and an efficient Bayes filter to update the map upon measurements taken at different instant times and from different locations.

A probabilistic model of a range sensor was presented, which enables to convert range measurements into voxels' coverage estimates. The range sensor model was then applied to stereo-vision range sensors, being described how the model's parameters can be calibrated.

In order to survey the environment and build the map iteratively, the frontier-based exploration concept [Yam98] was reformulated using entropy, by proposing an entropy gradient-based exploration method, whereby the robot's sensor is directed to frontier voxels between more explored and less explored regions.

After presenting details about the mobile robots, their stereo-vision sensors, the global localization scheme and the robots' software architecture for performing volumetric mapping missions, the chapter ended with the presentation of experimental results, obtained in volumetric mapping experiments with a mobile robot and stereo-vision, whose aim was to demonstrate the framework's validity.

An important assumption for the proposed representation model is to assume that the coverage of each individual voxel is independent from the other voxels' coverage. Although this might be questionable from a theoretical point of view, note that the probabilistic model's complexity would restrict too much its practical usability without the simplification. Moreover, the obtained results have demonstrated that the assumption is reasonable and not so unrealistic.

The problem of fusing sensory data in a common coordinates reference frame — the registration problem — is not addressed herein and is out of the scope of this thesis, by assuming that robots are externally localized. A global localization scheme based on a global camera covering the robots' workspace was described. Therefore, the work presented herein does not fall in the heading of registration, localization and SLAM [FBKT00, Thr01, RB02, RR04, MPS05, MR05, How05]. This thesis focus mainly on the way robots cooperate to share efficiently information and effectively coordinate their actions within mapping missions, and less on the specific research problems related with robotic mapping, which, for the scope of this thesis, is *just* the application domain and not the goal itself.

Some of those mapping problems are addressed herein; others are not. Clearly, future extensions of the research work developed within this thesis include: integrating current detailed metric maps into a more complex hierarchy of maps' representation models, containing also topological representations [KB91], so as to scale better with larger environments; and integrating more robust localization methods, such as cooperative localization schemes [MPS05, MR05, How05], which are well suited for outdoor applications where global localization schemes are not viable or are unreliable.

The former extension would take advantage from the complementary advantages of metric and topological representations, so as to be able to represent with high resolution local maps of more interesting regions, while being able to represent huge volumes through a topological representation, whose nodes are metric maps. The latter extension would take advantage from multi-robot localization methods, which have been demonstrated to be more accurate than single robot localization methods [FBKT00]. Using relative observations, different robots can refine their internal beliefs based on the other robots' estimate and improve localization accuracy.

The main research question of this thesis is not the map's representation model, but the way more than one robot, *i.e.* a team of several robots, should cooperate so as to accomplish a map in less time. This is the main subject of the next chapters — chapters 5 and 6 — which take the mapping framework presented in this chapter as a basis to develop cooperation methods for teams of mobile robots performing mapping missions. Chapter 5 proposes a distributed architecture model for a team of cooperative robots, including an information-theoretic measure of information utility, which rules the way robots share efficiently sensory data [RDC05b, RDC05a]. Chapter 6 refines the entropy gradient-based exploration method presented in this chapter with a coordination mechanism aiming at achieving more effective performance for the robotic team [RDC05c].

Chapter 5

Distributed architecture for cooperative 3-D mapping

Mobile robots have been evolving from simple mobile compounds of sensors and actuators to intelligent machines that, besides having the ability to sense the environment and act on it, have to reason about their interaction with the environment and perform autonomously complex tasks. Moreover, these tasks often require the robots' ability of coping with non structured environments and adapting to their changes. Additionally, multi-robot systems have been developed in order to take advantage from space and time distribution, though they have raised other challenging problems related with the inter-robot interaction, coordination and cooperation.

The former robots are often tele-operated from an automatic central controller, or even a human operator, which concentrates all the information, intelligence and decision power. Each robot is used as a set of resources, comprised of sensors and actuators, that is centrally controlled to perform useful tasks. Although centralized controllers are intrinsically coordinated and may lead to optimal, coherent and comprehensive solutions, they cannot cope with many robotics' requirements: they neither cannot scale up to teams of many robots nor deploy reliable robotic systems, due to the concentration of the decision power on a single machine; they require massive communication between robots and the central controllers; and, most of all, they cannot properly perform complex tasks with NP complexity ([1]), because the search space's dimension for taking optimal decisions

[1] Accomplishing a task may be viewed as performing a given algorithm to solve a specific problem. A *problem* states a relation between a set of independent input variables and the properties of the required solution, *i.e.* a set of restrictions that the solution must fulfill. An *algorithm* specifies a step-by-step procedure that allows to solve a problem. If a problem can be solved by a polynomial time algorithm, whose complexity $O(n)$ is a polynomial function $p(n)$ of the set of input variables' dimension n, it is

increases exponentially.

On the other hand, the latter robots own unavoidably on-board computation power so as to support their control autonomy and intelligence. Each robot is capable of using its computation power and on-board "intelligence" to perform autonomously useful and complex tasks in the absence of any centralized control. Sometimes this centralized control may still exist for supervising the robot's actions, or using the robot in a semi-automatic mode with some human intervention, especially in more critical systems wherein the robot's autonomy is not yet fully reliable (*e.g.* spatial exploration).

In multi-robot systems comprised of a set of such autonomous robots, the intelligence and the associated computation, and the information gathered from sensors, are inherently distributed. Therefore, in order to attain a coherent and useful global behavior for the team, a distributed control scheme is necessary instead of a centralized controller. The main advantage of distributing control and data is to easily scale up the robotic systems to an arbitrary number of robots, while maintaining the system's reliability and robustness.

The distributed control has to be able to share information among robots, so as to maintain on each robot a global and consistent representation model of the mission-relevant information, which is a result of fusing the information coming from the sensors of different robots. This global model is crucial to support the robot's local decisions with enough information about the environment, the tasks and the other robots' state, so as to attain the sufficient level of awareness required by coordinated teamwork and effective cooperation. Although there are other forms of interaction for conveying information ([2]), distributed control relies to a greater extent on interactions through explicit communication.

As communication is always limited, either in bandwidth of a communication channel or in resources applied to process the conveyed information, using efficiently these resources is crucial to scale up cooperative architectures for teams of many robots, without limiting them to simple reactive and loosely-cooperative systems, with very limited or no awareness. Current architectures extensively use explicit communication, not taking care [Par98, Jun98, Yam98, Seq99, MS01], giving low emphasis [KFO+04], or using no principled heuristics to avoid the communication of redundant information.

denoted as a *P-problem*. Conversely, a non-polynomial problem, denoted as a *NP-problem*, cannot be solved by a such polynomial time algorithm and necessarily requires an algorithm whose execution time increases exponentially with the problem dimension n. A general example of a NP-problem is one that requires a complete search over the whole set of possibilities. For instance, real-time scheduling problems are generally NP-problems [SSDNB95].

[2] See section 2.3.1, page 62, for other forms of interaction.

Although the research on communication for multi-robot systems has been mostly devoted to the *communication structure*, by addressing issues such as range and topology [Par93, Ark92, ABN93, BA94, FS94, Tam97, SV99, GM02, UA04], there is another general question, mostly related with the *communication content*, that must be addressed: "What is useful to be communicated?" or "What is task-relevant to be communicated?". The question requires in turn to answer the question "How to assess information utility?" [RDC03, RDC05b, RDC05a]. The goal behind these questions is to avoid communicating redundant information so as to use efficiently communication.

This chapter proposes a distributed architecture model for building volumetric maps with teams of cooperative robots [RDC05b, RDC05a]. After describing it, a specific case-study is used to get some insight about the information utility concept itself. In this case study, several robots are used to explore a given environment with the aim of finding regions of interest and perform work therein [RDC03]. Then, an entropy-based measure of information utility is formulated, in order to propose a cooperation scheme for sharing efficiently sensory data within a team of robots. After describing the implementation of the proposed cooperation scheme in mobile robots, the chapter ends with results obtained in cooperative volumetric mapping missions with mobile robots [RDC05b, RDC05a], including their discussion and main conclusions.

5.1 Distributed architecture model

Chapter 4 proposes a volumetric mapping framework based on the entropy concept, whereby a robot equipped with a range sensor can explore autonomously the environment and build iteratively a grid-based probabilistic map [RDC05d, RDC05a]. Now, instead of a single robot, consider a fleet $\mathcal{F} = \{1, \ldots, n\}$ of such robots performing the mapping mission. Although each robot is individually capable of performing the mission, obviously there is a need for organizing those robots in such a way whereby the space and time distribution yielded by multiple robots become an advantage through cooperation. Otherwise, instead of being an advantage, the robots' interaction would be worthless.

Figures 5.1 and 5.2 depict complementary views of a distributed architecture model for building volumetric maps with multiple cooperative robots [RDC05b, RDC05a]. Although individual robots belonging to the multi-robot system may be heterogeneous in terms of sensory skills and mobility, all of them follow that architecture model when performing the 3-D mapping mission. For this reason, both figures refer to an individual robot $i \in \mathcal{F}$. Nevertheless, the interaction with the rest of the team, *i.e.* the set of robots $\mathcal{F}\backslash i$,

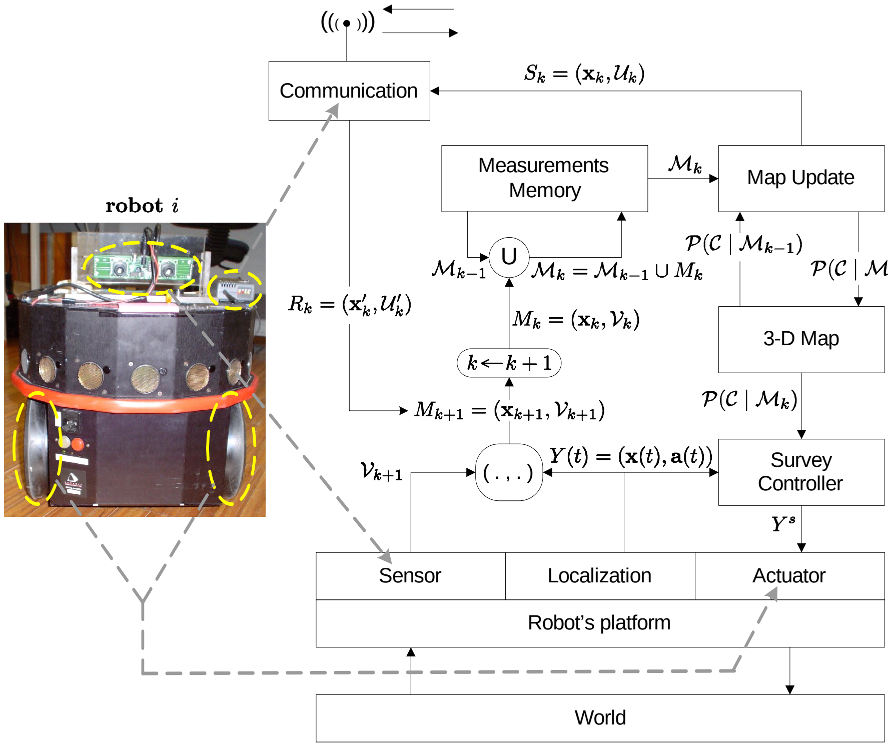

Communication
$S_k = (\mathbf{x}_k, \mathcal{U}_k)$
Measurements Memory
$\mathcal{M}_k$
Map Update
robot i
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_{k-1})$
$\mathcal{M}_{k-1}$
U
$\mathcal{M}_k = \mathcal{M}_{k-1} \cup M_k$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$R_k = (\mathbf{x}'_k, \mathcal{U}'_k)$
$M_k = (\mathbf{x}_k, \mathcal{V}_k)$
3-D Map
$k \leftarrow k+1$
$M_{k+1} = (\mathbf{x}_{k+1}, \mathcal{V}_{k+1})$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$\mathcal{V}_{k+1}$
$(.,.)$
$Y(t) = (\mathbf{x}(t), \mathbf{a}(t))$
Survey Controller
Y^s
Sensor
Localization
Actuator
Robot's platform
World

Figure 5.1: Block diagram showing the relation between different parts of the process and the resources of a given robot of the team.

is represented through the communication block and its associated data flow, which is represented in Fig. 5.2. Communication among robots is used for sharing useful range measurements among robots.

Fig. 5.1 shows the different parts of the mapping process and how they interact. The robot's platform is assumed to have a sensor, a localization module and an actuator. The sensor provides new sets of vectors $\mathcal{V}_{k+1}$ where obstacles are detected from the current sensor's pose $Y(t)$. The localization module gives the sensor's pose $Y(t)$, including position and attitude ([3]). The actuator changes the robot's sensor pose accordingly with new selected viewpoints Y^s. New data from the robot's sensor is associated with its current pose, given by the localization module, to form a new batch of measurements $M_{k+1} = (\mathbf{x}_{k+1}, \mathcal{V}_{k+1})$. Then, index k is incremented and the new batch of measurements becomes the current batch M_k. The memory of measurements is updated as $\mathcal{M}_k = \mathcal{M}_{k-1} \cup M_k$. The previous map $\mathcal{P}(\mathcal{C} \mid \mathcal{M}_{k-1})$ is updated upon the new batch of measurements M_k, which yields the current map $\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$. Robot i selects a new viewpoint $Y^s = Y^s_i$,

[3] The localization problem is not addressed herein. It is assumed that each robot is able to localize itself on a global coordinates frame through some global localization scheme.

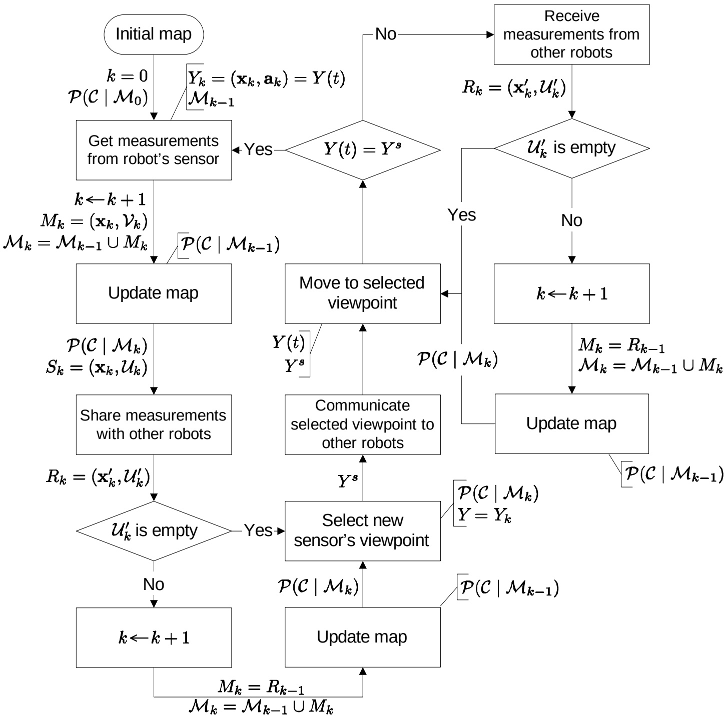

Initial map
$k = 0$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_0)$
$Y_k = (\mathbf{x}_k, \mathbf{a}_k) = Y(t)$
$\mathcal{M}_{k-1}$
No
Receive measurements from other robots
$R_k = (\mathbf{x}'_k, \mathcal{U}'_k)$
Get measurements from robot's sensor
Yes
$Y(t) = Y^s$
$\mathcal{U}'_k$ is empty
$k \leftarrow k+1$
$M_k = (\mathbf{x}_k, \mathcal{V}_k)$
$\mathcal{M}_k = \mathcal{M}_{k-1} \cup M_k$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_{k-1})$
Yes
No
Update map
Move to selected viewpoint
$k \leftarrow k+1$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$S_k = (\mathbf{x}_k, \mathcal{U}_k)$
$Y(t)$
Y^s
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$M_k = R_{k-1}$
$\mathcal{M}_k = \mathcal{M}_{k-1} \cup M_k$
Share measurements with other robots
Communicate selected viewpoint to other robots
Update map
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_{k-1})$
$R_k = (\mathbf{x}'_k, \mathcal{U}'_k)$
Y^s
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$Y = Y_k$
$\mathcal{U}'_k$ is empty
Yes
Select new sensor's viewpoint
No
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_{k-1})$
$k \leftarrow k+1$
Update map
$M_k = R_{k-1}$
$\mathcal{M}_k = \mathcal{M}_{k-1} \cup M_k$

Figure 5.2: Flowchart showing the data flow of a given robot of the team.

given the current map and its current pose $Y = Y_k$. The goal is to completely explore the environment by using the exploration method proposed in section 4.6.2, page 121, so as to attain maximum information gain in every sensing cycle. The new selected viewpoint Y^s is the reference input to the robot's actuator.

As part of map updating, a batch of measurements $S_k = (\mathbf{x}_k, \mathcal{U}_k)$ is built, which contains the most *useful* data from sensor $\mathcal{U}_k \subseteq \mathcal{V}_k$. Those selected measurements are shared between robot i and the other robots in the fleet $\mathcal{F} \backslash i$ through the communication module. This module can also provide the robot with batches of measurements $R_k = (\mathbf{x}'_k, \mathcal{U}'_k)$ given by other robots, being the map updated accordingly. Cooperation among robots arises because of this altruistic commitment to share useful sensory data.

Fig. 5.2 depicts a flowchart showing the sequence of the aforementioned robot's operations and interactions. At the beginning of the mission, an initial map is given to the robot. Then, it gets a new batch of measurements, updates the map and shares useful measurements with other robots. Then, it may receive measurements from other robots and, in that case, the map is updated accordingly. Given the new map, a new viewpoint for the sensor is selected and the robot starts moving itself. While moving, the robot continues to update the map whenever useful sensory data is received from other robots.

When the robot reaches the selected viewpoint, the process repeats itself with a new batch of measurements provided by the robot's sensor from its new pose (selected viewpoint).

Due to the information flow through the communication block represented in Fig. 5.1, the map of a given robot $i \in \mathcal{F}$ will be a result of measurements yielded by the robot's own sensor *and* also useful measurements sent by any robot $j \in \mathcal{F}\backslash i$ belonging to the rest of the team. It remains to clarify what is really a useful measurement and how to assess information utility. This issue is addressed in the following sections.

5.2 Information utility

Attaining effective cooperation within a team of intelligent robots, without restricting it to a simple reactive and loosely-cooperative multi-robot system, unavoidably requires some sort of interaction via explicit communication. Besides defining an architecture for the team's organization, it is crucial to rule the communication based on the relevance or *utility* associated to the conveyed data, so as to use efficiently communication resources and the computation power required to process communicated data.

The communicated data may be classified along *state information* and *goal-oriented information.* While the former type is aimed at increasing the robot's awareness about the state of its teammates and coordinating their activities (*e.g.* synchronizing the activities of two robots), the latter type is tailored to the application and is aimed at virtually magnifying the sensory capabilities of each individual robot.

In the mapping application domain, the goal-oriented information is range data coming from the sensors of different robots. The amount of state information and how it is used by robots depend on the specific coordination mechanism that is implemented to properly synchronize the robots' individual tasks ([4]). On the other hand, the communication of goal-oriented information, which is normally the most abundant type of information and the one that potentially consumes more communication resources, should be ruled by each emitter, *i.e.* by each robot, based on some criteria of *information utility.* In a volumetric mapping mission, instead of communicating to other robots *all* the sensory data coming from its sensor, the robot should be able to select a subset of that data that is really *useful* to be communicated to other robots.

Although goal-information is necessarily tailored to the application domain, there is a general basic principle, which is made clear in this section: *the information is as useful as it contributes to improve the team's performance with minimum cost.* In this statement,

[4] The coordination issue is carefully addressed in chapter 6, in the context of building volumetric maps.

the word *performance* is used in the sense of how close is the robotic team to accomplish its mission. Obviously, both *cost* and *performance* are specific to the application domain.

5.2.1 Case study

This section gives some insight about the information utility concept, by using a specific case-study: a consume mission. In this case study, several robots are used to explore a given environment with the aim of finding regions of interest and perform work therein [RDC03].

5.2.1.1 Specific notation

Herein it is introduced some notation that is specific to the case study [RDC03] presented in this section. It should not be confused with the general notation that is used in the rest of this thesis.

State set and Event set. The finite set $\mathcal{F} = \{1, \ldots, n\} \subset \mathbb{N}$ is the fleet of n robots forming a multi-robot system (MRS). Each robot is modeled as a discrete event system, *i.e.* an event-driven system [Cas93]. The finite discrete state set $\mathcal{X}$ contains all the possible and relevant states $x \in \mathcal{X}$ for a robot $r_i \in \mathcal{F}$, when performing a given mission.

State transitions are synchronized with the occurrence of events at discrete points in time. The time instant associated with the occurrence of the k-th event is $t_k,\ k \in \mathbb{N}$. The finite event set $\mathcal{E}$ contains all the events $e_k \in \mathcal{E}$ that are relevant to the mission execution. The triple $o_k = (e_k, t_k, r_k) \in \mathcal{O}$ denotes the occurrence of an event $e_k \in \mathcal{E}$ at $t = t_k$, detected by the robot $r_k \in \mathcal{F}$, wherein $\mathcal{O} = \mathcal{E} \times \mathbb{R} \times \mathcal{F}$ is the countable set of all possible triples.

The state of robot $r_i \in \mathcal{F}$ at $t = t_k$ is denoted by $x_i(k) \in \mathcal{X}$. The MRS's state at $t = t_k$ is the n-dimensional vector $\mathbf{x}(k) = [x_1(k), \ldots, x_n(k)]^T,\ \mathbf{x}(k) \in \mathcal{X}^n$. The instant time $t = t_0$ $(k = 0)$ denotes the initial time and $\mathbf{x}(0)$ denotes the corresponding initial state. The countable sequence $\mathbf{x}^*(k) = \{\mathbf{x}(0), \ldots, \mathbf{x}(k)\},\ \mathbf{x}^*(k) \in \mathcal{X}^{n*},$ is the MRS's state trajectory up to time $t = t_k,\ k \in \mathbb{N}_0$, being $\mathcal{X}^{n*}$ the space of countable sequences of elements in $\mathcal{X}^n$. The countable sequence

$$o^*(k) = \{o(1), \ldots, o(k)\},\ o^*(k) \in \mathcal{O}^*, \tag{5.1}$$

is the MRS's list of events up to time $t = t_k,\ k \in \mathbb{N}$, being $\mathcal{O}^*$ the space of countable sequences of elements in $\mathcal{O}$, obeying the condition $\forall_{o_i, o_j \in \mathcal{E}},\ r_i = r_j \Rightarrow t_i \neq t_j,$ which means that a robot can detect and process no more than one event at a given instant time.

System dynamics. A MRS is viewed herein as an event-driven dynamic system, wherein events are the inputs to the system and state transitions are synchronized with the occurrence of events [Cas93]. Time is included in the model so as to allow to keep track of system's performance. Since the last event occurrence, and until the next event occurrence, *i.e.* for $t_{k-1} \leq t < t_k$, the MRS's state is $\mathbf{x}(k-1) \in \mathcal{X}^n$. When an event occurs at $t = t_k$, we have a triple $o_k = (e_k, t_k, r_k) \in \mathcal{O}$ that can be used, in conjunction with the current state, to compute the new MRS's state vector, through the state transition function

$$f : \mathcal{X}^n \times \mathcal{O} \to \mathcal{X}^n. \tag{5.2}$$

Specifying a mission. A mission is viewed as a set of goals, representing physical and logical restrictions that must be fulfilled. On one hand, the cost of using physical resources required by a given mission — a fleet $\mathcal{F}$ of n robots and a communication channel — has to be considered. On the other hand, the resource concept must be generalized in order to assess performance. A *resource* is defined as any phenomenon that is physically observable. When specifying a mission, a set of such resources is defined in order to measure performance. Some examples of these resources are: space (*e.g.* coverage area), energy, time, number of robots, *etc.* Performance metrics are generally relative metrics, such as: number of tasks per robot, number of tasks per time unit, energy per robot, *etc.*

Let $\mathcal{R} = \{1, \ldots, m\}$ be a finite set of resources that are required to assess the MRS's performance when executing a given mission, and let m be the cardinality of the set $\mathcal{R}$. Let $\mathcal{H}$ denote a space of resources measuring functions of the form $h : \mathcal{X}^{n*} \times \mathcal{O}^* \to \mathbb{R},\ h \in \mathcal{H}$. Such functions measure resources given the MRS's past history, *i.e.* the state trajectory and the list of events up to current instant time.

Let $h_i \in \mathcal{H}$ be the measuring function of the resource $i \in \mathcal{R}$ and $a_i(k) = h_i(\mathbf{x}^*(k), \mathbf{o}^*(k))$ be the measure of the resource i at $t = t_k$. The m-dimensional vector of resources measures (positive real numbers) at $t = t_k$ is $\mathbf{a}(k) = [a_1(k), \ldots, a_m(k)]^T \in \mathbb{R}^m$.

The MRS's performance can be evaluated through a function $p : \mathbb{R}^m \to \mathbb{R}^+$. It is assumed that $p(\mathbf{a}(k))$ is a combination (whether linear or non-linear) of the resources measures $\mathbf{a}(k)$ and that it always computes to positive real numbers. The function is also assumed to be monotonous increasing with performance, *i.e.* higher values mean better performance. By simplicity, the quantity $p(\mathbf{a}(k))$ is abbreviated as p_k.

Given the MRS's current state $\mathbf{x}(k)$ and the current vector of resources measures $\mathbf{a}(k)$, the accomplishment of the mission is evaluated through a function

$$g : \mathcal{X}^n \times \mathbb{R}^m \rightarrow \{success, fail, ongoing\}, \tag{5.3}$$

wherein *success* means mission accomplished, $fail$ means mission failed and *ongoing* means that the mission may be still accomplished at a future instant time $t_j > t_k$. The mission is successful iff (if and only if)

$$\exists_{j \in \mathbb{N}} : \; g(\mathbf{x}(j), \mathbf{a}(j)) = success. \tag{5.4}$$

If it exists such an index j, the mission execution time is t_j.

Information utility as a balance between performance and cost. When an event $e_k \in \mathcal{E}$ occurs, we can say that it is associated with some kind of information. In a MRS, events may be classified along three classes depending on the type of conveyed information:

- *Internal events* concerning the robot's own activities and information gathered internally by the robot (*e.g.* end of an internal processing task, reaching some position, *etc.*);
- *External events* concerning changes on the mission execution environment and information obtained through the robot's sensors (*e.g.* detection of an obstacle, finding an object relevant to the mission execution, observing the movements of another robot in the team, *etc.*)
- *Received messages* concerning information provided by other team members (*e.g.* detection of an environmental condition, shared sensory data, individual state information, a negotiation bid, synchronization-related information, *etc.*).

The two former classes of events convey information about the environment (sensing, perception) and through the environment (implicit communication and stigmergy) and are not controllable. The latter class of events is concerned with information conveyed via explicit communication and are controllable. The event set $\mathcal{E}$ is thus partitioned into two disjoint subsets: the subset $\mathcal{E}_{nc}$ of *non-controllable events* and the subset $\mathcal{E}_c$ of *controllable events*. This partitioning is such that $\mathcal{E} = \mathcal{E}_{nc} \cup \mathcal{E}_c$ and $\mathcal{E}_{nc} \cap \mathcal{E}_c = \{\emptyset\}$.

The occurrence of any event may always be seen as some gain of information and is this information that enables the MRS to evolve in time through its state space $\mathcal{X}$, using the state transition function given by equation (5.2). When a non-controllable event

$o_k = (e_k, t_k, r_k),\ e_k \in \mathcal{E}_{nc}$, occurs, the robot $r_k \in \mathcal{F}$ gets some information. Then, it may decide to communicate to other robots the information that it has just acquired, by sending them a message.

When a robot decides to send a message to another robot, the former robot generates on the latter robot a controllable event $e_l \in \mathcal{E}_c,\ l > k$. The communication cost associated with controllable events is modeled through a function $en : \mathcal{E}_c \to \mathbb{R}^+$ and a communication cost per information unit $ccom$. The function en computes the entropy associated with a controllable event, *i.e.* the amount of information units (*e.g.* bits) that is communicated (see section 3.2, page 82). Given an event $e \in \mathcal{E}_c$, the associated communication cost is $en(e) \cdot ccom$.

Each state $x \in \mathcal{X}$ is assumed to have an associated constant cost per time unit, which is always positive and is given by a function $cstate : \mathcal{X} \to \mathbb{R}^+$. The mission execution cost during the time interval $t_{k-1} \le t < t_k,\ k > 0$ is given by

$$\Delta c_k = \Delta cc_k + (t_k - t_{k-1}) \sum_{i=1}^{n} cstate(x_i(k-1)), \tag{5.5}$$

wherein

$$\Delta cc_k = \begin{cases} en(e_k) \cdot ccom & ,\ e_k \in \mathcal{E}_c, \\ 0 & ,\ e_k \in \mathcal{E}_{nc}. \end{cases} . \tag{5.6}$$

The cumulative mission cost up to time $t = t_k$ is given by the recursive function

$$c_k = \begin{cases} 0 & ,\ k = 0 \\ c_{k-1} + \Delta c_k & ,\ k > 0 \end{cases} . \tag{5.7}$$

Suppose that an event $o_k = (e_k, t_k, r_k),\ o_k \in \mathcal{O}$, occurs. Let $v : \mathcal{O}^* \to \mathbb{N}_0$ be a function that computes the time index of the very last event occurrence detected by the robot $r_k \in \mathcal{F}$, given the list of events (5.1) up to time $t = t_k$. If the proposition

$$\exists_{o_w=(e_w,t_w,r_w)\in\mathcal{O}^*} :\ r_w = r_k,\ t_w < t_k, \forall_{o_m=(e_m,t_m,r_m)\in\mathcal{O}^*}, r_m = r_w,\ t_m > t_w \Rightarrow t_m = t_k$$

is true, the function v returns the index $w,\ w \in \mathbb{N}_0$, that satisfies the proposition, otherwise it returns 0, *i.e.* o_k is the first event occurrence detected by robot r_k. The event occurrence o_k is viewed as an information gain, but it has also an associated cost given by the function

$$\Delta ic : \mathcal{X}^{n*} \times \mathcal{O}^* \to \mathbb{R}^+,\ \Delta ic \in \mathcal{H}. \tag{5.8}$$

Note that this function belongs to the space $\mathcal{H}$ of resources measuring functions. The computation of $\Delta ic(\mathbf{x}(k), \mathbf{o}^*(k)) = \Delta ic_k$ depends on whether the event is non-controllable or controllable. Given the very last event occurrence $w = v(\mathbf{o}^*(k))$ before t_k,

$$\Delta ic_k = \begin{cases} (t_k - t_w) \cdot cstate(x_{r_k}(w)), & e_k \in \mathcal{E}_{nc} \\ (t_k - t_w) \cdot cstate(x_{r_k}(w)) + en(e_k) \cdot ccom, & e_k \in \mathcal{E}_c. \end{cases} \qquad (5.9)$$

Given the relative performance variation due to the event occurrence o_k, which is given by

$$\frac{\Delta p_k}{p_k} = \frac{p_k - p_{k-1}}{p_k},\ k > 0, \qquad (5.10)$$

and the relative mission cost variation due to the event occurrence o_k, which is denoted as $\Delta ic_k/c_k,\ k > 0$, the *information utility* associated with the event occurrence o_k is measured by the dimensionless ratio

$$u_k = \frac{\frac{\Delta p_k}{p_k}}{\frac{\Delta ic_k}{c_k}}. \qquad (5.11)$$

The *average information utility* during the mission execution is

$$\overline{u}_k = \frac{1}{k} \cdot \sum_{i=1}^{k} u_i,\ k > 0. \qquad (5.12)$$

Let $l(k)$ be the number of controllable events up to time $t = t_k$. The influence of explicitly communicated information on the average information utility can be assessed by the quantity

$$\overline{uc}_k = \begin{cases} 0,\ l(k) = 0\ \vee\ \overline{u}_k = 0, \\ \frac{\sum_{i=1}^{k} \left\{ \begin{array}{l} u_k,\ e_k \in \mathcal{E}_c \\ 0,\ e_k \in \mathcal{E}_{nc} \end{array} \right\}}{l(k) \cdot \overline{u}_k},\ otherwise. \end{cases}, \qquad (5.13)$$

which is denoted as *communication utility.* This quantity measures the average communicated information utility relative to the average information utility including all event occurrences (all types of information).

If the mission is successful, *i.e.* there is a time t_j that verifies equation (5.4), the mission's information utility is $\overline{u}_j$, which can be used to compare two different systems performing the same mission, or to compare the performance of a given system when performing different missions. For that same mission, the communication utility is $\overline{uc}_j$.

5.2.1.2 Specification of the consume mission

In order to apply the aforementioned definition of information utility to an example, consider a *consume mission* whose definition is somewhat similar to the consume task

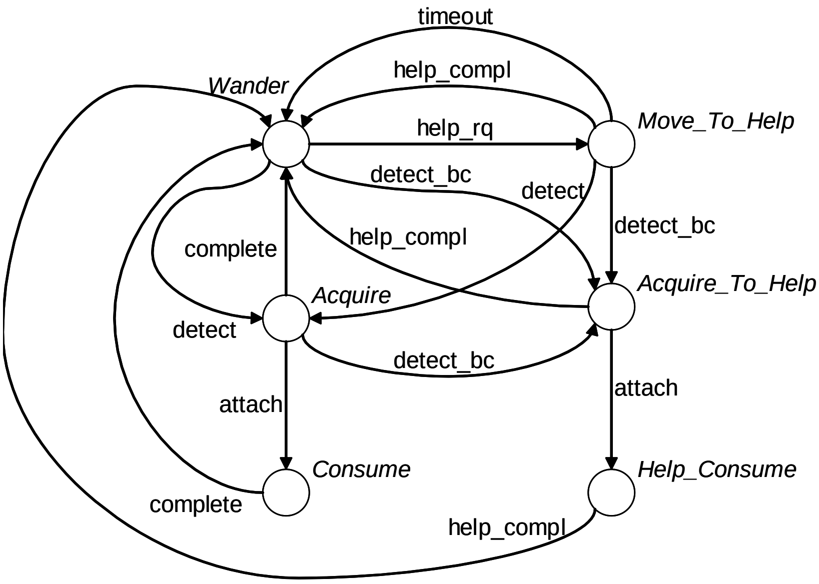


Figure 5.3: State transition graph of a robot performing the consume mission (common to all robots).

described in [BA94]. The state space $\mathcal{X}$ for this mission contains states `Wander`, `Acquire`, `Consume`, `Move_To_Help`, `Acquire_To_Help` and `Help_Consume`. The event set $\mathcal{E}$ contains events `detect`, `attach`, `complete`, `detect_bc`, `help_rq`, `help_compl` and `timeout`. The state transition graph for a robot is depicted in Fig. 5.3. This graph can be used to compute the state transition function (5.2) for this mission [RDC03].

The mission of a team of n homogeneous and non-holonomic robots (differential drive robots) is to wander about a 2-D environment, looking for static items of interest. Once a robot encounters one of these items (event `detect`), it acquires the item (state `Acquire`), attaches itself to the item (event `attach`) and consumes it (state `Consume`), *i.e.* performs some work on it. The required time to consume the item is t_c.

When a robot encounters a new item, it can request help for other robots (event `help_rq`), in order to reduce the item's consume time to t_c/n_c^2, wherein n_c is the number of robots consuming an item. The event `detect_bc` models the situation in which a wandering robot finds an item that is already being consumed by other teammate or teammates. When a robot is moving to help another robot (state `Move_To_Help`), if it could not reach the item within a predefined time, a `timeout` event occurs and the robot changes to state `Wander`.

The 2-D environment is populated with static obstacles, which must be avoided by the robots. Robots must also avoid colliding each other. The goal of the mission is to

Table 5.1: Measured resources for assessing performance in the *consume* mission.

$i \in \mathcal{R}$	$h \in \mathcal{H}$	**Description**
1	h_1	Elapsed time since the beginning of the mission.
2	h_2	Number of detected items (events `detect`, `detect_bc` and `help_rq`) for every robots.
3	h_3	Number of consumed items (events `complete`) for every robots).
4	h_4	Average time spent in state `Consume` for every robots.

find and consume b items before $t = t_{max}$.

The unique controllable event is `help_rq`. It is assumed that this event models the reception of a message with the 2-D coordinates of a requesting help robot. Thus, its associated entropy is $en($`help_rq`$) = 32$ bits if we assume that each coordinate is coded through a 16 bits binary code.

The number of resources for this mission is $m = 4$. Table 5.1 presents the meaning of the resources measuring functions used to assess the mission performance. The vector of resources measures at $t = t_k$ is

$$\begin{aligned} \mathbf{a}(k) &= [a_1(k),\ a_2(k),\ a_3(k),\ a_4(k)]^T \\ &= [h_{1k},\ h_{2k},\ h_{3k},\ h_{4k}]^T, \end{aligned} \tag{5.14}$$

wherein $h_{ik} = h_i(\mathbf{x}^*(k), \mathbf{o}^*(k)),\ i \in \mathcal{R}$.

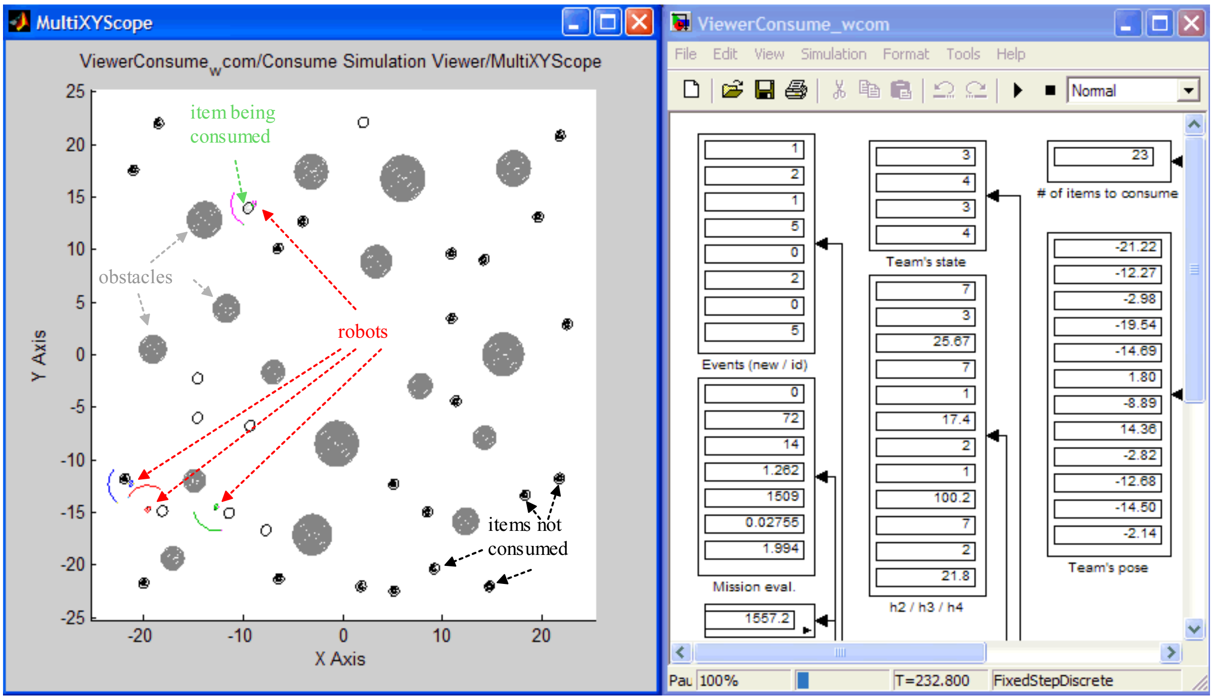
The mission accomplishment function $g_k = g(\mathbf{x}(k), \mathbf{a}(k))$ is defined as

$$g_k = \begin{cases} success & ,\ a_3(k) = b \wedge a_1(k) \leq t_{max}, \\ ongoing & ,\ a_3(k) < b \wedge a_1(k) < t_{max}, \\ fail & ,\ a_3(k) < b \wedge a_1(k) \geq t_{max}. \end{cases} \tag{5.15}$$

The performance function $p_k = p(\mathbf{a}(k))$ is defined as

$$p_k = \begin{cases} \alpha & ,\ k = 0, \\ \alpha + \beta \frac{a_2(k)}{a_1(k)} + \gamma \frac{a_3(k)}{b} + \delta \frac{t_c}{a_4(k)} & ,\ k > 0, \end{cases} \tag{5.16}$$

wherein $\alpha,\ \beta,\ \gamma$ and δ are weights, such as $\alpha, \beta, \gamma, \delta > 0$, $\alpha \ll \beta$, $\alpha \ll \gamma$ and $\alpha \ll \delta$. The constant α ensures that the performance function always computes to positive real numbers.



Normal

Events (new / id): 1, 2, 1, 5, 0, 2, 0, 5

Mission eval.: 0, 72, 14, 1.262, 1509, 0.02755, 1.994

1557.2

Team's state: 3, 4, 3, 4

h2 / h3 / h4: 7, 3, 25.67, 7, 1, 17.4, 2, 1, 100.2, 7, 2, 21.8

# of items to consume: 23

Team's pose: -21.22, -12.27, -2.98, -19.54, -14.69, 1.80, -8.89, 14.36, -2.82, -12.68, -14.50, -2.14

Pau 100% T=232.800 FixedStepDiscrete

Figure 5.4: Simulator of a consume mission with four mobile robots running in Matlab©.

5.2.1.3 Simulation results

The performance of a team of robots executing a consume mission was simulated in Matlab© (see Fig. 5.4) with the aim of getting some insight about the information utility concept [RDC03]. The robots' behavior along their possible states were modeled through potential field techniques and following an approach very similar to the one that is described in [BA94].

Five basic behaviors were implemented: `noise`, `avoid_obstacles`, `avoid_robots`, `move_to_goal` and `consume`. The robots' behavior for each state was a linear combination of those basic behaviors, with predefined weights. For example, in state `Wander`, the active basic behaviors were `noise`, `avoid_obstacles` and `avoid_robots`. The basic behavior `noise` was active for all robot's states in order that local minima and maxima were avoided.

Table E.1, page 271, presents the main parameters used in the experiments, whose values were selected empirically. The maximum mission time was $t_{max} = 11000\ s$. Several experiments were carried out along different combinations of three variables: *team size*, *obstacles' coverage* and *communication range*.

Fig. 5.5 shows the simulation results through some representative graphs [RDC03].

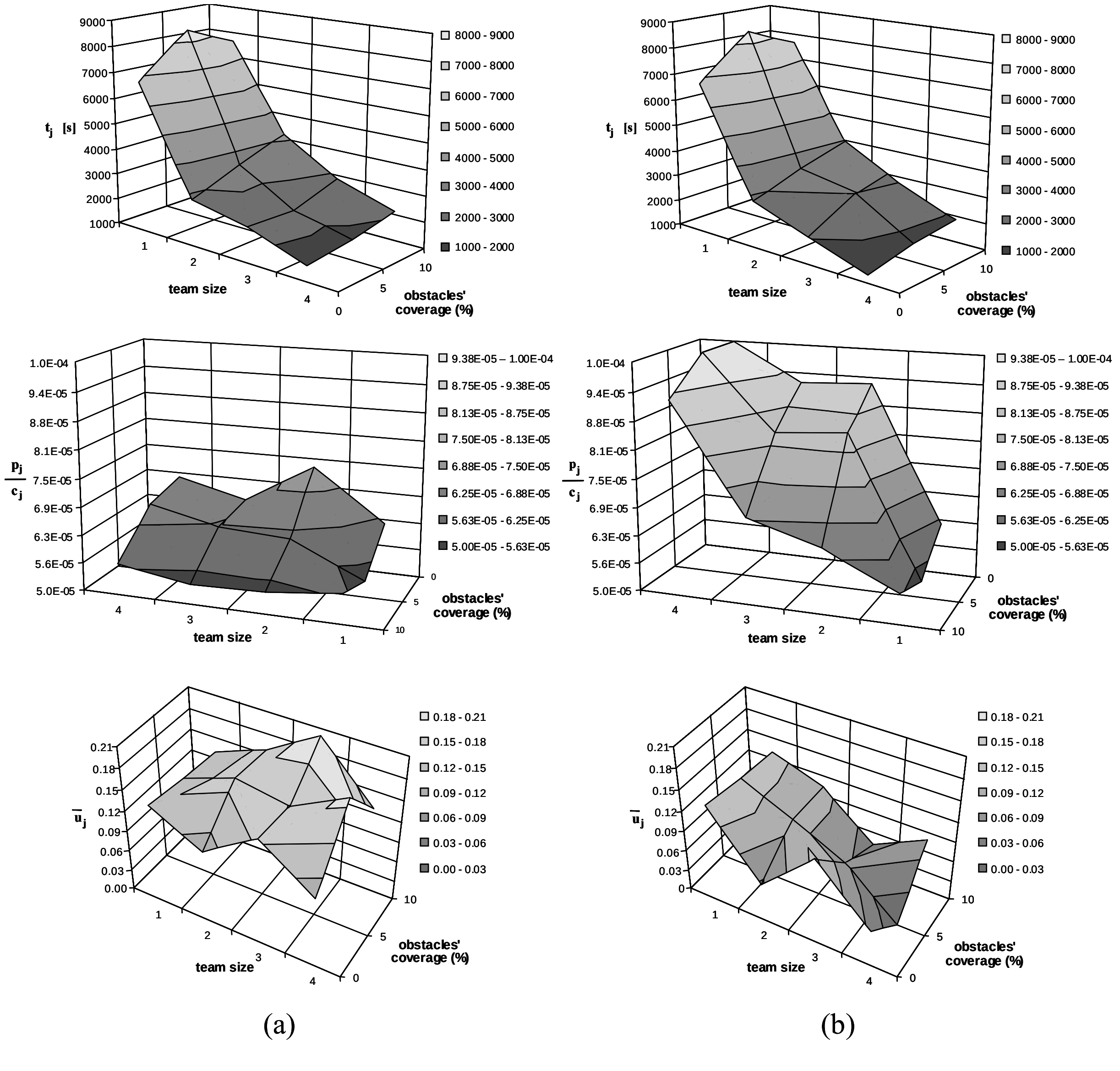
The graphs on the left show the simulation results with the robots' communication range configured to 0 m, *i.e.* with event `help_rq` disabled. In this situation, a robot can never reach the state `Move_To_Help`. Conversely, the graphs on the right show the simulation results with robots' communication range configured to 20 m, *i.e.* with event `help_rq` enabled. The multi-robot systems always accomplished the mission with success within the experiments that yielded the results shown in Fig. 5.5, *i.e.* they always consumed all the 30 items before $t_{max} = 11000\ s$.

Observing the graphs of the mission execution time t_j, we can see that its reduction due to team size increasing or due to obstacles' coverage decreasing, is more significant when we go from the single robot case to teams of two robots, than we go from two robots to three or four robots. This means that the benefit of increasing redundancy is significant for small teams and less important for more populated teams. The reduction in t_j is also more significant for teams that are allowed to use explicit communication. Obviously, obstacles' coverage influences the mission execution time, because obstacles must be avoided by the robots, thus conditioning their progression along the workspace.

Observing both graphs of the ratio p_j/c_j, *i.e.* the performance versus cost ratio at the end of the mission, we can identify a local maxima in the same point (a team of two robots and 0% of obstacles' coverage), whether robots are allowed to communicate or not. It is also noticeable a very significant influence of explicit communication in the ratio p_j/c_j, denoting its utility for this mission. On average, communication improved p_j/c_j by 35%. For a team of four robots working in an environment with 10% of obstacles' coverage, that increase was much higher (about 94%). This increase was generally higher for combinations of teams with more robots and lower obstacles' coverage values.

Comparing the graphs of p_j/c_j with the corresponding graphs of information utility $\overline{u}_j$, we can observe a correlation between the two measures, which means that optimizing the balance between performance and cost seems to be equivalent to optimize the information utility. That correlation is particularly noticeable in the absence of communication. For example, note a local maxima in the graph of $\overline{u}_j$ without communication, exactly in the same point where the ratio p_j/c_j has also a maxima.

Observing now the graph of communication utility, which is shown in Fig. 5.6, we can observe that this measure has also a strong correlation with the ratio p_j/c_j, though that correlation is not so evident for the information utility measure.



(a) (b)

Figure 5.5: Simulation results of the consume mission case study: (a) without explicit communication; (b) with the communication range configured to 20 m.

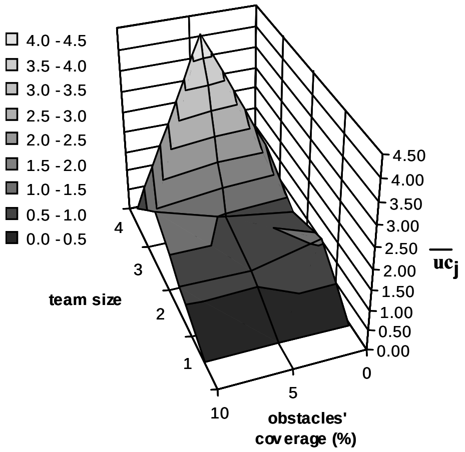


Figure 5.6: Graph of communication utility for the consume mission case study.

5.3 Using entropy to assess information utility

The previous section defined information utility as a balance between performance improvement and cost increasing. After illustrating the concept within a case study (see section 5.2.1.2, page 155), this section applies it to the context of building volumetric maps with a team of mobile robots, by using the formal measure of information — entropy — and the mapping framework introduced in chapter 4.

In a mapping mission with a team of mobile robots, the most obvious measures of performance are the map's uncertainty and the mission execution time. We can say, in a rather informal way, that achieving a good team's performance means not to spend much time to reduce the map's uncertainty to a given pre-specified lower bound entropy level.

The map's uncertainty can be formally measured through the map's entropy $H(t_k)$, given by equation (4.9), page 105. Given a desired map's entropy lower bound H_{th}, the aforementioned informal specification of a mapping mission can be formally written through equation (4.10), page 106, which states that the mission execution time $t_{k_{max}}$ is the first instant time for which the condition $H(t_k) < H_{th}$ is verified. Thus, the team's goal should be to minimize the time $t_{k_{max}}$ required to attain the entropy threshold H_{th}.

Whenever a robot obtains a new batch of measurements M_k, we can say that this event has an associated information utility, since it contributes to reduce the map's joint entropy $H(t_k)$ by a certain value [RDC05b, RDC05a]. Besides this positive contribute to the team's performance, that batch of measurements also contributes to cost, since it uses several resources (*e.g.* time, robot's mobility, computation power, *etc.*).

The most abundant type of information that a robot may share with its teammates is range data coming from its sensor. An important question that a robot has to answer, after obtaining a new batch M_k containing m_k different range measurements, is what measurements from that batch are *really* useful to be communicated to its teammates. This answer obviously depends on the robot's ability to assess the information utility of each measurement, by balancing performance improvement and cost increasing.

5.3.1 Information utility associated with a range measurement

In order to compute the information utility associated with range measurements, it is sufficient to compare the map's entropy decrease yielded by each measurement of a batch M_k and compare it with other measurements contained in the same batch, *i.e.* to compare the performance variation due to each measurement. This is because the cost associated with each measurement of a batch is virtually the same. It is the same because, for example, the time needed to process each measurement and integrate it in the map is roughly the same. Although this time is not exactly the same, because, accordingly with equation (4.17), page 108, long distances require some more computation power, since they influence a greater number of voxels, the constant cost assumption is nevertheless quite reasonable.

Therefore, the decrease of the map's joint entropy $H(t_k)$ within a period of time is a measure of the information utility of the measurements gathered within the same period of time, in terms of their utility on improving the map's accuracy [RDC05b, RDC05a]. This period of time is the time required by a robot to move to a new exploration viewpoint, measure the environment from that point and integrate measurements in the map.

Consider a batch of measurements $M_k = (\mathbf{x}_k, \mathcal{V}_k)$ yielded by the robot's sensor. Each measurement $\overrightarrow{\mathbf{v}}_{k,i} \in \mathcal{V}_k$ influences the coverage of the set of voxels $\mathcal{Z}_{k,i}$ and has an associated information utility. Let $l \in \mathcal{Z}_{k,i}$ be a voxel whose coverage C_l is influenced by the new measurement $\overrightarrow{\mathbf{v}}_{k,i}$; for the same voxel, let also $p(c_l \mid \mathcal{D}^l_{n-1}) = p(c_l \mid D^l_1, \ldots, D^l_{n-1})$ and $p(c_l \mid \mathcal{D}^l_n) = p(c_l \mid D^l_1, \ldots, D^l_n)$ be the coverage belief, respectively, before and after voxel l is updated with the new influencing measurement $D^l_n = (d_n, d^l_n)$, through equation (4.37), page 116.

Using the conditional mutual information definition given by equation (3.28), page 91, the *information utility* associated with a measurement $\overrightarrow{\mathbf{v}}_{k,i} \in \mathcal{V}_k$ is

$$\begin{aligned} I_{k,i} &= \sum_{l \in \mathcal{Z}_{k,i}} H(C_l \mid D_1^l, \ldots, D_{n-1}^l) - H(C_l \mid D_1^l, \ldots, D_n^l) \\ &= \sum_{l \in \mathcal{Z}_{k,i}} I_{k,i}^l. \end{aligned} \tag{5.17}$$

Each term $I_{k,i}^l$ in equation (5.17) measures the mutual information between $p(c_l \mid D_n^l)$ and $p(c_l)$, conditioned to the past history $\mathcal{D}_{n-1}^l$, *i.e.* the contribution of each influencing measurement D_n^l to reduce the voxel's uncertainty [RDC05b, RDC05a]. For instance, the information utility of the influencing measurement depicted in Fig. 4.8, page 118, is equal to 0.321.

The information gain due to the k-th batch of measurements is given by

$$I_k = \sum_{i=1}^{m_k} I_{k,i} = H(t_{k-1}) - H(t_k), \tag{5.18}$$

which measures the mutual information between the current map and the new acquired batch of measurements, *i.e.* the contribution of batch M_k to reduce the map's uncertainty [RDC05b, RDC05a].

Since equation (4.9), page 105, requires the computation of the entropy function for *every* voxels $l \in \mathcal{Y}$, it represents a time-consuming computation if it is used at each time step, whenever a new batch of measurements is gathered. But, as equation (5.18) suggests, the map's joint entropy can be recursively updated as $H(t_k) = H(t_{k-1}) - I_k$, which is a much more efficient computation procedure because it is only computed the equation (5.17) for each measurement $\overrightarrow{\mathbf{v}}_{k,i} \in \mathcal{V}_k$ belonging to the batch $M_k = (\mathbf{x}_k, \mathcal{V}_k)$. Thus, equation (4.9), page 105, is only required for computing the maps's initial entropy $H(0) = H(\mathcal{C} \mid \mathcal{M}_0)$.

5.3.1.1 Efficient computation using differential entropy

Recall that, accordingly with the framework proposed in chapter 4, the coverage belief $p(c_l \mid \mathcal{M}_k) = p(c_l \mid \mathcal{D}_k)$ is modeled through a Gaussian probability density function (pdf). Recall also that discrete entropy is used as an absolute measure of the voxel's uncertainty, through equation (4.8), page 104, which computes the voxel's entropy upon a quantized version of the voxel's pdf with b bins.

Although differential entropy cannot be used as an absolute measure of entropy, it is a valid relative measure of entropy, *i.e.* both discrete entropy and differential entropy can be used to compute the variation (difference) between two entropy values, and this difference is equal for both. Equation (3.39), page 94, provides a very convenient procedure for computing the differential entropy of a Gaussian, because it is a closed-form equation [RDC05b, RDC05a]. Instead, computing the discrete entropy through equation (4.8), page 104, is computationally heavier.

For each influenced voxel $l \in \mathcal{Z}_{k,i}$ in equation (5.17), let the Gaussians $p(c_l)$ and $p'(c_l)$, having standard deviation σ_l and σ'_l, be its coverage belief before and after the new measurement is integrated, having differential entropy $h(l)$ and $h'(l)$ and discrete entropy $H(l)$ and $H'(l)$, respectively. Let $I^{l\delta}_{k,i} = h(l) - h'(l)$ be the measurement's information utility for that voxels computed through equation (3.39), page 94, and $I^{l\triangle}_{k,i} = H(l) - H'(l)$ the same utility computed upon the pdf's quantized version using discrete entropy. It can be easily shown that [RDC05a]

$$I^{l}_{k,i} \approx I^{l\delta}_{k,i} = \log\left(\frac{\sigma_l}{\sigma'_l}\right). \tag{5.19}$$

Equation (5.19) states that $I^{l}_{k,i}$ is approximately equal for Gaussian pairs with the same standard deviation ratio σ_l/σ'_l.

Fig. 5.7 compares $I^{l\triangle}_{k,i}$ and $I^{l\delta}_{k,i}$ as a function of σ_l, when $\sigma_l/\sigma'_l = 1/0.99$, *i.e.* when the new measurement yields a decrease of 1% in the standard deviation of the voxel's Gaussian coverage. It shows that: for $\sigma_l > 0.3$, the differential entropy-based estimate cannot be used because it neglects the pdf's truncation to the interval $0 \leq C_l \leq 1$ and the normalization introduced by equation (4.31), page 112; and, for $\sigma_l < 3 \times 10^{-3}$, the discrete entropy-based estimate cannot be used, because the histogram's bins have not sufficient resolution to model pdf's with smaller standard deviation. However, for $3 \times 10^{-3} \leq \sigma_l \leq 0.3$, which encompasses most of the situations with range sensors, like the stereo-vision sensors shown in Fig. 4.11, page 126, both estimates are approximately equivalent.

This conclusion is roughly the same for other values of the ratio σ_l/σ'_l, though it is not explicitly shown herein with other graphs similar to the graph shown in Fig. 5.7. Therefore, equation (5.19) is generally used for computing efficiently the information utility of a measurement, which is given by equation (5.17).

5.4 Cooperation through sharing useful sensory information

The distributed architecture model shown in Fig. 5.1, page 148, includes a communication module, which is used by each robot to cooperate with its teammates by sharing *useful*

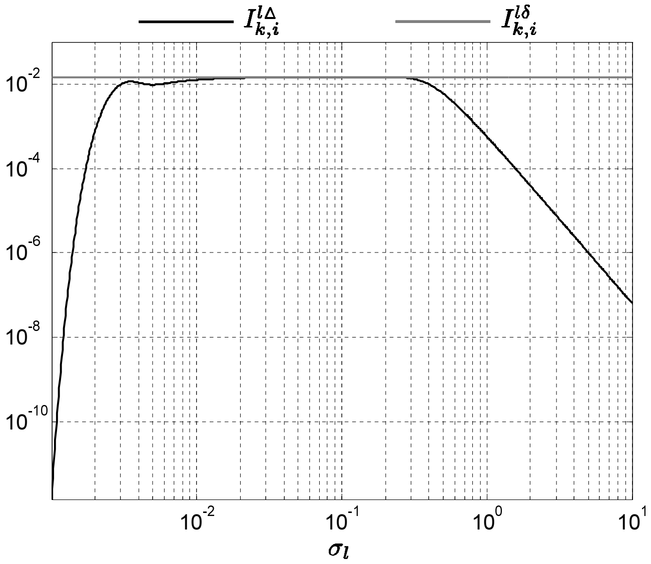


Figure 5.7: Computing information utility $I^l_{k,i}$ of a measurement for a given influenced voxel l, using either discrete or differential entropy. Being σ_l and $\sigma'_l = 0.99\sigma_l$ the standard deviation of the voxel's coverage Gaussian before and after the measurement, respectively, the graph plots the computed values of $I^l_{k,i}$ as a function of σ_l, using either the histogram's discrete entropy $I^{l\Delta}_{k,i}$ (histogram with $b = 128$ bins) or the approximated voxel's coverage differential entropy $I^{l\delta}_{k,i} = 1.45 \times 10^{-2}$.

sensory data. Section 2.1.2, page 23, and section 2.1.3, page 25, provide some insight about group behavior in biological systems and human societies, wherein reciprocal altruism is a means to effectively barter resources and information and thus attain good overall performance.

This section presents a cooperation strategy based on reciprocal altruism, whereby a team of mobile robots, populating a 3-D environment being mapped, can be cooperative on building a volumetric map through sharing useful measurements. Accordingly with that architecture model, each robot is able to build and update its own local 3-D map alone, based on range data coming from its own sensor, but it is *also* committed to share new acquired sensory information with its teammates via explicit communication. Due to this altruistic commitment, which applies to every robots in the team, each robot may also receive *useful* sensory data from its teammates [RDC05b, RDC05a].

Whenever a given robot gets a batch of measurements $M_k = (\mathbf{x}_k, \mathcal{V}_k)$, it is committed to send to other robots a subset of *useful measurements* $S_k = (\mathbf{x}_k, \mathcal{U}_k)$, which follows a criteria based on the measurements' information utility computed through equation (5.17). The set

$$\mathcal{U}_k = \{\overrightarrow{\mathbf{u}}_{k,1}, \ldots, \overrightarrow{\mathbf{u}}_{k,s_k}\} \subseteq \mathcal{V}_k \tag{5.20}$$

contains s_k measurements selected to be communicated. The sensor's position $\mathbf{x}_k$ from where those measurements were gathered is also sent, since it is required for registering those measurements in the local map of other robots ([5]) [RDC05b, RDC05a].

Different communication topologies may be used, depending on the capacity and range of the available communication channel. Whenever possible, the robot acting as information provider (the emitter robot) should send data to all robots in the team, so that all of them can take advantage of new sensory information; otherwise, the communication is restricted to a team's subset, such as the nearest robots to the information provider.

When a robot receives a batch of u_k communicated measurements $R_k = (\mathbf{x}'_k, \mathcal{U}'_k)$, it updates its local map as if measurements $\mathcal{U}'_k$ would have been gathered by its own sensor when located at position $\mathbf{x}'_k$.

As communication channels have always limited capacity, when a robot is acting as an information provider, it has to limit the amount of communicated data and select the most useful measurements gathered from its own sensor. On doing it, the robot uses equation (5.17) to assess the information utility of measurements $\overrightarrow{\mathbf{v}}_{k,i} \in \mathcal{V}_k$ and classifies them by utility.

Let define $s_{k_{max}}$ as being the maximum number of allowable communicated measurements at a given time instant. Let also define I_{min} as being the minimum allowable information utility for a communicated measurement. The set (5.20) is built in such a way that the proposition

$$\begin{aligned}
&(s_k \leq s_{k_{max}} \wedge \\
&s_k < s_{k_{max}} \Rightarrow \forall_{\overrightarrow{\mathbf{v}}_{k,z} \in \mathcal{V}_k \backslash \mathcal{U}_k},\ I_{k,z} < I_{min} \wedge \\
&\forall_{\overrightarrow{\mathbf{u}}_{k,j} \in \mathcal{U}_k},\ I_{k,j} \geq I_{min} \wedge \forall_{\overrightarrow{\mathbf{v}}_{k,w} \in \mathcal{V}_k \backslash \mathcal{U}_k},\ I_{k,w} \leq I_{k,j})
\end{aligned} \tag{5.21}$$

is true [RDC05b, RDC05a]. This proposition is true, *i.e.* the set of communicated measurements is valid, if the following conditions are met:

- The size of the set is not greater than $s_{k_{max}}$;
- If the size of the set is less than $s_{k_{max}}$, it includes necessarily all measurements in the set $\mathcal{V}_k$ having an information utility not less than I_{min};
- The information utility of any communicated measurement is at least equal to I_{min};
- Every non communicated measurements have lower or equal utility than those that are selected to be communicated.

[5] As it was referred earlier, it is assumed that each robot is able to localize itself and correctly register measurements coming from its range sensor.

The parameter $s_{k_{max}}$ restricts the bandwidth used by the robot to send sensory data to its teammates, whereas the parameter I_{min} ensures that communicated measurements yield a minimum contribution to reduce the map's uncertainty.

If the robot senses already explored regions due, for instance, to some non-optimal exploration algorithm that does not prevent that situation, most of the measurements will have intrinsically lower information utility than when the robot explores regions for the first time. In the absence of the strategy proposed herein, the robot could send those rather redundant measurements to other robots, thus using resources (*e.g.* communication bandwidth or computation power) that could be used to process more useful information. This disadvantage may not seem obvious, but the key point is that processing a redundant measurement, which has a lower utility, consumes as much resources as a highly useful measurement, *i.e.* the required processing time is the same in both cases. Therefore, processing redundant data would lead definitely to losses of efficiency and performance.

Accordingly with equation (5.17), the sender robot assesses the measurement's utility by assuming that if the measurement is useful for itself it is equally useful for its teammates. Although different robots may have different maps, as we shall see in section 5.6, typically the robots' maps are just slightly different and, thus, the assumption remains valid.

5.5 Implementation in mobile robots

With the aim of implementing in mobile robots the aforementioned cooperation strategy for sharing useful sensory information, the mobile robots described in section 4.7, page 123, were used to carry out some experiments. Recall that those robots are differential drive robots equipped with stereo-vision, sonars and wireless communication (see Fig. 4.10, page 124).

The software architecture represented in Fig. 4.17 (see section 4.7.5, page 132, for details) was further extended accordingly with Fig. 5.8, in order to implement the distributed architecture model shown in Fig. 5.1, page 148. It is intrinsically scalable to a team having an arbitrary number of robots. This version has two additional modules:

- `MEASPROVID` – *Measurements Provider*
  A TCP/IP client that sends to other robots useful measurements, whenever the map is updated through the module `MAPUPD`, after a new batch of measurements has been provided by the robot's sensor.

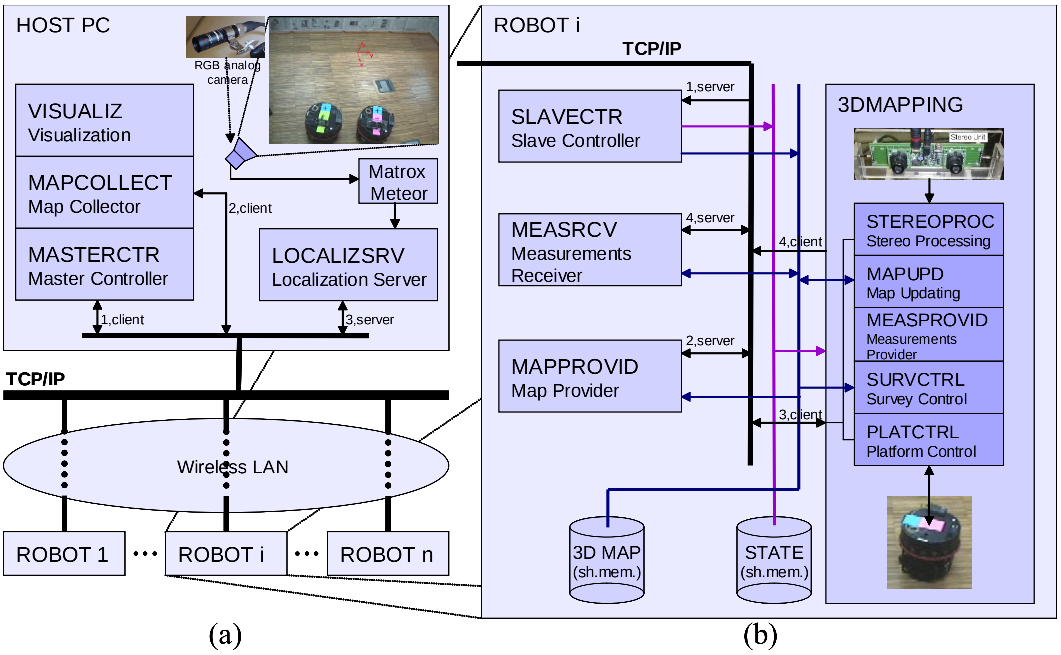


Figure 5.8: Diagram of the software architecture with information sharing: (a) interaction between the robot and the host PC used for supervising 3-D mapping missions and providing global localization through a global camera; (b) software modules running locally on the robot. This is a revised version of the diagram represented in Fig. 4.17, page 135.

- `MEASRCV` – *Measurements Receiver*
  A TCP/IP server that receives batches of useful measurements sent by module `MEASPROVID` of other robots; the robot's map is then updated accordingly, as if those batches have been provided by the robot's own sensor.

5.6 Results and discussion

This section presents results obtained within experiments carried out with the two cooperative mobile robots shown in Fig. 4.10, page 124, which were programmed accordingly with the software architecture shown in Fig. 5.8. These experiments aimed at studying the relation between the information sharing parameters I_{min} and $s_{k_{max}}$ and the team's performance, by comparing the mission execution time $t_{k_{max}}$ with different values for those parameters [RDC05b, RDC05a].

As in the experiments reported in section 4.8, page 136, the robots started each experiment with a maximum entropy map and used the entropy gradient-based method described in section 4.6.2, page 121, for exploring the environment until a given entropy threshold was attained. The map's resolution was $\epsilon = 0.1\ m$. The initial map had an entropy $H(0) = 12.936 \times 10^4$ bits, wherein each voxel belonging to the grid had an entropy value equal to 7 bits ([6]). The stopping criteria, which is formally stated through equation (4.10), page 106, assumed $H_{th} = 2.65 \times 10^3$ bits ([7]). Nevertheless, the robots' workspace was slightly different from the one used in the experiments reported in section 4.8, mainly in width. Moreover, the represented maximum height in the map was higher (1 m against 0.7 m).

Fig. 5.9 shows a photo of it and a plan of the room's layout. As Fig. 5.9-b shows, the robots' workspace was roughly a rectangle having 4.1 m in width and 4.4 m in length. It was completely delimited by walls, which was either covered with wood (real walls of the laboratory) or covered with newspaper sheets (added vertical panels). The dotted polygon in Fig. 5.9-b, in the middle of the robots' arena, represents the area covered by the global camera ([8]), while the surrounding rectangle represents the area covered by the volumetric grid $\mathcal{Y}$ associated with the grid-based representation of the map ([9]). The value of most of the parameters used in the experiments are presented in Table B.6, page 252.

The environment, the initial map and the stopping criteria were fixed for all the experiments. In order to compare the performance of the team with a single robot, experiments with a single robot, similar to the one reported through Fig. 4.19, page 139, were also performed in those conditions. On average, the single robot needed $t_{k_{max}}(1) = 9053\ s$ to accomplish the mission, gathering a total of 2.742×10^6 measurements, distributed along $k_{max} = 303$ batches of measurements, with an average size $\overline{m_k} = 9049$ measurements.

Fig. 5.10 shows the maps' evolution during one of those experiments. The map of each robot is a result of measurements yielded by its own sensor and measurements received from the other robot. Robot 2 attained first a map having entropy less or equal to the pre-defined threshold H_{th} (see the map on the bottom-right of Fig. 5.10).

Table 5.2 summarizes the results obtained with the team of two robots, which are however extensible and can be generalized to teams having an arbitrary number of robots, because the robots' program is intrinsically scalable to any team size. Each line of the table represents an experiment with a different pair $(s_{k_{max}}, I_{min})$. The results in each experiment (each line of the table) refer to the robot that first attained the entropy

[6] This value assumes that a histogram having $b = 128$ bins is used to compute the voxel's discrete entropy, through the voxel's coverage quantized probability density function given by equation (4.7), page 104.

[7] For instance, this map's entropy level would be achieved if the standard deviation of every Gaussians representing the voxels' coverage was equal to $\sigma_l = 0.01,\ \forall_{l \in \mathcal{Y}}$.

[8] See section D, page 263, for further details about global localization with a color camera.

[9] See section 4.2, page 101, for further details about the grid-based representation.

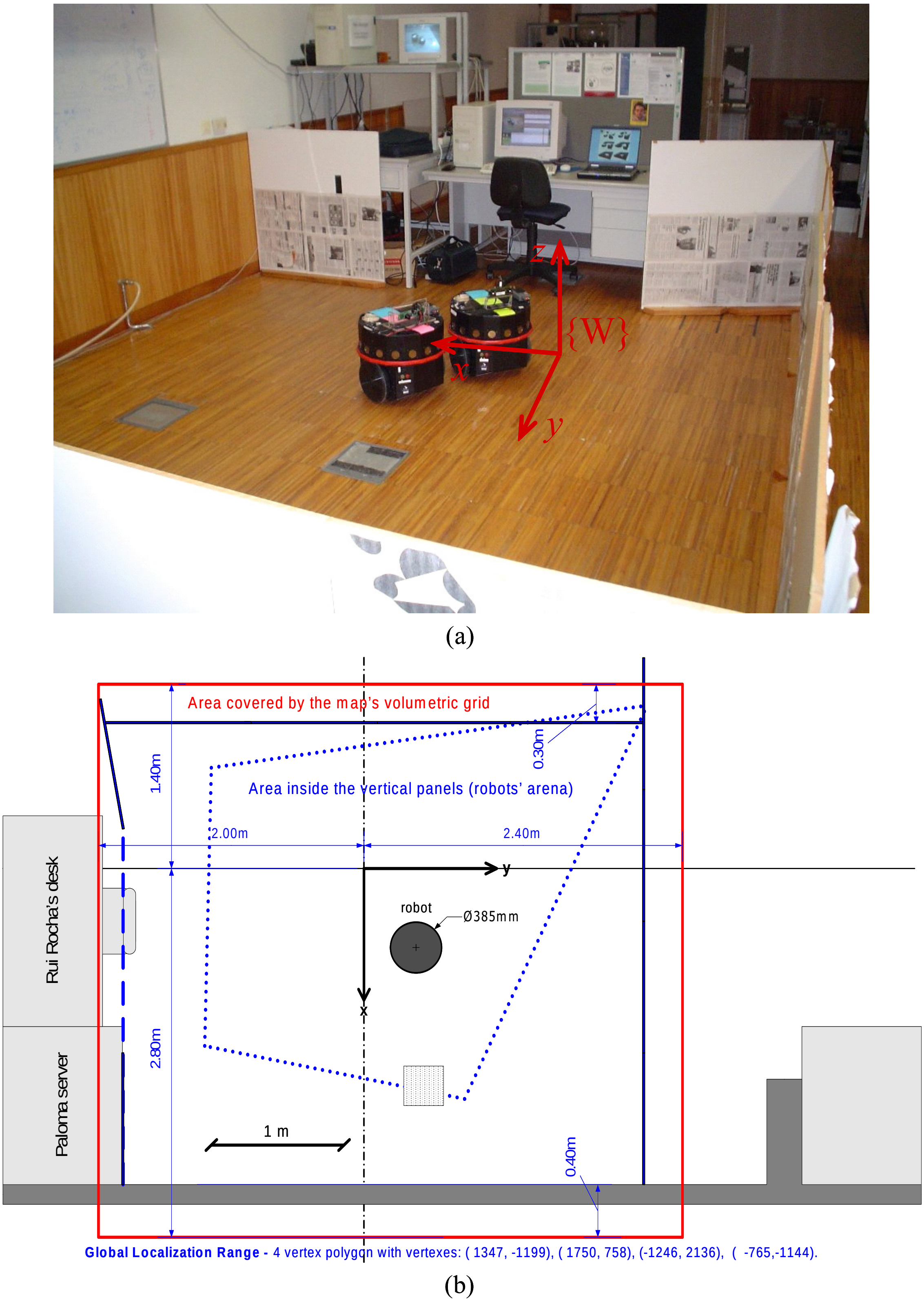

z
{W}
x
y
(a)
Area covered by the map's volumetric grid
0.30m
1.40m
Area inside the vertical panels (robots' arena)
2.00m
2.40m
y
robot
Ø385mm
x
Rui Rocha's desk
Paloma server
2.80m
1 m
0.40m
Global Localization Range - 4 vertex polygon with vertexes: (1347, -1199), (1750, 758), (-1246, 2136), (-765,-1144).
(b)

Figure 5.9: Workspace used in the experiments (July 2004): (a) photo showing the region of the Mobile Robotics Lab of ISR Coimbra, where the experiments took place; (b) plan of the workspace's layout.

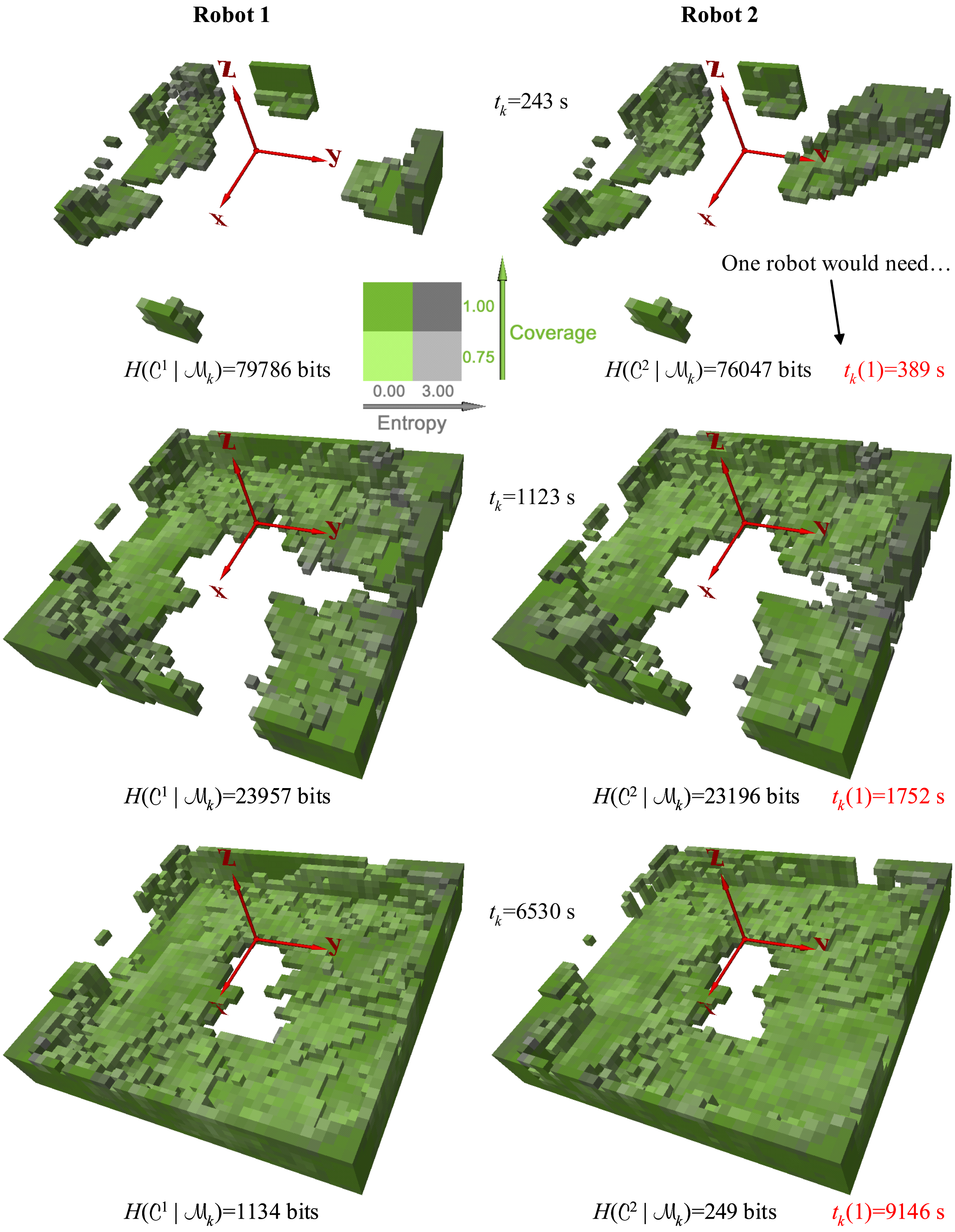

Robot 1
Robot 2
z
y
x
t_k=243 s
One robot would need…
1.00
0.75
Coverage
0.00
3.00
Entropy
$H(\mathcal{C}^1 \mid \mathcal{M}_k)$=79786 bits
$H(\mathcal{C}^2 \mid \mathcal{M}_k)$=76047 bits
$t_k(1)$=389 s
t_k=1123 s
$H(\mathcal{C}^1 \mid \mathcal{M}_k)$=23957 bits
$H(\mathcal{C}^2 \mid \mathcal{M}_k)$=23196 bits
$t_k(1)$=1752 s
t_k=6530 s
$H(\mathcal{C}^1 \mid \mathcal{M}_k)$=1134 bits
$H(\mathcal{C}^2 \mid \mathcal{M}_k)$=249 bits
$t_k(1)$=9146 s

Figure 5.10: Maps' evolution along a 3-D mapping mission with two robots: each row shows a snapshot of the map of each robot registered in the global reference frame $\{W\}$, at a different instant time t_k and entropy level $H(\mathcal{C} \mid \mathcal{M}_k)$. In every rows, the best map was held by robot 2 (maps shown on the right). The time $t_k(1)$ that a single robot would need to obtain a map with the same entropy is shown on the bottom-right of the maps of robot 2. For this case, the parameters $I_{min} = 0.1520$ and $s_{k_{max}} = 2500$ were used.

Table 5.2: Results obtained within experiments with two robots and different parameters ruling the information sharing (parameters $s_{k_{max}}$ and I_{min}).

$s_{k_{max}}$	I_{min}	$t_{k_{max}}$	$\frac{t_{k_{max}}(2)}{t_{k_{max}}(1)}$	m_T	u_T	
500	0.01450	8483	0.94	2795351	74729	3 %
1000	0.01450	8387	0.93	2726837	135661	5 %
1750	0.01450	7332	0.81	2447091	184550	8 %
2500	0.01450	6530	0.72	2375273	207636	9 %
5000	0.01450	7955	0.88	2643728	271612	10 %
20000	0	9450	1.04	3192788	1134455	36 %
20000	0.00723	7563	0.84	2453021	457390	19 %
20000	0.01450	6571	0.73	2345844	332270	14 %
20000	0.07400	7007	0.77	2676612	128345	5 %
20000	0.15200	7301	0.81	2595398	59499	2 %
20000	0.32193	7727	0.85	2930155	27323	1 %

threshold H_{th}, *i.e.* the robot having the best map at the end of the mission.

The 4th column shows the ratio between the mission execution time $t_{kmax}(2)$ with two robots and the mission execution time $t_{kmax}(1)$ with one robot. Since the coverage of each voxel was represented through a Gaussian model, the values used for I_{min}, $\{0,\ 0.00723,\ 0.01450,\ 0.07400,\ 0.15200,\ 0.32193\}$, meant an average reduction on the standard deviation of the influenced voxels by a measurement of at least $\{0\%,\ 0.5\%,\ 1\%,\ 5\%,\ 10\%,\ 20\%\}$, respectively.

Recall that when a robot acquired a new batch of m_k measurements $M_k = (\mathbf{x}_k, \mathcal{V}_k)$ through its own sensor, it sent to the other robot a batch of s_k useful measurements $S_k = (\mathbf{x}_k, \mathcal{U}_k)$, with $s_k \leq m_k \wedge s_k \leq s_{k_{max}} \wedge u_k = 0$. Conversely, a k-th batch of m_k measurements $M_k = (\mathbf{x}_k, \mathcal{V}_k)$ might not be acquired from its own sensor and thus might be a batch of u_k useful measurements sent (shared) by the other robot $M_k = R_k = (\mathbf{x}'_k, \mathcal{U}'_k)$, with $u_k = m_k \leq s_{k_{max}} \wedge s_k = 0$.

The 5th column shows the total number of measurements m_T gathered by a robot along the mission, which is given by

$$m_T = \sum_{k=1}^{k_{max}} m_k. \tag{5.22}$$

The 6th column shows the total number of received measurements from the other robot

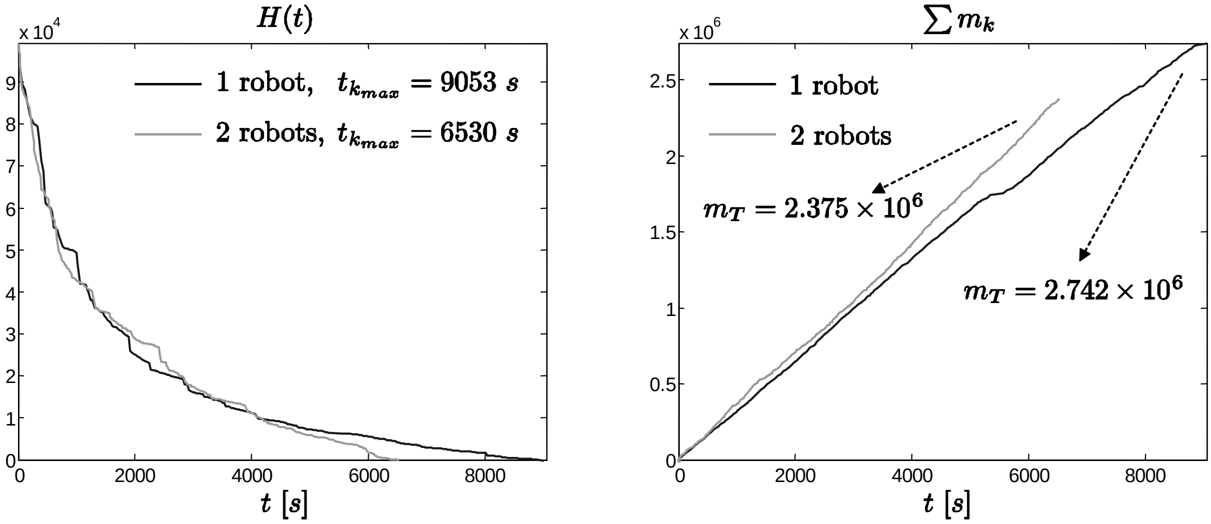


Figure 5.11: Comparison of a 3-D mapping mission using a single robot or two robots: entropy of the map along the mission (left) and cumulative number of processed measurements along the mission (right). For the presented case, the parameters' values $I_{min} = 0.0145$ and $s_{k_{max}} = 2500$ were used.

u_T, which is computed as

$$u_T = \sum_{k=1}^{k_{max}} u_k. \tag{5.23}$$

It also shows the percentage of received measurements over the total number of measurements m_T.

5.6.1 Advantages provided by cooperation

The graph on the left of Fig. 5.11 compares the map's entropy $H(t)$ for the single robot case and for the fastest experiment with two robots (4th row of Table 5.2). The robots' cooperation accelerated the reduction of the map's entropy and led to a reduction of 28% in $t_{k_{max}}$. As robots shared useful measurements through communication, each robot was able to integrate in its map a greater number of measurements per time unit and achieved a faster reduction of its map's entropy [RDC05b, RDC05a]. The graph on the right of Fig. 5.11 shows that the two values of m_T were similar, though that amount of measurements was obtained within time intervals $t_{k_{max}}$ quite different.

Besides enabling cooperation and its aforementioned benefits, the coexistence of several robots in the same workspace and the communication among robots also yield some pitfalls contributing for the degradation of the team's overall performance. Firstly, robots must share the workspace, which leads to some mutual interference. Secondly, the time spent on communicating measurements to other robots sometimes delays other robot's computational operations ([10]). Thirdly, the time required for processing received measurements through communication and updating the map upon them might not be negligible, especially for higher values of $s_{k_{max}}$.

While the two latter problems depend mostly on the communication bandwidth and the robots' computation power, the former problem, *i.e.* minimizing the interference among robots, is a key point for taking the maximum advantage from the cooperation among robots. This issue is thoroughly addressed in chapter 6.

5.6.2 Influence of communication selectivity

Fig. 5.12 presents a graph of $t_{k_{max}}$ as a function of parameter I_{min} and curves of the cumulative sum $\sum u_k$ of received measurements from the other robot along the mission, for different values of I_{min}. For this figure, the maximum number of allowable communicated measurements at a given time instant was $s_{k_{max}} = 20000$. Since the number of measurements yielded by the sensor was about 10^4 measurements, $s_{k_{max}}$ did not restrict in this case the communication for any acquired batch of measurements, because $m_k < s_{k_{max}}$, $1 \leq k \leq k_{max}$.

The graph on the left of Fig. 5.12 shows that decreasing I_{min} from 0.32193 to 0.01450 led to smaller mission execution times. However, for $I_{min} < 0.01450$, the graph of $t_{k_{max}}$ presents a remarkable inflection, which led to a fast degradation of the team's performance. This observation puts on evidence the importance of selecting the most useful information to be communicated. If the selection is too weak, most of the communicated information becomes redundant and the time spent on communicating and processing that superfluous information becomes very significant [RDC05a, RDC05b].

The curves on the right of Fig. 5.12 show that the first derivative is the same at the beginning of the mission, because $s_{k_{max}}$ is common to all of them. However, as long as the mission is executed, the derivative decreases to an extent which depends on the selectivity introduced by I_{min}.

[10] Note that, accordingly the software architecture shown in Fig. 5.8, after a robot updates its map (module MAPUPD), it must wait to completely send useful measurements to other robots (module MEASPROVID), before selecting another exploration viewpoint (module SURVCTRL) and thus starting a new sensing cycle. Nevertheless, note that the robot's communication protocol prevents the robot to be hung on for many time when communication errors occur.

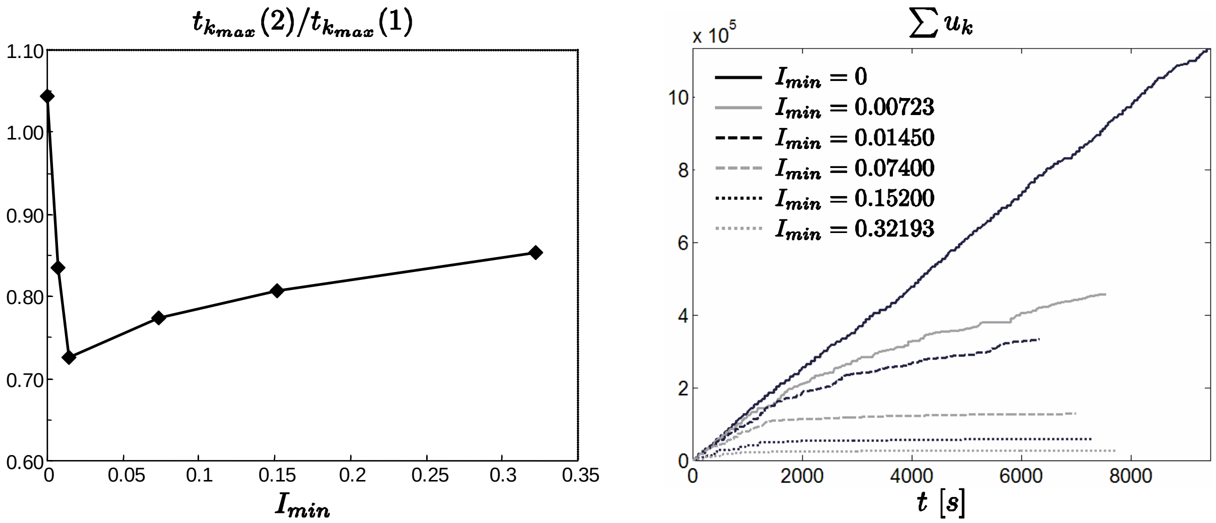


Figure 5.12: Influence of the information selectivity — parameter I_{min} — on the mission execution time with two robots, using $s_{k_{max}} = 20000$: mission execution time (left) and cumulative number of received measurements from the other robot along the mission (right).

5.6.3 Influence of the amount of communication

Fig. 5.13 presents a graph of $t_{k_{max}}$ as a function of parameter $s_{k_{max}}$ and curves of the cumulative sum $\sum u_k$ of received measurements from the other robot along the mission, for different values of $s_{k_{max}}$. For this figure, I_{min} was fixed and set to 0.01450.

The graph on the right of Fig. 5.13 shows that reducing the communication bandwidth $s_{k_{max}}$ always led to an increase of $t_{k_{max}}$ and thus a worse team's performance [RDC05b, RDC05a]. As cooperation in a 3-D mapping mission relies completely on explicit communication, restricting it also restricts the extent of cooperation. However, in the case of I_{min}, being selective to some extent is beneficial in order to select the most useful information and to avoid the communication of redundant information.

The curves on the right of Fig. 5.13 show that the first derivative by the end of the mission is the same, because I_{min} is common to all of them; and that it is as high as $s_{k_{max}}$ at the start, since this parameter configures a bandwidth limitation.

5.7 Summary and conclusion

This chapter started by proposing a distributed architecture model for building volumetric maps with a team of cooperative robots, whereby each robot is altruistically committed to share useful measurements with its teammates [RDC05b, RDC05a].

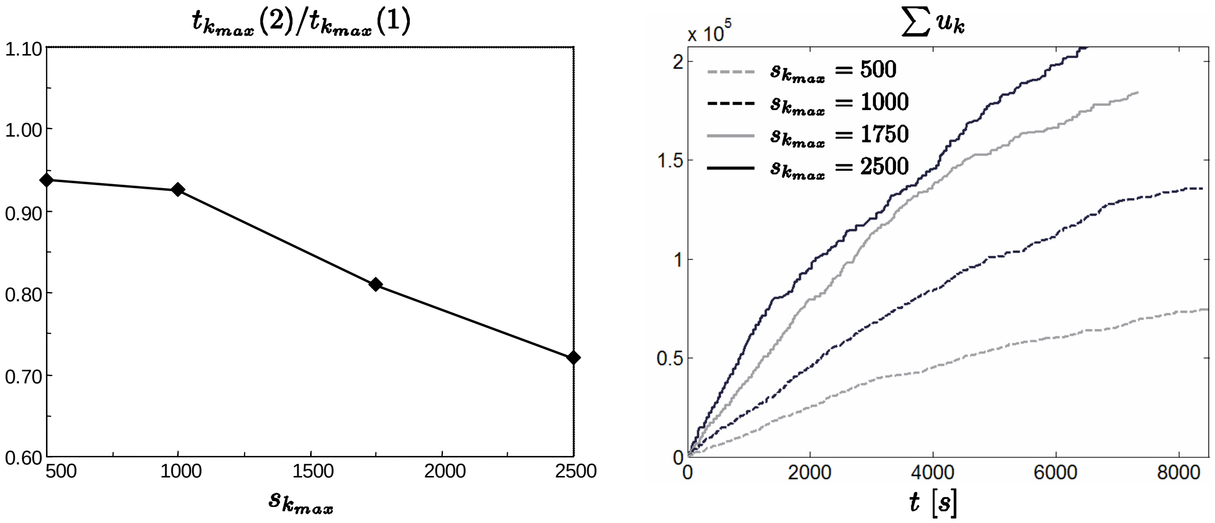


Figure 5.13: Influence of the maximum number of communicated measurements — parameter $s_{k_{max}}$ — on the mission execution time with two robots, using $I_{min} = 0.01450$: mission execution time (left) and cumulative number of received measurements from the other robot along the mission (right).

Given a specific multi-robot mission and its associated performance measure, a piece of information is as useful as it yields a high ratio between performance improvement and cost increasing. In order to devise a formal measure of information utility, a specific case-study was thoroughly described to get some insight about the information utility concept itself. In this case study, several robots explore a given environment with the aim of finding regions of interest and perform work therein [RDC03]. Results obtained in simulations of this type of multi-robot mission demonstrated the correlation between lesser execution times and higher utility of the information shared among robots.

Using the grid-based probabilistic volumetric mapping framework proposed in chapter 4, an entropy-based measure of information utility was formulated, whereby the utility of a given range measurement is as useful as it contributes to improve the robot's map. This measure was then applied to the distributed architecture proposed at the beginning of the chapter, in order to support a cooperation scheme for sharing efficiently sensory data within a team of robots. At the end of the chapter, after describing the implementation of the proposed cooperation scheme in mobile robots, experimental results obtained in cooperative volumetric mapping missions with mobile robots [RDC05b, RDC05a] were presented and discussed.

Those experimental results yielded some general guidelines for developing communication schemes aiming at fostering cooperation. The maximum size of a communicated batch $s_{k_{max}}$ is imposed by the communication channel capacity, whereas the minimum information utility I_{min} of each communicated measurement has to be tuned in an intelligent way, whereby its selective power is beneficial for the robotic team's performance: on one hand, it should be selective enough to avoid communicating redundant information; on the other hand, it should not be too selective, so as to enable efficient information sharing and cooperation among robots.

The proposed cooperation scheme based on sharing useful sensory data allows to accomplish a volumetric mapping mission with two robots in less time than a single robot. This was demonstrated in the experiments reported in this chapter. This kind of cooperation is essential to take advantage from the spatial distribution and computational parallelism yielded by multiple robots. It was shown that if communication is not used properly, it may degrade significantly the team's performance due to redundant information that is communicated among robots.

Nevertheless, even when that cooperation via explicit communication is well configured, there is an important challenge raised by the multi-robot system, which has not been addressed yet: since robots must share the workspace, they may either perform not useful work or interfere each other, if they do not coordinate properly their exploration actions.

There are mainly three types of undesirable phenomena that may occur within a team of robots performing a mapping mission. Firstly, a robot may sense a region that has already been sensed by another robot. Secondly, a robot may be blocked by other robots when moving between two consecutive exploration viewpoints. And, thirdly, the robot's sensor may be occluded by the presence of other robots in its field of view, when measuring the environment.

These phenomena degrade unavoidably the team's performance and justify why the mission execution time reduction yielded by two robots, when compared with a single robot, was not so impressive. In the best case, two robots took 72% of the time needed by a single robot to accomplish the same mission, which is far away from a linear performance gain with team size, wherein two robots would spend just half the time of a single robot.

Next chapter addresses the aforementioned problems by complementing the distributed architecture proposed in this chapter with a suitable coordinated exploration method, aiming at attaining more effective cooperation and better team's performance.

Chapter 6

Improving cooperation through coordinated exploration

Previous chapters addressed important issues related with building volumetric maps by using mobile robots equipped with range sensors, especially stereo-vision sensors.

Chapter 4 proposed a grid-based probabilistic framework for representing volumetric maps and updating them upon range measurements, which modeled explicitly uncertainty through the entropy concept [RDC05d, RDC05a]. Using this important feature, the framework was used to propose a straightforward entropy-based formulation of frontier-based exploration [Yam98], whereby the robot's sensor is directed to frontier voxels between more explored and less explored regions.

Chapter 5 used the aforementioned framework to propose a completely distributed architecture model, aiming at controlling a team of cooperative mobile robots and building a volumetric map in less time than a single robot [RDC05b, RDC05a]. Besides each robot being capable of building a map alone (as in chapter 4), it is also committed to share altruistically with its teammates the most useful measurements obtained from its sensor. This cooperation scheme takes advantage from the space and time distribution yielded by multiple robots, by virtually magnifying the sensory capabilities of each individual robot.

Building a volumetric map is essentially an exploration mission. The goal is to completely sense the whole environment so as to accumulate sufficient sensory evidence and build a consistent spatial model with low uncertainty, *i.e.* a map with low entropy. Formally, accordingly with equation (4.10), page 106, given a lower entropy bound H_{th} representing the maximum uncertainty of the final map, the goal is to decrease the map's uncertainty (entropy) until it is less than H_{th}. Obviously, the time required to accomplish this goal should be minimized, by acquiring as much new information about the environment as possible with every sensing cycle. The goal of the gradient entropy-based exploration strategy proposed in section 4.6.2, page 121, is precisely to minimize the mission execution time, by maximizing the information gain in each sensing cycle [RDC05b, RDC05d, RDC05a].

Chapter 4 validated experimentally the entropy-gradient exploration strategy with a single robot; it demonstrated that the exploration algorithm nicely converges to a map with lower entropy. However, chapter 5 presented experimental results with two cooperative robots that allowed to conclude that it does not yield impressive results with multiple robots, because it does not avoid interference among robots due to lack of coordination [RDC05b, RDC05a]. In the best case, two robots took 72% of the time needed by a single robot to accomplish the same mission, which is far away from a linear performance gain with team size, wherein two robots would spend just half the time of a single robot.

This chapter addresses the aforementioned problem by complementing the distributed architecture proposed in chapter 5 with a suitable coordinated exploration method [RDC05c], aiming at achieving more effective cooperation and better team's performance. After clearly identifying the undesirable behaviors that arise because of lack of coordination, this chapter extends the distributed architecture model for volumetric mapping, whereby each robot has a higher level of awareness about the other robots' state in order to coordinate its exploration actions with the rest of the team.

The most important feature of the coordinated exploration method is to avoid sensing overlapping regions with different robots. This problem is formulated as a mutual information minimization problem. In order to support this formulation, mathematical expressions for computing mutual information between sets of random variables are derived. Then, the entropy gradient-based exploration method, previously proposed in chapter 4, is refined with a coordination mechanism, which, besides minimizing sensing overlapping, also avoids other undesirable inter-robot interference phenomena [RDC05c].

The chapter ends by presenting and discussing experimental results, obtained from volumetric mapping experiments with mobile robots and from computer simulations, so as to demonstrate the performance improvement yielded by the proposed coordinated exploration method.

6.1 How robots can improve the team's performance?

The experiments reported in section 5.6, page 168, regarding volumetric maps obtained with multiple cooperative mobile robots, lead to the conclusion that cooperation through

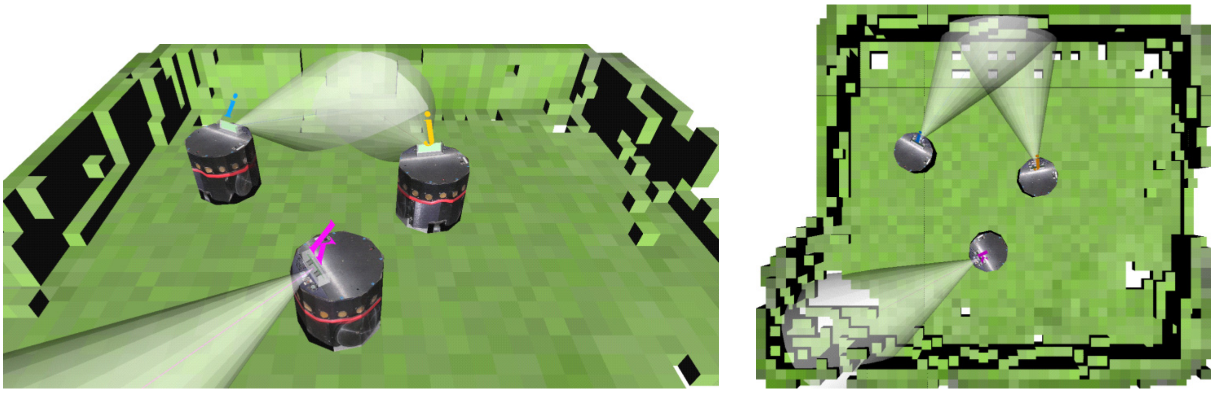


Figure 6.1: Example showing two robots sensing regions that overlap each other (robots i and j).

information sharing should be complemented with a suitable coordination mechanism, in order to achieve better performance and effective cooperation. This section describes thoroughly the main problems that arise due to lack of cooperation. The goal is to identify what coordination actions robots should follow in order to overcome those problems and improve the team's overall performance. This knowledge is used in the following sections to propose a coordination mechanism.

All of the undesirable phenomena that arise due to lack of coordination in the exploration with multiple robots share a common and obvious root: *several robots working in the same workspace may interfere each other.* There are essentially three types of undesirable phenomena [RDC05c]: robots sensing regions that overlap each other; constraining the robot's motion within the workspace due to stopped robots that either block some trajectories or cause some regions to be unreachable; the robot's sensor may be occluded by the presence of other robots in front of it.

6.1.1 Avoiding sensing overlapping regions

The first problem occurs when a robot senses a region that is already being sensed by another robot. Fig. 6.1 shows an example wherein two of the robots — robots i and j — are sensing regions that overlap each other. This is an undesirable situation because of two reasons.

Firstly, sensing coincident regions means a loss of efficiency, since it yields a less attainable information gain than when robots sense regions that do not overlap each other. Recall that the main goal in an exploration mission is to minimize the time required to completely sense the environment. If two or more robots sense overlapping regions, they are using their sensors to acquire information from the environment that could be acquired by just one of them. Obviously, this redundancy significantly degrades the team's overall performance, because the robot's resources (*e.g.* sensors, processor, memory *etc.*), used to acquire and process the *same* information than other robots, could be used to acquire and process other non-redundant sensory information.

Secondly, motion interference (*e.g.* a robot being blocked by another stopped robot) and mutual sensor occlusions are naturally more likely to occur when robots explore nearby regions.

For these reasons, each robot should be programmed so as to explore regions that are not already being explored by other robots [RDC05c]. This necessarily requires that each robot be aware of what are the selected viewpoints by the other robots in the team, so as to estimate what are the regions already covered by the rest of the team. With this purpose, the architecture model proposed in section 5.1, page 147, must be extended in order to support that increased level of awareness through an extra communication flow among robots.

6.1.2 Avoiding not reachable exploration viewpoints

The second problem occurs when a robot selects a given exploration viewpoint for which its chosen trajectory is blocked by other robots. These robots are usually stopped in order to acquire new range data and process it and, thus, remain therein until starting to move to their next selected viewpoint. There are cases for which these stopped robots may cause the selected exploration viewpoint be temporarily completely unreachable, because there is no any alternative trajectory that is not blocked by them.

Fig. 6.2 shows an example wherein a robot — robot i — cannot reach its selected viewpoint by following a straight trajectory from its current pose, due to the presence of other two stooped robots — robots j and k — located in front it. Note that it is certainly possible in this case the robot choosing another alternative trajectory surrounding robots j and k, though it has a longer distance than the straight trajectory and, thus, requires necessarily more time to be completed.

When a robot selects a trajectory that is blocked by other stopped robots, it may choose between: looking for another alternative trajectory, for which it is not blocked; selecting another exploration viewpoint; or waiting until the blocking robots free the trajectory. In any case, there is always some loss of performance.

In the first and second cases, there is an extra required computation after the robot

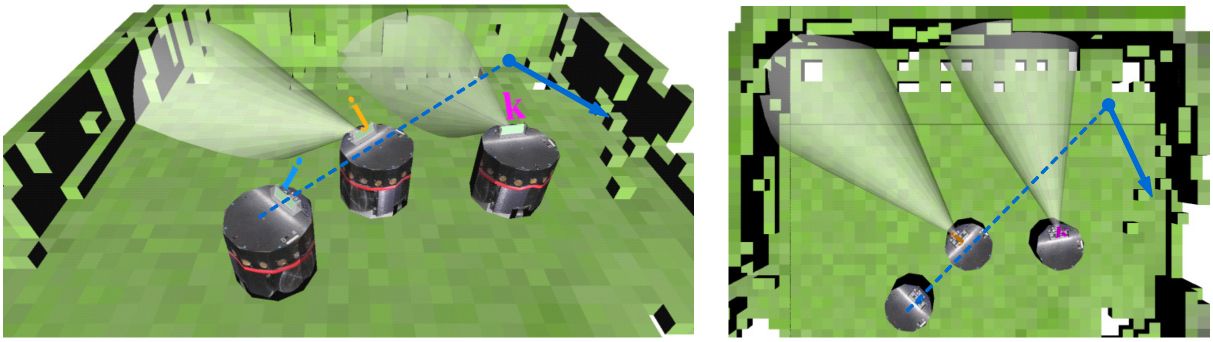


Figure 6.2: Example showing a robot blocked by other robots: robot i is blocked by robots j and k when moving to its selected viewpoint.

detects the blocking situation, which necessarily requires some computation time. In the third case, the robot's waiting time is completely worthless for the mapping mission, since the robot is just doing nothing until the other robots do not block it; this choice is thus the worst one and should not be followed. Moreover, in the first two cases, the robot spends some worthless time to move until reaching the stopping robot and detecting the blocking situation.

By simplicity, it is assumed that robots always follow straight trajectories between consecutive exploration viewpoints. This is indeed the case of experiments reported in section 4.8, page 136, and section 5.6, page 168. This assumption is justified by two reasons. Firstly, because the environments being mapped are mainly uncluttered ([1]), following straight trajectories is a viable strategy, which has the advantage of minimizing the traveled distance and the time required to move the robots. Secondly, when the aforementioned blocking situation occurs, it is usually more worth to select another exploration viewpoint than recovering from it by choosing a more complex trajectory; this non-straight trajectory would increase the robot's motion control complexity and would be more time-consuming.

Nevertheless, better than recovering from the aforementioned blocking situation is just to avoid it. Likewise in the case of sensing overlapping, this only requires that the robot be aware of the viewpoints selected by its teammates. Using this knowledge and its current map, when the robot has to choose an exploration viewpoint from a set of viewpoints, it just assesses how reachable they are and penalizes those viewpoints that are likely not reachable due to the presence of stopped robots or other obstacles cluttering the environment [RDC05c]. This necessarily slightly increases the computation burden

[1] See, for example, Fig. 4.18, page 137.

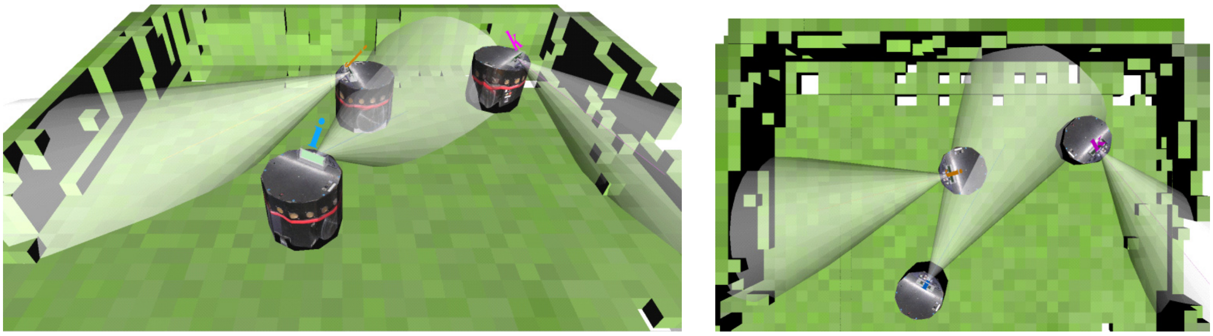


Figure 6.3: Example showing a robot being interfered by other two robots: the sensor of robot i is partially occluded by robots j and k, located in front of it.

required to select the robot's next viewpoint, but it prevents the robot to lose time on making worthless movements towards blocking situations, or recovering from them.

6.1.3 Avoiding partial occlusions of the robot's sensor

The third problem occurs when the robot's sensor is being used to measure the environment and is partially occluded by the presence of other robots located within its field of view. These robots usually remain stopped for some time while they acquire and process therein new range data.

Fig. 6.3 shows an example wherein the sensor of a robot — robot i — is partially occluded by two robots located in front of it — robots j and k. Since the team's goal is to build a map of the environment *itself*, wherein the robots should not appear on it, those partial occlusions are highly undesirable because they reduce the robot's field of view and thus limit the robot's sensory capabilities.

There are two possible choices to deal with an occlusion of the robot's sensor. Firstly, if the robot is aware of the other robots' localization, it may just filter, *i.e.* discard, those range measurements that are due to the presence of its teammates. Secondly, if the robot is not aware of its teammates' localization, it cannot detect the occlusion situation and will add to its map some noise, which is caused by integrating range measurements whereby moving obstacles located somewhere in the environment's free space are represented in the map. Nevertheless, the robot suffers from some loss of efficiency in both cases: in the former case, the robot cannot take full advantage of its sensory capabilities; in the latter case, the robot will obtain a noisier map of the environment due to integrating on it undesirable dynamic objects, *i.e.* moving robots.

Therefore, likewise in the cases of sensory overlapping and not reachable exploration viewpoints, the robot should be aware of the viewpoints selected by the other robots in the team, so as to be able to predict if its sensor will be occluded by other robots if a given viewpoint is selected [RDC05c]. Using that knowledge and extra computation, the robot may avoid to select an exploration viewpoint yielding undesirable partial occlusions.

6.2 Distributed architecture for 3-D mapping with improved coordination

The previous section identifies the main problems that arise in a team of mobile robots performing volumetric missions, because of the lack of coordination of their exploration actions. It also points out informally solutions to overcome those problems, which basically requires that each robot be aware of what are the selected viewpoints by the other robots in the team. Moreover, the robot needs to have sufficient information about the other robots' sensors, so as to be able to estimate what are the regions covered by its teammates. Therefore, the distributed architecture model described in section 5.1, page 147, must be extended in order to support this increased level of awareness.

Consider again a fleet $\mathcal{F} = \{1, \ldots, n\}$ of n robots equipped with on-board range sensors. Figures 6.4 and 6.5 depict complementary views of a revised version of the distributed architecture model for building volumetric maps with multiple cooperative robots [RDC05c]. Although both figures refer to an individual robot $i \in \mathcal{F}$, the interaction with the rest of the team, *i.e.* the set of robots $\mathcal{F} \backslash i$, is represented through the communication block and its associated data flow, which is represented in Fig. 6.4.

Within this revised architecture model, communication among robots is used with two purposes:

- For sharing useful range measurements among robots;
- To let other robots know the selected sensor's viewpoint and the visibility parameters of each individual robot, in order to coordinate the exploration process with multiple robots.

While the former type of communication is already supported by the previous version (see section 5.1, page 147), the latter type of communication is introduced in this revised version to increase the robot's awareness about its teammates.

Now, when the robot i has to select its next exploration viewpoint Y^s, besides using its current map $\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$, it uses its current visibility parameters r_i and α_i, and visibility

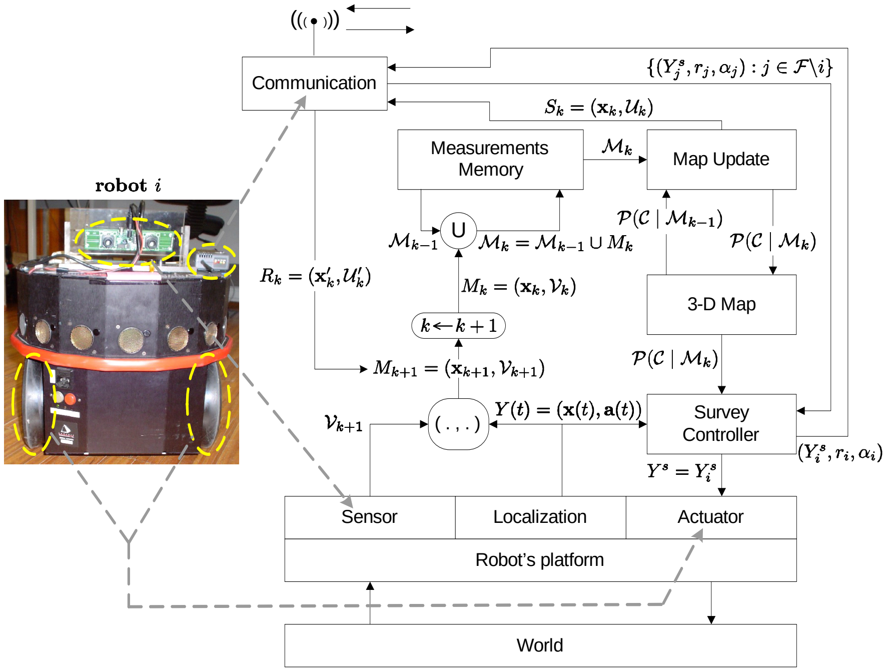

Communication
$\{(Y_j^s, r_j, \alpha_j) : j \in \mathcal{F}\backslash i\}$
$S_k = (\mathbf{x}_k, \mathcal{U}_k)$
Measurements Memory
$\mathcal{M}_k$
Map Update
robot i
U
$\mathcal{M}_{k-1}$
$\mathcal{M}_k = \mathcal{M}_{k-1} \cup M_k$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_{k-1})$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$R_k = (\mathbf{x}'_k, \mathcal{U}'_k)$
$M_k = (\mathbf{x}_k, \mathcal{V}_k)$
3-D Map
$k \leftarrow k+1$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$M_{k+1} = (\mathbf{x}_{k+1}, \mathcal{V}_{k+1})$
$\mathcal{V}_{k+1}$
$(\cdot,\cdot)$
$Y(t) = (\mathbf{x}(t), \mathbf{a}(t))$
Survey Controller
(Y_i^s, r_i, α_i)
$Y^s = Y_i^s$
Sensor
Localization
Actuator
Robot's platform
World

Figure 6.4: Block diagram showing the relation between different parts of the process and the resources of a given robot of the team, including the interaction required to support the coordinated exploration.

information $\{(Y_j^s, r_j, \alpha_j) : j \in \mathcal{F}\backslash i\}$ about all the other robots in the team $\mathcal{F}\backslash i$ ([2]). The goal is to select the most uncertain regions of the map and coordinate the exploration with multiple robots, by minimizing mutual sensory information and interference.

Whenever a robot selects a viewpoint Y^s for its sensor, the communication module is used to communicate the tuple (Y_i^s, r_i, α_i), *i.e.* the new selected viewpoint and its current visibility parameters. This minimal extra communication ensures that all robots have sufficient awareness to coordinate the exploration. Note that each robot updates its visibility parameters r and α whenever it gets a new batch of measurements (see Fig. 6.5).

[2] The definition of the visibility parameters r and α is given latter on in this chapter. For now, assume that these parameters can be used, in conjunction with the pose of a robot, to estimate its sensor's field of view. Recall that this is required to avoid selecting an exploration viewpoint whose sensed region overlaps with regions already being sensed by other robots.

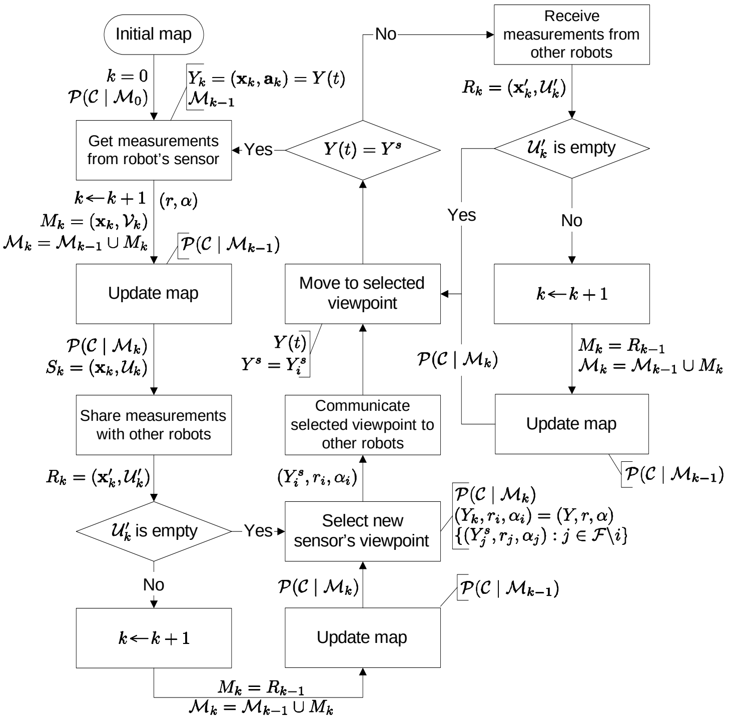

Initial map
$k = 0$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_0)$
$Y_k = (\mathbf{x}_k, \mathbf{a}_k) = Y(t)$
$\mathcal{M}_{k-1}$
Get measurements from robot's sensor
Yes
$Y(t) = Y^s$
No
Receive measurements from other robots
$R_k = (\mathbf{x}'_k, \mathcal{U}'_k)$
$\mathcal{U}'_k$ is empty
Yes
No
$k \leftarrow k+1$
(r, α)
$M_k = (\mathbf{x}_k, \mathcal{V}_k)$
$\mathcal{M}_k = \mathcal{M}_{k-1} \cup M_k$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_{k-1})$
Update map
Move to selected viewpoint
$k \leftarrow k+1$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$S_k = (\mathbf{x}_k, \mathcal{U}_k)$
$Y(t)$
$Y^s = Y^s_i$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$M_k = R_{k-1}$
$\mathcal{M}_k = \mathcal{M}_{k-1} \cup M_k$
Share measurements with other robots
Communicate selected viewpoint to other robots
Update map
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_{k-1})$
$R_k = (\mathbf{x}'_k, \mathcal{U}'_k)$
(Y^s_i, r_i, α_i)
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$(Y_k, r_i, \alpha_i) = (Y, r, \alpha)$
$\{(Y^s_j, r_j, \alpha_j) : j \in \mathcal{F} \backslash i\}$
$\mathcal{U}'_k$ is empty
Yes
Select new sensor's viewpoint
No
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_k)$
$\mathcal{P}(\mathcal{C} \mid \mathcal{M}_{k-1})$
$k \leftarrow k+1$
Update map
$M_k = R_{k-1}$
$\mathcal{M}_k = \mathcal{M}_{k-1} \cup M_k$

Figure 6.5: Flowchart showing the data flow of a given robot of the team, including the information flow required to support the coordinated exploration.

6.3 Mutual information between sets of discrete random variables

The situation wherein a robot senses a region that overlaps with regions already being sensed by other robots is addressed in section 6.1.1, page 181. The reasons why this is an undesirable behavior are also addressed therein. In order to overcome the problem, when the robot has to choose its next exploration viewpoint from a set of alternatives, it should quantify the overlapping associated with each potential exploration viewpoint, so as to minimize that overlapping.

Using the probabilistic framework proposed in chapter 4, which is based on entropy, the aforementioned overlapping can be quantified through the mutual information between the volumes sensed by different robots. The coverage of each one of those volumes is modeled through sets of discrete random variables ([3]), being each one of these variables a representation of a given voxel's coverage. The sensing overlapping problem can be thus

[3] Due to the statistically independence assumption concerning the coverage of different voxels (see section 4.2, page 101), those sets of random variables are, indeed, *sets of statistically independent random variables*.

formulated as a mutual information minimization problem. This section uses the theoretical background presented in chapter 3 to extend the definition of mutual information between two discrete random variables to sets of random variables [RDC05c]. This knowledge is used afterwards to deal with the aforementioned problem through a coordination mechanism.

The definition of mutual information given by equation (3.22), page 90, can be easily generalized to the mutual information between a set of random variables $\mathcal{X} = \{X_1, \dots, X_n\}$ and a variable Y as [RDC05c]

$$
\begin{aligned}
I(\mathcal{X}; Y) &= I(X_1, \dots, X_n; Y) && (6.1)\\
&= H(X_1, \dots, X_n) - H(X_1, \dots, X_n \mid Y) && (6.2)\\
&= \sum_{i=1}^{n} H(X_i \mid X_1, \dots, X_{i-1}) - \sum_{i=1}^{n} H(X_i \mid X_1, \dots, X_{i-1}, Y) && (6.3)\\
&= \sum_{i=1}^{n} H(X_i \mid X_1, \dots, X_{i-1}) - H(X_i \mid X_1, \dots, X_{i-1}, Y) && (6.4)\\
&= \sum_{i=1}^{n} I(X_i; Y \mid X_1, \dots, X_{i-1}). && (6.5)
\end{aligned}
$$

Equation (6.2) means that the mutual information is the information of the set $\mathcal{X}$ minus its information when the variable Y is given ([4]). Equation (6.3) is obtained by applying to both terms of equation (6.2) the joint entropy definition for a set of discrete random variables, which is given by equation (3.18), page 89. Equation (6.5) is obtained from equation (6.4) by using the definition of conditioned mutual information provided by equation (3.28), page 91. A simple example of the application of equation (6.5) is

$$I(\mathcal{X}; X_1) = H(X_1). \tag{6.6}$$

See section (A.2.1), page 247 for the details about this example. It shows that the information which is contained both in $\mathcal{X}$ and X_1 is simply the information of the random variable X_1.

Note that substituting the variable Y in equation (6.1) by a set of discrete random variables $\mathcal{Y}$ nothing changes in the steps whereby equation (6.5) is derived. Therefore, we can easily generalize it as [RDC05c]

$$I(\mathcal{X}; \mathcal{Y}) = \sum_{i=1}^{n} I(X_i; \mathcal{Y} \mid X_1, \dots, X_{i-1}). \tag{6.7}$$

[4] This is analogous to the mutual information concept that raised the definition given by equation (3.22), page 90.

Equation (6.7) can be recursively applied to define the *mutual information between two sets of random variables* $\mathcal{X}^1 = \{X_1^1, \ldots, X_m^1\}$ and $\mathcal{X}^2 = \{X_1^2, \ldots, X_n^2\}$ as [RDC05c]

$$\begin{aligned} I(\mathcal{X}^1; \mathcal{X}^2) &= I(X_1^1, \ldots, X_m^1; X_1^2, \ldots, X_n^2) \\ &= \sum_{i=1}^{m} I(X_i^1; X_1^2, \ldots, X_n^2 \mid X_1^1, \ldots, X_{i-1}^1) \quad (6.8) \\ &= \sum_{i=1}^{m} I(X_1^2, \ldots, X_n^2; X_i^1 \mid X_1^1, \ldots, X_{i-1}^1) \quad (6.9) \\ &= \sum_{i=1}^{m} \sum_{j=1}^{n} I(X_j^2; X_i^1 \mid X_1^1, \ldots, X_{i-1}^1, X_1^2, \ldots, X_{j-1}^2) \quad (6.10) \\ &= \sum_{i=1}^{m} \sum_{j=1}^{n} I(X_i^1; X_j^2 \mid X_1^1, \ldots, X_{i-1}^1, X_1^2, \ldots, X_{j-1}^2) \quad (6.11) \end{aligned}$$

From equation (6.8) to equation (6.9), and from equation (6.10) to equation (6.11), it was used the mutual information symmetry property provided by equation (3.25), page 90. An example of the application of equation (6.11) is

$$I(X_1, X_2, X_3; Y_1, Y_2) = H(X_1, X_2, X_3) + H(Y_1, Y_2) - H(X_1, X_2, X_3, Y_1, Y_2). \quad (6.12)$$

The proof of this equality is presented in section A.2.2, page 248. The mutual information between the two sets of random variables is thus the sum of the information contained in each set — first and second terms — minus the information contained in the union of both sets — third term. If the mutual information is null, those two quantities are equal; otherwise, the union set has less information than the sum of information contained by each individual set. An interesting particular case for the above example is

$$I(X_1; X_2, X_3; X_1, X_2) = H(X_1, X_2), \quad (6.13)$$

wherein two variables — X_1 and X_2 — are contained in both sets.

The previous example suggests a generalization of equation (3.23), page 90, which relates joint entropy and mutual information of two random variables, to a *relation between joint entropy and mutual information* of two sets of random variables [RDC05c]. Indeed, given any pair of sets of random variables, it can be easily obtained by induction the relation

$$\begin{aligned} I(\mathcal{X}^1; \mathcal{X}^2) = H(\mathcal{X}^1) + H(\mathcal{X}^2) - H(\mathcal{X}^1 \cup \mathcal{X}^2) \Leftrightarrow \quad (6.14) \\ H(\mathcal{X}^1 \cup \mathcal{X}^2) = H(\mathcal{X}^1) + H(\mathcal{X}^2) - I(\mathcal{X}^1; \mathcal{X}^2), \quad (6.15) \end{aligned}$$

wherein $H(\mathcal{X}^1 \cup \mathcal{X}^2)$ denotes the joint entropy of the union set of random variables $\mathcal{R} = \mathcal{X}^1 \cup \mathcal{X}^2$. This means that $\mathcal{R}$ may have less than $m + n$ variables if $\mathcal{I} = \mathcal{X}^1 \cap \mathcal{X}^2 \neq \emptyset$, *i.e.* if one or more random variables are contained in both sets (strict statistical dependence).

6.3.1 Special case of sets of statistically independent variables

Accordingly with the inequality given by equation (3.19), page 89, the joint entropy of a set $\mathcal{X} = \{X_1, \ldots, X_n\}$ of statistically independent random variables is given by

$$H(\mathcal{X}) = H(X_1) + H(X_2) + \ldots + H(X_n) = \sum_{i=1}^{n} H(X_i) \tag{6.16}$$

and the union of sets of statistically independent random variables is given by

$$H(\mathcal{X}^1 \cup \mathcal{X}^2 \cup \ldots \cup \mathcal{X}^n) = \sum_{X_k \in \mathcal{X}^1 \cup \mathcal{X}^2 \cup \ldots \cup \mathcal{X}^n} H(X_k), \tag{6.17}$$

i.e. it is simply the sum of its variables' entropy [RDC05c].

Consider two subsets of *independent random variables* $\mathcal{X}^1 \subseteq \mathcal{X}$ and $\mathcal{X}^2 \subseteq \mathcal{X}$. Using equation (6.17), equation (6.14) may be re-written as

$$\begin{aligned} I(\mathcal{X}^1; \mathcal{X}^2) &= H(\mathcal{X}^1) + H(\mathcal{X}^2) - H(\mathcal{X}^1 \cup \mathcal{X}^2) \\ &= \sum_{X_i^1 \in \mathcal{X}^1} H(X_i^1) + \sum_{X_j^2 \in \mathcal{X}^2} H(X_j^2) - \sum_{X_k \in \mathcal{X}^1 \cup \mathcal{X}^2} H(X_k), \end{aligned} \tag{6.18}$$

which is a special case that is only applicable to sets of independent random variables. All of the terms in the two first sums of equation (6.18) are cancelled by the terms in the last sum, except the terms related with variables belonging to both sets. Thus, we have two cases:

$$I(\mathcal{X}^1; \mathcal{X}^2) = \begin{cases} \sum_{X_i \in \mathcal{X}^1 \cap \mathcal{X}^2} H(X_i), & \mathcal{X}^1 \cap \mathcal{X}^2 \neq \emptyset, \\ 0, & \text{otherwise.} \end{cases}, \tag{6.19}$$

which means that any mutual information between the two sets is due to variables belonging to both sets [RDC05c].

6.4 Uncertainty covered by the robot's sensor

The main feature of the multi-robot coordinated exploration strategy proposed in this chapter is to avoid the undesirable situation wherein a robot senses a region that overlaps with regions already being sensed by other robots. This undesirable behavior is thoroughly addressed in section 6.1.1, page 181. Moreover, accordingly with section 6.1.3, page 184, each robot should also avoid the interference caused by teammates appearing within the region covered by its sensor, since it yields undesirable occlusions. The robot's ability to overcome both of the aforementioned situations depends on its ability to formally represent the region covered by the sensor of each robot, including itself.

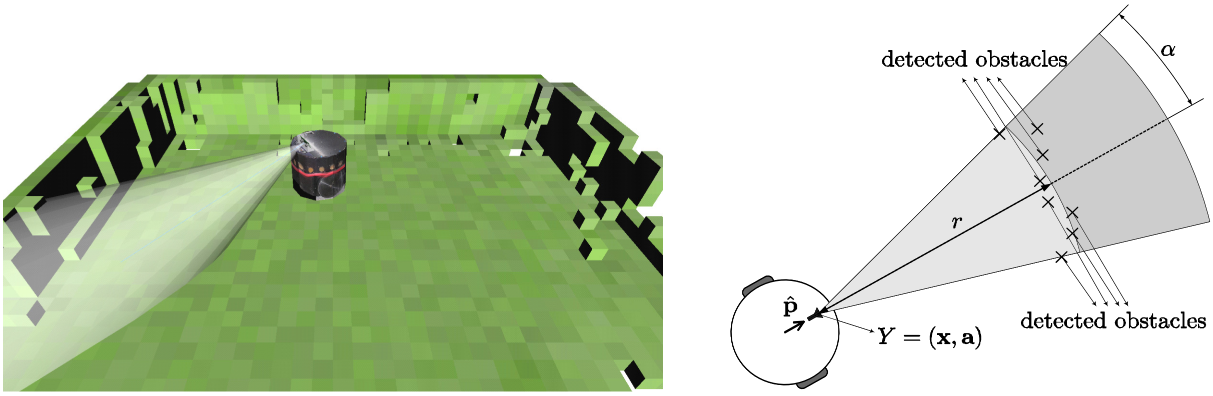


Figure 6.6: Robot's visibility: it is defined as the light-grey shaded region represented in the diagram shown on the right, which is computed upon the maximum sensor's range r and the maximum angle α with the heading $\hat{\mathbf{p}}$. The pose of the robot's sensor is $Y = (\mathbf{x}, \mathbf{a})$, being $\mathbf{x} \in \mathbb{R}^3$ its position and $\mathbf{a} = [\theta, \phi, \psi]^T$ its orientation. Both parameters r and α change dynamically with the surrounding environment and, thus, the robot's visibility is usually smaller than the potential sensor's range (dark-grey shaded region in the bottom diagram).

This section uses the probabilistic framework proposed in chapter 4 to present the notions of robot's visibility and robot's visible map [RDC05c]. Associated with these notions, the definitions of robot's visible map entropy and mutual information between the visible maps of different robots are also presented. The former definition is a formal measure of the map's uncertainty that is covered by the robot's sensor. The latter definition is a formal measure of the overlapping between the regions sensed by different robots. These definitions are used afterwards to formulate the sensing overlapping as a mutual-information minimization problem, and to propose a multi-robot coordinated exploration strategy that aims at overcoming the problems described in section 6.1.

6.4.1 Robot's visibility

Recall that $Y = (\mathbf{x}, \mathbf{a})$ denotes the robot's pose, which includes its position $\mathbf{x} \in \mathbb{R}^3$ and orientation $\mathbf{a} = [\theta, \phi, \psi]^T$. The *robot's visibility* is defined as the maximum volume that a robot can sense from its current pose (see Fig. 6.6). Given the maximum range distance r and the maximum angle α with the heading $\hat{\mathbf{p}}$ of the robot's sensor, the robot's visibility $\mathbf{V}(\mathbf{x}, \mathbf{a}, r, \alpha) \subset \mathbb{R}^3$ is a region defined as the continuous set of points [RDC05c]

$$\begin{aligned} \mathbf{V}(\mathbf{x}, \mathbf{a}, r, \alpha) = \{\mathbf{y} \in \mathbb{R}^3 : & \|\mathbf{y} - \mathbf{x}\| \leq r, \\ & 0 \leq \arccos\left(\frac{(\mathbf{y} - \mathbf{x}) \cdot \hat{\mathbf{p}}}{\|\mathbf{y} - \mathbf{x}\|}\right) \leq \alpha\}, \end{aligned} \tag{6.20}$$

wherein the robot's heading $\hat{\mathbf{p}}$ is given by equation (4.49), page 122. The operation $(\mathbf{y} - \mathbf{x}) \cdot \hat{\mathbf{p}}$ denotes the internal product of vectors $(\mathbf{y} - \mathbf{x})$ and $\hat{\mathbf{p}}$.

As Fig. 6.6 shows, the robot's visibility is both limited by the sensor itself and the environment surrounding it. Although the methods proposed in this thesis can be used with any type of range sensor, they were validated by using stereo-vision range sensors, whose sensing ability is both limited in distance and angle (see section 4.7.2, page 125). The volume delimited by the maximum and minimum distances that a stereo-range sensor can measure ([5]) — the horopter — depends on the stereo-rig's baseline and the selected parameters for the stereo correlation algorithm, namely the number of disparities and the horopter offset. The angular limitation is introduced by the sensor through its lens' focal length, which determines a wider or narrower visibility region around the normal vector $\hat{\mathbf{p}}$ to the image plane, through an angle α.

But, whether the robot is currently exploring a wide open area or a narrower space, the robot's visibility is also dynamically conditioned by the presence of obstacles in front of the sensor, which hide the space behind them and reduce the sensor's intrinsic range (see Fig. 6.6). For instance, if the robot was exploring a wide room and enters into a corridor, or if it gets closer to a wall, both r and α will likely be reduced, which will reduce the robot's visibility. Note, however, that while parameter r tends to vary significantly if the robot is moving towards an obstacle, parameter α remains almost constant, unless the robot moves laterally to an obstacle (*e.g.* if the robot moves along a wall, very close to it).

In order to dynamically adapt the robot's visibility with the surrounding environment, the latest data provided by the sensor can be used to estimate r and α [RDC05c]. Given the latest batch of measurements M_k, having m_k measurements, the robot's visibility parameters are estimated as:

$$\hat{r} = \frac{1}{m_k} \sum_{i=1}^{m_k} \| \overrightarrow{\mathbf{v}}_{k,i} \|, \tag{6.21}$$

$$\hat{\alpha} = \max_i \left[\arccos \left(\frac{\overrightarrow{\mathbf{v}}_{k,i} \cdot \hat{\mathbf{p}}}{\| \overrightarrow{\mathbf{v}}_{k,i} \|} \right) \right]. \tag{6.22}$$

Equation (6.21) computes the average distance and equation (6.22) computes the maximum angle with the sensor's heading along all measurements contained in the latest batch. Note that the flow diagram represented in Fig. 6.5, page 187, indicates that equations

[5] The minimum measurable distance is not considered explicitly herein, because it corresponds to open space (null coverage) between the sensor and its real visibility region. Therefore, by simplicity, that lower bound is neglected and it is assumed to be null.

(6.21) and (6.22) are used to update the visibility parameters r and α, whenever the robot acquires a new batch of measurements (see the output of the box "Get measurements from robot's sensor").

6.4.2 Visible map

Consider a fleet $\mathcal{F} = \{1, \ldots, n\}$ of n robots performing a volumetric mapping mission and one of the robots in the team $i \in \mathcal{F}$. The visibility $\mathbf{V}_i = \mathbf{V}(\mathbf{x}_i, \mathbf{a}_i, r_i, \alpha_i) \subset \mathbb{R}^3$ of this robot represents a sub region of the environment being mapped that robot i is able to sense and, thus, measurements gathered from its current pose $Y_i = (\mathbf{x}_i, \mathbf{a}_i)$ will only influence its knowledge about that sub region. Accordingly with the volumetric model proposed in chapter 4, which divides the volume being mapped into a set $\mathcal{Y}$ of equally sized voxels (see Fig. 4.2, page 102), that sub region refers to the subset of voxels

$$\mathcal{Z}^i = \{l \in \mathcal{Y} : \mathbf{w}(l) \in \mathbf{V}(\mathbf{x}_i, \mathbf{a}_i, r_i, \alpha_i)\} \subset \mathcal{Y}, \tag{6.23}$$

wherein $\mathbf{w}(l)$ denotes the center coordinates of a voxel $l \in \mathcal{Y}$.

The *robot's visible map* is the subset of coverage random variables

$$\mathcal{C}^i = \{C_l,\ l \in \mathcal{Z}^i\} \subset \mathcal{C}. \tag{6.24}$$

These random variables are described statistically by the subset of probability density functions

$$\mathcal{P}(\mathcal{C}^i) = \{p(c_l) : l \in \mathcal{Z}^i\},\ \mathcal{P}(\mathcal{C}^i) \subset \mathcal{P}(\mathcal{C}), \tag{6.25}$$

representing the robot's knowledge about the visible sub region covered by the set of voxels $\mathcal{Z}^i \subset \mathcal{Y}$. As in the case of equations (4.6) and (4.9), page 104, derived for the whole map, the joint probability density function of the robot's visible map is

$$p(\mathcal{C}^i) = \prod_{l \in \mathcal{Z}^i} p(c_l), \tag{6.26}$$

and the *joint entropy of the robot's visible map* is

$$H(\mathcal{C}^i) = \sum_{l \in \mathcal{Z}^i} H(C_l) < H(\mathcal{C}). \tag{6.27}$$

Equation (6.27) is a formal measure of the uncertainty covered by the sensor of robot i [RDC05c], wherein the inequality means that the robot's visible map covers only part of the uncertainty associated with the whole map.

The other robots in the fleet $\mathcal{F}\backslash i$ cover the set of voxels

$$\mathcal{W}^i = \bigcup_{j \in \mathcal{F}\backslash i} \mathcal{Z}^j \subseteq \mathcal{Y} \tag{6.28}$$

and have a joint visible map denoted as

$$\mathcal{T}^i = \bigcup_{j \in \mathcal{F}\backslash i} \mathcal{C}^j \subseteq \mathcal{C}, \tag{6.29}$$

whose joint entropy is given by

$$H(\mathcal{T}^i) = \sum_{l \in \mathcal{W}^i} H(C_l). \tag{6.30}$$

Equation (6.30) measures the uncertainty covered by the rest of the team [RDC05c].

The fleet covers the set of voxels

$$\mathcal{W} = \mathcal{Z}^i \cup \mathcal{W}^i \subseteq \mathcal{Y} \tag{6.31}$$

and has a joint visible map denoted as

$$\mathcal{T} = \mathcal{C}^i \cup \mathcal{T}^i \subseteq \mathcal{C}. \tag{6.32}$$

Accordingly with equation (6.15), page 189, the *joint entropy of the team's visible map* is given by

$$H(\mathcal{T}) = H(\mathcal{C}^i) + H(\mathcal{T}^i) - I(\mathcal{C}^i; \mathcal{T}^i) < H(\mathcal{C}), \tag{6.33}$$

which measures the uncertainty being covered by the whole team, which is just part of the uncertainty $H(\mathcal{C})$ associated with the whole map [RDC05c]. Since both sets of coverage random variables $\mathcal{C}^i$ and $\mathcal{T}^i$ are subsets of $\mathcal{C}$, and being $\mathcal{C}$ a set of statistically independent random variables containing one variable for each voxel of the grid $\mathcal{Y}$, the mutual information $I(\mathcal{C}^i; \mathcal{T}^i)$ between the robot's visible map and the joint visible map of the other robots cab be computed through equation (6.19), page 190. This mutual information is a formal measure of the overlapping between the robot's visible map and the joint visible map of the rest of the team. It is null if the robot's visible map does not overlap with the other robots' visible maps; otherwise, it is the sum of the entropy of the voxels belonging to the overlapping ([6]).

[6] Note that, theoretically, $I(\mathcal{C}^i; \mathcal{T}^i)$ may be also null when the visible maps overlap each other, if the uncertainty associated with the overlapping, *i.e.* the uncertainty associated with the intersection of the regions being sensed by the different robots, is null. Nevertheless, this is a quite unlikely situation, because robots use the exploration algorithm proposed in section 4.6.2, page 121, which seeks for regions with high uncertainty (entropy).

6.5 Multi-robot coordinated exploration strategy

In an exploration mission, the objective is to acquire as much new information about the environment as possible with every sensing cycle, *i.e.* to maximize the information gain [RDC05c]. Intuitively, this is equivalent to select new regions to explore so that the robot's sensor covers as much uncertainty as possible. For this reason, each single robot $i \in \mathcal{F}$ should maximize its visible map joint entropy $H(\mathcal{C}^i)$. This is indeed the goal of the entropy gradient-based exploration method proposed in section 4.6.2, page 121 [RDC05b, RDC05d, RDC05a].

However, due to the problem described in section 6.1.1, page 181, in the case of multiple robots, it is also crucial to minimize the mutual information due to the overlapping between the visible map of a given robot and the visible map of the rest of the team, because that situation yields a lower joint entropy for the team's visible map, and thus a less attainable information gain than in the non-overlapping case [RDC05c]. This conclusion can be obtained from equation (6.33), which is also schematically represented in Fig. 6.7 through an illustrative example, wherein the team's goal should be clearly to minimize the mutual information $I(\mathcal{C}^i;\mathcal{T}^i)$. Moreover, as the experiments reported in section 5.6, page 168 reveal, motion interference and mutual sensor occlusions are more likely to occur when robots explore nearby regions that eventually overlap each other, which leads to further performance degradation [RDC05b, RDC05a].

Therefore, with multiple robots, each robot $i \in \mathcal{F}$ should select a new exploration viewpoint in such a way so that the joint entropy of the team's visible map $H(\mathcal{T})$ is maximized. Accordingly with equation (6.33), this is a twofold objective [RDC05c]:

- Likewise in the single robot case (see section 4.6.2, page 121), to maximize the joint entropy of its own visible map $H(\mathcal{C}^i)$ ([7]);

- *And* to avoid the overlapping with the other robots' visible maps, so as to minimize the mutual information $I(\mathcal{C}^i;\mathcal{T}^i)$.

Furthermore, achieving successfully the former part of the objective with multiple robots also requires the minimization of interference phenomena related with not reachable exploration viewpoints (see section 6.1.2, page 182) and robots' mutual occlusions (see section 6.1.3, page 184). This section refines the exploration method proposed in section 4.6.2, page 121, for a single robot, with a coordination mechanism based on mutual-information minimization and interference minimization [RDC05c].

[7] If this behavior is common to all robots, the quantity $H(\mathcal{T}^i)$ will be also maximized.

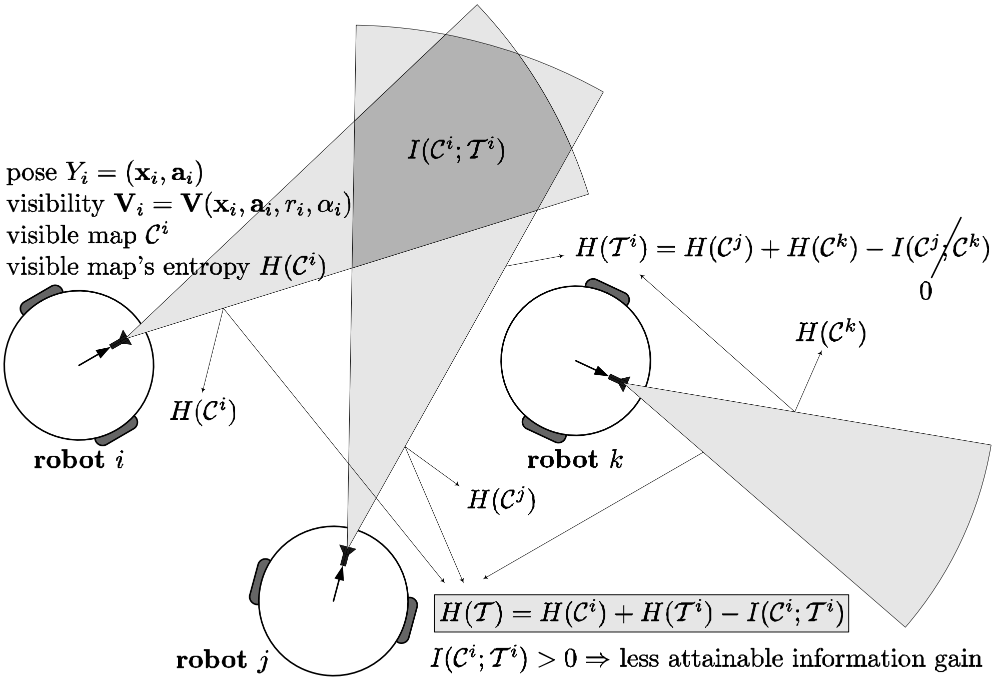

$I(\mathcal{C}^i;\mathcal{T}^i)$
pose $Y_i = (\mathbf{x}_i, \mathbf{a}_i)$
visibility $\mathbf{V}_i = \mathbf{V}(\mathbf{x}_i, \mathbf{a}_i, r_i, \alpha_i)$
visible map $\mathcal{C}^i$
visible map's entropy $H(\mathcal{C}^i)$
$H(\mathcal{T}^i) = H(\mathcal{C}^j) + H(\mathcal{C}^k) - \cancel{I(\mathcal{C}^j;\mathcal{C}^k)}$
0
$H(\mathcal{C}^k)$
$H(\mathcal{C}^i)$
robot i
robot k
$H(\mathcal{C}^j)$
$\boxed{H(\mathcal{T}) = H(\mathcal{C}^i) + H(\mathcal{T}^i) - I(\mathcal{C}^i;\mathcal{T}^i)}$
robot j
$I(\mathcal{C}^i;\mathcal{T}^i) > 0 \Rightarrow$ less attainable information gain

Figure 6.7: Example showing visible maps with 3 robots i, j and k. The mutual information $I(\mathcal{C}^i;\mathcal{T}^i) > 0$, decreases the team's visible map joint entropy $H(\mathcal{T})$, *i.e.* the team covers a smaller part of the map's uncertainty $H(\mathcal{C})$.

Considering a given robot $i \in \mathcal{F}$, the exploration method selects the best voxel from a subset of $\mathcal{Y}$ located in its neighborhood, by computing:

- The entropy gradient;
- The associated visible map's mutual information;
- A coefficient measuring how much it is reachable;
- An interference coefficient related with occlusions due to other robots.

Accordingly with the revised version of the distributed architecture model, which is presented in section 6.2, page 185, it is assumed that, whenever a given robot $j \in \mathcal{F}$ selects a new pose $Y_j^s = (\mathbf{x}_j^s, \mathbf{a}_j^s)$, all the other robots in the team $\mathcal{F}\backslash j$ are informed through direct communication about its new selected pose and its current range parameters r_j and α_j, *i.e.* the other robots receive the tuple (Y_j^s, r_j, α_j). This minimal communication enables each robot $i \in \mathcal{F}$ to implement the aforementioned exploration coordination mechanism. For instance, it provides the robot with the ability to compute the mutual information $I(\mathcal{C}^i;\mathcal{T}^i)$ between its visible map $\mathcal{C}^i$ and the joint visible map of the rest of the team $\mathcal{F}\backslash i$.

6.5.1 Measuring the visible maps' mutual information

Before a robot $i \in \mathcal{F}$ selects its new pose, it can compute the other robots' visibility through equation (6.20), because, as it was already mentioned, the robot knows (Y_j^s, r_j, α_j), $\forall_{j \in \mathcal{F} \setminus i}$. Then, using equations (6.28), (6.29) and (6.30), that robot can compute, respectively, the set $\mathcal{W}^i$ of visible voxels by the other robots, their joint visible map $\mathcal{T}^i$ and the joint entropy $H(\mathcal{T}^i)$. Note that the pose Y_j of a given robot $j \in \mathcal{F}$ can be different from its current selected pose Y_j^s, from the instant time it selects a new viewpoint until the instant time the new viewpoint is reached. During that interval, the robot is moving itself between the old viewpoint and the new selected viewpoint.

Consider the set of candidate voxels $\mathcal{N}_\Gamma(\mathbf{x}, r) \subset \mathcal{Y}$ that are intersected by the robot' sensor motion plane Γ and are located in the robot's neighborhood, which can be computed through equation (4.52), page 122, by taking $\varepsilon = r$ ([8]). Now consider a given voxel $l \in \mathcal{N}_\Gamma(\mathbf{x}, r)$ whose center $\mathbf{w}(l)$ is a candidate point to the robot's next selected position $\mathbf{x}^s$, being the new sensor's gaze $\mathbf{a}(l)$ determined through equation (4.54), page 123. Hereafter, this robot's pose is denoted as $Y^l = (\mathbf{w}(l), \mathbf{a}(l))$.

The current range parameters r and α of the robot's sensor define a visibility region $\mathbf{V}(\mathbf{w}(l), \mathbf{a}(l), r, \alpha)$, computed through equation (6.20). Using equations (6.23), (6.24) and (6.27), the robot computes the visible voxels $\mathcal{Z}^i(Y^l)$, the visible map $\mathcal{C}^i(Y^l)$ and the visible map's joint entropy $H(\mathcal{C}^i(Y^l))$, when its pose is Y^l. Then the mutual information $I(\mathcal{C}^i(Y^l); \mathcal{T}^i)$ between the visible map from that voxel and the other robots' joint visible map can be computed through equation (6.19), being equal to the joint entropy of the intersection voxels $\mathcal{Z}^i(Y^l) \cap \mathcal{W}^i$.

The non-redundancy coefficient for a candidate voxel is defined as the function $\lambda : \mathcal{Y} \rightarrow]0, 1]$, which is given by

$$\lambda(l) = \exp\left[-\frac{1}{\xi} I(\mathcal{C}^i(Y^l); \mathcal{T}^i)\right], \tag{6.34}$$

wherein ξ is a scale factor [RDC05c]. The coefficient $\lambda(l)$ decays exponentially with $I(\mathcal{C}^i(Y^l); \mathcal{T}^i)$, thus penalizing those viewpoints for which the mutual information, between the associated robot's visible map and the visible map of the other robots, is higher. Greater values of ξ increase the function's sensitivity to that mutual information.

[8] Taking $\varepsilon = r$ means that the robot's search space depends on whether it is exploring an open space or a narrower space. The parameter r is greater in the former case than in the latter case and, thus, it makes sense to have a wider search space in the former case, since the robot's sensor covers most likely a wider volume.

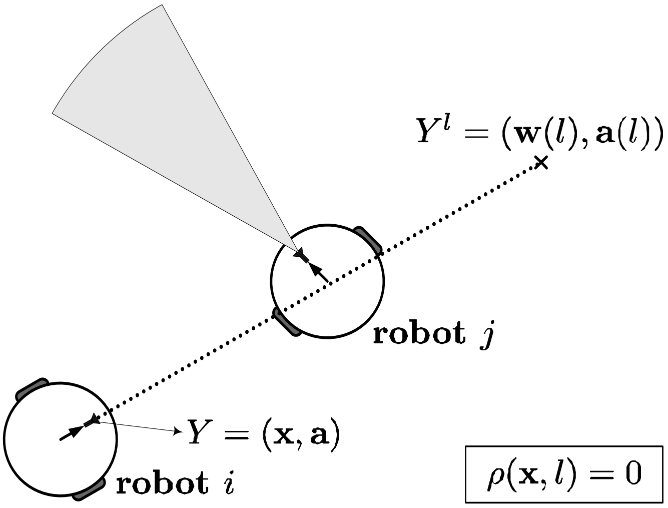

$Y^l = (\mathbf{w}(l), \mathbf{a}(l))$
robot j
$Y = (\mathbf{x}, \mathbf{a})$
robot i
$\rho(\mathbf{x}, l) = 0$

Figure 6.8: Measuring how reachable a potential exploration viewpoint is: in this example, given the current sensor's pose $Y = (\mathbf{x}, \mathbf{a})$ of robot i, the reachability $\rho(\mathbf{x}, l)$ of voxel l by robot i is null, because robot j is located in the path from pose Y to pose Y^l.

6.5.2 Measuring how reachable a potential exploration viewpoint is

Assuming by simplicity that the robot's path between two consecutive exploration viewpoints is a straight line, how reachable a given voxel l is — denoted hereafter as the *voxel's reachability* — depends of how much covered are the voxels traversed by the robot when it moves its sensor from the current pose Y to the pose Y^l. These voxels may be either occupied with obstacles in the environment or other robots.

Let $\mathcal{O}^k(Y) = \mathcal{O}^k(\mathbf{x}, \mathbf{a}) \subset \mathcal{Y}$ denote the set of voxels occupied by a robot $k \in \mathcal{F}$, when its sensor's pose is $Y = (\mathbf{x}, \mathbf{a})$. Being $\mathcal{Z}(\mathbf{x}, l) = \mathcal{Z}(\vec{\mathbf{u}}(\mathbf{x}, l), \mathbf{x})$ the set of voxels traversed by a vector connecting the current robot's position $\mathbf{x}$ to the center of voxel l, which can be computed through equation (4.16), page 107, the set of traversed voxels by the robot is

$$\mathcal{O}(Y, Y^l) = \left[\bigcup_{m \in \mathcal{Z}(\mathbf{x}, l)} \mathcal{O}^i(\mathbf{w}(m), \mathbf{a}(\mathbf{x}, m)) \right] \cup \mathcal{O}^i(Y^l), \tag{6.35}$$

wherein $\mathbf{a}(\mathbf{x}, m)$ is the robot's attitude associated with the direction of vector $\vec{\mathbf{u}}(\mathbf{x}, m)$. The set of voxels occupied by the rest of the team $j \in \mathcal{F} \backslash i$ is

$$\mathcal{O}^{\mathcal{F} \backslash i} = \bigcup_{j \in \mathcal{F} \backslash i} \mathcal{O}^j(Y_j^s). \tag{6.36}$$

Note that robot i is able to compute equation (6.36) because, accordingly with the distributed architecture described in section 6.2, page 185, it is aware of the other robots' selected viewpoints, *i.e.* it knows the quantity $Y_j^s = (\mathbf{x}_j^s, \mathbf{a}_j^s)$, $\forall_{j \in \mathcal{F} \backslash i}$.

The *reachability* of a voxel $l \in \mathcal{N}_\Gamma(\mathbf{x}, r)$, when the robot's sensor is moved from pose Y to pose Y^l, is defined as the function

$$\rho(\mathbf{x}, l) = \begin{cases} \min\limits_{m \in \mathcal{O}(Y, Y^l)} [1 - E(C_m)], & \mathcal{O}(Y, Y^l) \cap \mathcal{O}^{\mathcal{F} \backslash i} = \emptyset \\ 0, & \text{otherwise} \end{cases}, \tag{6.37}$$

taking values between 0 (invalid path) and 1 (path completely clear of obstacles and other robots) [RDC05c]. The second case of equation (6.37) occurs when there is at least one robot $j \in \mathcal{F} \backslash i$ in the path of robot i between its current sensor's position $\mathbf{x}$ and the center of voxel l (see Fig. 6.8 for an example).

6.5.3 Measuring the interference of other robots

The presence of other robots within the robot's visibility region yields undesirable partial occlusions and interference. Let denote a vector connecting the center of mass of a robot $i \in \mathcal{F}$, whose sensor's pose is Y_i^s, to the center of mass of another robot $j \in \mathcal{F}$, whose sensor's pose is Y_j^s, as $\overrightarrow{\mathbf{u}}(Y_i^s, Y_j^s)$. The *non-interference coefficient* of robot j over robot i, when the pose of robot i is equal to the candidate pose Y^l, is defined as the function $\eta : \mathcal{F} \times \mathcal{F} \rightarrow]0, 1]$, given by [RDC05c]

$$\eta(l, j) = \begin{cases} \frac{\|\hat{\mathbf{p}}(l) \times \overrightarrow{\mathbf{u}}(Y^l, Y_j^s)\|}{r}, & j \neq i, \ \mathcal{Z}^i(Y^l) \cap \mathcal{O}^j(Y_j^s) \neq \emptyset \\ 1, & \text{otherwise} \end{cases}, \tag{6.38}$$

wherein $(\times)$ denotes the external product of two vectors.

Recall that $Z^i(Y^l)$ denotes the set of visible voxels of robot $i \in \mathcal{F}$ from the candidate exploration viewpoint Y^l. The first case of equation (6.38) computes a non-occlusion ratio, when the other robot $j \in \mathcal{F} \backslash i$ belongs to the visible region of robot i. This ratio is smaller when the other robot is nearer to the position of robot i and when the angle formed by vectors $\hat{\mathbf{p}}(l)$ and $\overrightarrow{\mathbf{u}}(Y^l, Y_j^s)$ is smaller, *i.e.* when interference is more significant (see Fig. 6.9 for an example). The second case means no interference, which occurs when $j = i$, or when the other robot does not appear in the visibility region of robot i. Note that robot i is able to compute equation (6.38) because, accordingly with the distributed architecture described in section 6.2, page 185, it is aware of the other robots' selected viewpoints, *i.e.* it knows the quantity Y_j^s, $\forall_{j \in \mathcal{F} \backslash i}$.

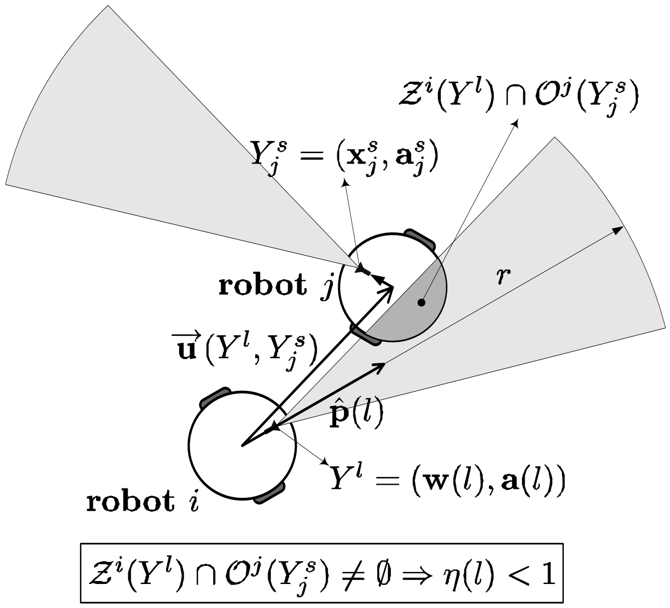

$\mathcal{Z}^i(Y^l) \cap \mathcal{O}^j(Y_j^s)$
$Y_j^s = (\mathbf{x}_j^s, \mathbf{a}_j^s)$
robot j
r
$\vec{\mathbf{u}}(Y^l, Y_j^s)$
$\hat{\mathbf{p}}(l)$
$Y^l = (\mathbf{w}(l), \mathbf{a}(l))$
robot i
$\mathcal{Z}^i(Y^l) \cap \mathcal{O}^j(Y_j^s) \neq \emptyset \Rightarrow \eta(l) < 1$

Figure 6.9: Measuring the interference of other robots due to partial occlusions of the robot's sensor: in this example, when the sensor's pose of robot i is $Y^l = (\mathbf{x}(l), \mathbf{a}(l))$, it is subject to an occlusion due to the presence of robot j in its visibility region $\mathcal{Z}^i(Y^l)$, which yields $\eta(l) < 1$ for its non-interference coefficient.

The *overall non-interference coefficient* for robot i is defined as the minimum of equation (6.38) along all the other robots in the team $j \in \mathcal{F} \backslash i$, *i.e.* it is determined upon the worst case of interference over robot i [RDC05c]. It is computed as

$$\eta(l) = \min_{j \in \mathcal{F} \backslash i} \eta(l, j). \tag{6.39}$$

6.5.4 Measuring the traveling cost

The three coefficients defined in previous sections are concerned with the utility of a given potential exploration viewpoint Y^l. Although the traveling cost is not included in the exploration method proposed in section 4.6.2, page 121, it may be important to consider it for some applications wherein, besides minimizing the mission execution time, is also important to minimize the energy spent on the mission by the robots.

Therefore, it may be worth reducing the traveled distance by the robot during exploration. This distance can be computed as

$$d_T = \sum_{k=1}^{k_{max}} \|\mathbf{x}_k - \mathbf{x}_{k-1}\|, \tag{6.40}$$

which accumulates the distance traveled by the robot between consecutive exploration positions ([9]). Note that if the k-th batch of measurements was received from another robot, *i.e.* if $u_k > 0$, we have $\mathbf{x}_k = \mathbf{x}_{k-1}$ and, thus, the batch's contribution to d_T is null. With the purpose of reducing d_T, the cost associated with each candidate voxel $l \in \mathcal{N}_\Gamma(\mathbf{x}, r)$ should be considered in the exploration strategy. This cost can be measured through the distance $\|\overrightarrow{\mathbf{u}}(\mathbf{x}, l)\|$ between the current robot's position $\mathbf{x}$ and the center of the candidate voxel l. The *cost coefficient* is thus defined as the function $\vartheta : \mathbb{R}^3 \times \mathcal{Y} \to [0, 1]$

$$\vartheta(\mathbf{x}, l) = \frac{\|\overrightarrow{\mathbf{u}}(\mathbf{x}, l)\|}{r}, \tag{6.41}$$

wherein the denominator's aim is to normalize the result between 0 and 1 [RDC05c].

6.5.5 Determination of the robot's next exploration viewpoint

Consider the projection on the robot's sensor motion plane Γ of the entropy gradient $\overrightarrow{\nabla} H_\Gamma(l)$ of each candidate voxel $l \in \mathcal{N}_\Gamma(\mathbf{x}, r)$, which is given by equation (4.53), page 123. Since the maximum value of discrete entropy is the number of bits b used in equation (4.8), page 104, the gradient magnitude $\left\|\overrightarrow{\nabla} H_\Gamma(l)\right\|$ can be normalized to the interval $[0, 1]$ [RDC05c] as

$$\left\|\overrightarrow{\nabla} H_\Gamma(l)\right\|_N = \frac{\epsilon}{\sqrt{2}\log_2 b}\left\|\overrightarrow{\nabla} H_\Gamma(l)\right\|. \tag{6.42}$$

Using the entropy-gradient criteria proposed in section 4.6.2, page 121, and the aforementioned coefficients, whose aim is either to coordinate the exploration with multiple robots or considering the traveling cost, given the search space $\mathcal{N}_\Gamma(\mathbf{x}, r)$, the robot's sensor is directed to the voxel

$$\boxed{l^s = \operatorname*{argmax}_{l \in \mathcal{N}_\Gamma(\mathbf{x}, r)} \left(\left\|\overrightarrow{\nabla} H_\Gamma(l)\right\|_N .\lambda(l).\rho(\mathbf{x}, l).\eta(l) - \kappa.\vartheta(\mathbf{x}, l)\right),} \tag{6.43}$$

with a gaze on arrival defined by the unitary vector $\hat{\mathbf{p}}(l^s)$, which is computed through equation (4.54), page 123, by taking $l = l^s$. In the argument of equation (6.43), the left term measures utility and the right term measures cost, being κ a cost sensitivity coefficient [RDC05c]. Note that both terms vary between 0 and 1 when $\kappa = 1$, which makes easy to balance their relative weight.

Equation (6.43) states that the robot is directed to a voxel located in the robot's neighborhood and in a frontier between more explored and less explored regions [RDC05c]. That voxel minimizes the mutual information with other robots' visible maps, is reachable,

[9] Recall that k_{max} is the index of the last batch of measurements obtained by the robot in the mission.

is likely not occluded by other robots and, depending on the value of κ, may restrict the traveled distance. If $\overrightarrow{\nabla} H_\Gamma(l^s) = \overrightarrow{0}$, the entropy gradient-based criteria is not conclusive and the robot should wander randomly until that condition is not verified.

As the center of the voxel l^s is likely unreachable, the robot is directed towards the projection of that point on the plane Γ computed through

$$\mathbf{x}^s = \operatorname*{proj}_{\Gamma} \mathbf{w}(l^s). \tag{6.44}$$

6.6 Implementation in mobile robots and in a computer simulator

In order to validate experimentally the coordination mechanism proposed in this chapter, the mobile robots described in section 4.7, page 123, were used to carry out some mapping experiments. Recall that those robots are differential drive robots equipped with stereovision, sonars and wireless communication (see Fig. 4.10, page 124). Based on the software architecture depicted in Fig. 5.8, page 168, whereby robots share useful measurements, a revised version was further developed to implement the coordinated exploration method represented by equation (6.43). The diagram of this new version is shown in Fig. 6.10.

The main difference to the previous version of the software, which was used in the experiments reported in section 5.6, page 168, to build maps through the uncoordinated version of the exploration method given by equation (4.55), page 123, is the implementation of the module `AWAREPROVID`. This module is aimed at providing the robot with the minimal communication required by the coordinated exploration method, so as to maintain a sufficient level of awareness in each robot $i \in \mathcal{F}$ about the exploration state of its teammates, *i.e.* the set of robots $\mathcal{F}\backslash i$. This requires the following information flow (see section 6.2, page 185):

- Receiving tuples (Y_j^s, r_j, α_j), whenever one of the other robots $j \in \mathcal{F}\backslash i$ selects a new exploration viewpoint (pose) $Y_j^s = (\mathbf{x}_j^s, \mathbf{a}_j^s)$, or whenever one of those robots updates its visibility parameters r_j and α_j.

- Sending to each robot $j \in \mathcal{F}\backslash i$ the robot's current selected viewpoint $Y_i^s = (\mathbf{x}_i^s, \mathbf{a}_i^s)$ and its most recent estimate about the visibility parameters r_i and α_i, whenever that information is updated by the other robot's software modules (*e.g.* the module `SURVCTRL` in Fig. 6.10).

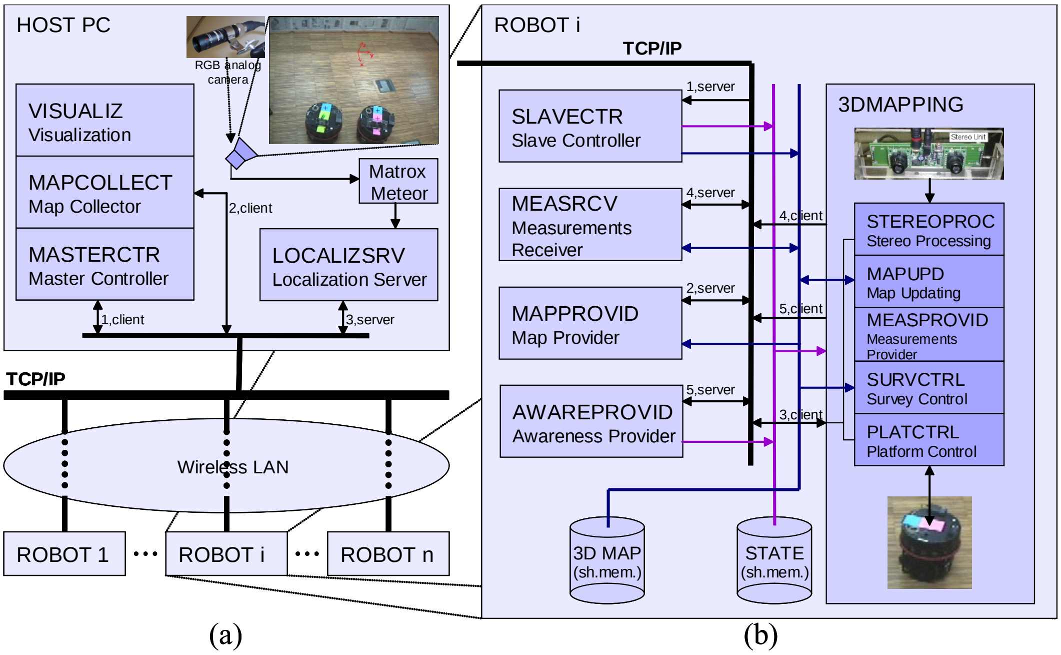


Figure 6.10: Diagram of the software architecture implementing the coordinated exploration method: (a) interaction between the team of robots and the host PC used for supervising 3-D mapping missions and localizing the robots through a global camera; (b) software modules running locally on each robot.

Besides implementing the software in mobile robots, a simulator was also built in Matlab©. The purpose of this simulator was twofold: firstly, to be able to easily predict the real robots' behavior through simulations ([10]); secondly, to be able to predict the behavior of teams with an arbitrary size and thus not being restricted to the maximum number of robots and their associated equipment (stereo vision and hardware related with wireless communication) that was available in the laboratory.

A snapshot of the simulator is presented in Fig. 6.11. It was programmed with enough detail, so as to allow to predict the mobile robots' behavior through computer simulations. It was used to easily extend the results obtained with mobile robots and arbitrary team sizes. The environment's features and size were the same both for the mobile robots and the simulator, and the robots' parameters (*e.g.* velocity, computation times, sensors' parameters, *etc.*) were also nearly the same. The comparison of similar experiments carried out with either mobile robots or the simulator validated indeed the results obtained in simulations.

[10] For example, a mapping mission that took about 2 hours could be simulated in just a few minutes with much less effort, though providing realistic results.

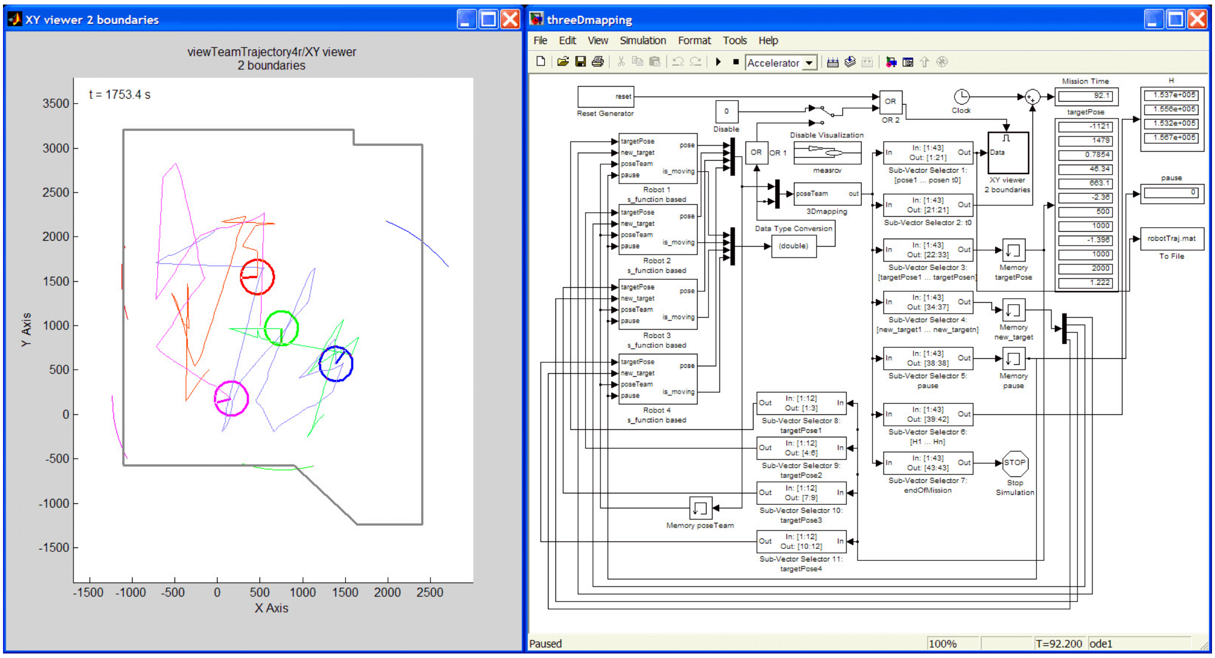


Figure 6.11: Matlab simulator that was used to predict the mobile robots' behavior with teams having an arbitrary number of robots.

6.7 Results

In order to compare the team's performance when using either the uncoordinated exploration method, given by equation (4.55), page 123, or the coordinated exploration method, given by equation (6.43), page 201, a set of experiments were carried out. All the experiments were performed within the *same* environment but with varying number of robots, in the interval $n \in \{1 \dots 10\}$.

A photo and a plan of the robots' workspace is shown in Fig. 4.18, page 137. Although the proposed coordinated exploration method was implemented in mobile robots (see Fig. 4.10, page 124), the previously validated computer simulator depicted in Fig. 6.11 was used to carry out more extensive experiments than those that were carried out with mobile robots, with several trials for each combination of the parameters, namely the team size n and the cost sensitivity coefficient κ.

6.7.1 Uncoordinated exploration versus coordinated exploration

Consider the execution time $t_{k_{max}}(n)$ for a mission with n robots. Based on the speedup measure [BA94] given by equation (2.1), page 35, let define the speedup measure for a team of n robots performing a volumetric mapping mission, as

$$\nu(n) = \frac{t_{k_{max}}(1)}{n.t_{k_{max}}(n)}. \tag{6.45}$$

Table 6.1 presents the results obtained in the aforementioned experiments, with the cost sensitivity coefficient κ set to 0, *i.e.* without restricting the robots' traveling distance between consecutive exploration viewpoints. The first column is the team size n. The second column presents the mean value (left) and the variation around the mean (right) of the mission execution time $t_{k_{max}}(n)$. The third column presents the mean value (left) and the variation around the mean (right) of the speedup measure $\nu(n)$. The fourth column compares the mission execution time for n robots with the time for a single robot, by presenting the mean value (left) and the variation around the mean (right) of the ratio $t_{k_{max}}(n)/t_{k_{max}}(1)$. The variation around the mean of these variables was estimated statistically by assuming a t-Student distribution and a confidence level equal to 95% ([11]).

The results contained in Table 6.1 are summarized in the graphs of Fig. 6.12. The mission execution time $t_{k_{max}}$ is compared on the top of Fig. 6.12 along different values of the team size n, using either the uncoordinated and the coordinated exploration methods.

Using the uncoordinated method, two robots took on average 81% of the time needed by a single robot, with a speedup equal to 0.62, though the team's performance became significantly worse for $n > 2$. The graph on the top of Fig. 6.12 shows that, for $n > 2$, adding more robots to the system always led to an increase of $t_{k_{max}}$. For $n > 5$, the average mission execution time was even greater than the time of a single robot, which is a disastrous performance. These results reveal that robots' interference and sensing overlapping, due to uncoordinated exploration, definitely compromised the potential benefit of cooperation through sharing sensory information, by using the cooperation strategy proposed in section 5.4, page 164. Moreover, this pernicious effect increased as long as the team size increased, especially with more than two robots.

On the other hand, using the coordinated method, two robots took on average only 59% of the time needed by a single robot, being the speedup slightly sub-linear and equal to 0.85. By adding more than two robots, and up to eight robots, *i.e.* for $3 \leq n \leq 8$, the mission execution always decreased on average, though yielding a degradation of $\nu(n)$, which was however slower than with the uncoordinated method.

For $n > 8$, the average mission execution time tended to increase with the team size, which indicates that using more than eight robots is completely worthless for the workspace considered in the experiments. This reveals that, although the coordinated

[11] See appendix F.3, page 274, for some background about confidence intervals.

Table 6.1: Results obtained within 3-D mapping experiments using both the uncoordinated and coordinated exploration methods.

n	$t_{k_{max}}(n)$	$[s]$	$\nu(n)$		$\frac{t_{k_{max}}(n)}{t_{k_{max}}(1)}$	
	average	*variation*	*average*	*variation*	*average*	*variation*
			Uncoordinated Exploration			
1	10782	586	1.00	-	1.00	-
2	8682	931	0.62	0.10	0.81	0.13
3	8826	1535	0.41	0.10	0.82	0.19
4	10187	2180	0.26	0.07	0.94	0.25
5	11341	705	0.19	0.02	1.05	0.12
6	12145	847	0.15	0.02	1.13	0.14
7	12848	2476	0.12	0.03	1.19	0.30
8	13897	3527	0.10	0.03	1.29	0.40
9	15852	3441	0.08	0.02	1.47	0.40
10	18451	2335	0.06	0.01	1.71	0.31
			Coordinated Exploration			
1	10782	586	1.00	-	1.00	-
2	6377	681	0.85	0.14	0.59	0.10
3	6134	432	0.59	0.07	0.57	0.07
4	5859	606	0.46	0.07	0.54	0.09
5	5390	1288	0.40	0.12	0.50	0.15
6	5082	326	0.35	0.04	0.47	0.06
7	4952	240	0.31	0.03	0.46	0.05
8	4642	415	0.29	0.04	0.43	0.06
9	5151	140	0.23	0.02	0.48	0.04
10	5376	1224	0.20	0.06	0.50	0.14

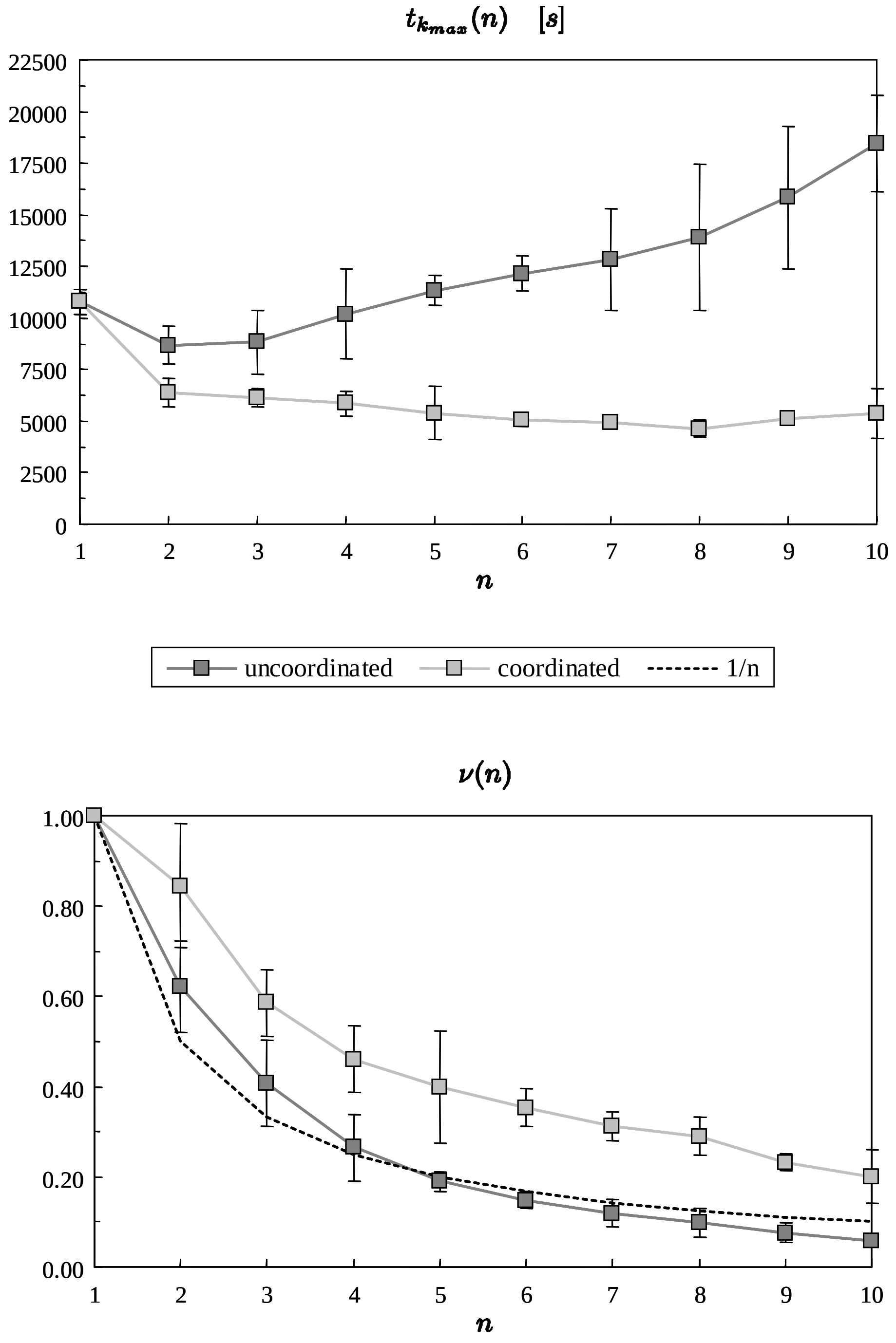


Figure 6.12: Robots' performance using both the uncoordinated and coordinated exploration methods: graph of the mission execution time $t_{k_{max}}$ (top) and graph of the speedup measure ν (bottom), as a function of the team size n.

method minimized the interference among robots, the benefit of having teams with more than two robots was not particularly noticeable, due to the relatively confined workspace where the experiments took place: an area having 5.3 m in width and 4.3 m in length.

This conclusion is particularly visible on the right of Fig. 6.12, wherein the speedup is plotted as a function of the team size. The dashed line represents the speedup frontier equal to $1/n$, wherein the mission execution time with n robots is equal to the time needed by a single robot. Using the uncoordinated method, a team with $2 \leq n \leq 4$ took on average less time to perform the mission than a single robot, but with a speedup noticeably poor. The coordinated method yielded a much more successful performance, since adding more robots always accelerated the mission execution until eight robots, with a speedup clearly above the single robot frontier $1/n$, though $\nu(n)$ tended to get worse with an increase in n.

6.7.1.1 Conclusions obtained through statistical tests of hypotheses

Aiming at better demonstrating the advantage of the coordinated method, when compared with its uncoordinated counterpart, a more rigorous statistical analysis of the data contained in Table 6.1 was carried out, by using the t-Student test of hypothesis for comparing the mean of two sampled populations ([12]).

Tests using a confidence level equal to 99% allowed to conclude that:

- The coordinated method yielded a faster mission execution time than the uncoordinated method for any team size ($n > 1$).
- Teams with two coordinated robots yielded a lesser mission execution time than teams with more than two uncoordinated robots.
- Teams with three coordinated robots always led to a lesser mission time than a single robot.
- Teams with two robots always performed faster than a single robot with both methods.

Although the last conclusion did not apply to teams containing more than to robots ($n > 2$), the coordinated method was able to reduce the mission execution time with less than eigth robots, whereas the uncoordinated version yielded worse execution times with more than two robots than with two robots.

[12] See appendix F.4.1, page 277, for some background about this statistical test of hypothesis.

Other tests using a lesser confidence level, equal to 95%, allowed to conclude that:

- Teams with two robots were faster than teams of five robots when using the uncoordinated method.

- Teams with two coordinated robots reduced the mission execution time by at least one hour when compared with a single robot.

Given a confidence level equal to 90%, a team with more than four coordinated robots were always faster than a team with two robots ([13]).

The detailed data about these statistical tests of hypotheses is presented in appendix F.5, page 280.

6.7.2 Impact of the cost sensitivity in the team's performance

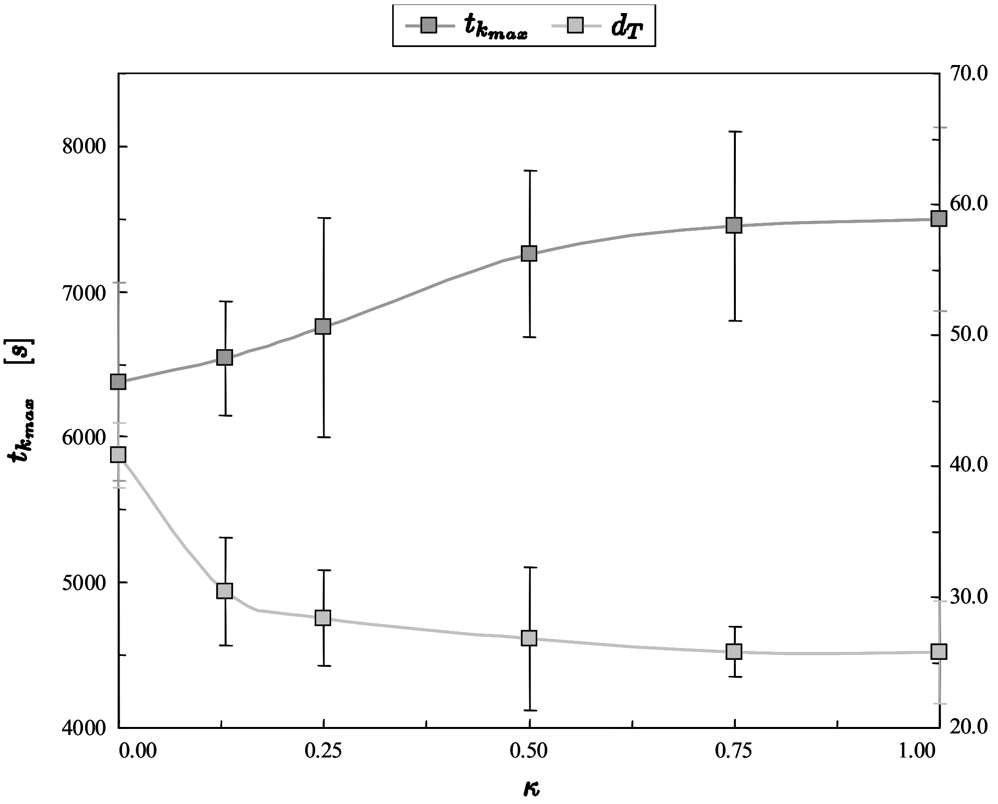
In order to evaluate how the cost sensitivity coefficient κ influences the team's performance, further experiments were carried out with two coordinated robots, using other values for that parameter than zero. Table 6.2 presents values for the mission time $t_{k_{max}}$, the traveled distance d_T given by equation (6.40), page 200, the speedup ν, and also the product of mission time and distance $t_{k_{max}}.d_T$ as a function of κ. Likewise in Table 6.1, the variation around the mean $(\hat{\Delta})$ was computed by assuming a t-Student distribution and a confidence level equal to 95% ([14]).

The most relevant data of Table 6.2 is plotted in the graphs of Fig. 6.13. The top graph shows that being more sensible to the traveled distance, *i.e.* increasing κ, leads to a monotonous increase of the mission execution time $t_{k_{max}}$ and to a monotonous decrease of the traveled distance d_T. Furthermore, these variations are not linear: the reduction of d_T is more noticeable in the interval $0 \leq \kappa \leq 0.25$; the increase of $t_{k_{max}}$ is more accelerated in the interval $0.125 \leq \kappa \leq 0.5$.

If both variables — mission execution time $t_{k_{max}}$ and traveled distance d_T — are required to be optimized, the goal can be expressed as the minimization of their product. The graph on the bottom of Fig. 6.13 shows a plot of the variables' product, which exhibits a minimum for roughly $\kappa = 0.5$. This global minimum means that distance can be significantly reduced without compromising too much the mission time for roughly $\kappa <$

[13] This conclusion is only valid up to the maximum team size that was simulated (10 robots). Since $t_{k_{max}}(n)$ started to increase at $n = 9$, the system would spend more time than a team of two robots for some value of n greater than 10.

[14] See appendix F.3, page 274, for some background about confidence intervals.



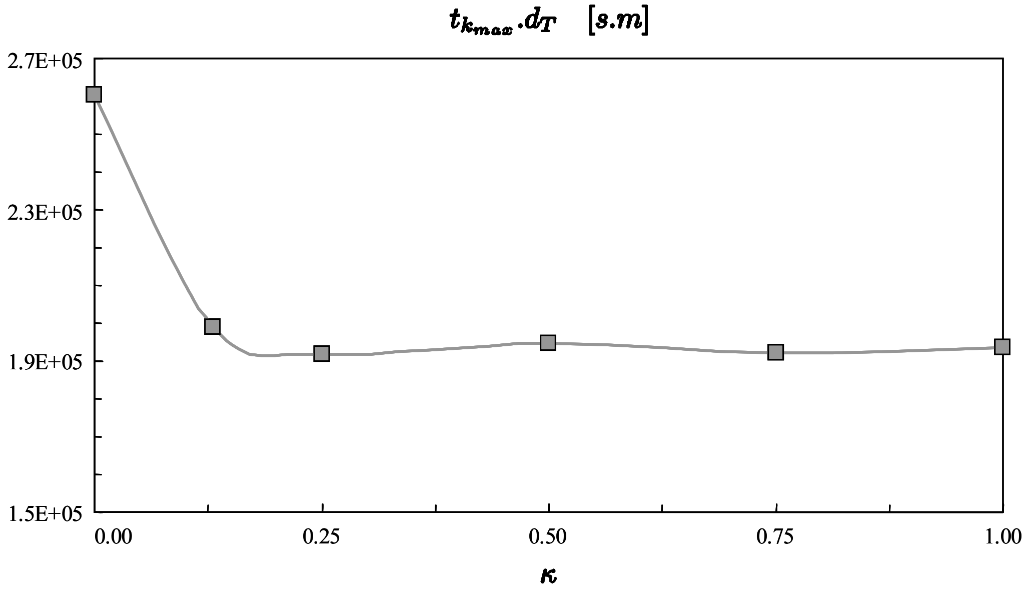


Figure 6.13: Performance of a team of two robots using the coordinated exploration method with different values for the cost sensitivity coefficient κ: graph of the mission execution time $t_{k_{max}}$ and of the traveled distance d_T (left) and graph of the product $t_{k_{max}}.d_T$ (right), as a function of κ.

Table 6.2: Results obtained with two coordinated robots using different values for the cost sensitivity coefficient κ.

κ	$t_{k_{max}}$	$[s]$	d_T	$[m]$	ν		$t_{k_{max}}.d_T$
	$\hat{\mu}$	$\hat{\Delta}$	$\hat{\mu}$	$\hat{\Delta}$	$\hat{\mu}$	$\hat{\Delta}$	$[10^5\ s.m]$
0	6377.2	518.2	40.8	1.9	0.85	0.11	2.602
1/8	6542.2	297.0	30.4	3.1	0.82	0.08	1.989
1/4	6752.6	572.9	28.4	2.8	0.80	0.11	1.918
1/2	7259.0	434.5	26.8	4.2	0.74	0.08	1.945
3/4	7447.4	494.0	25.8	1.4	0.72	0.08	1.921
1	7498.4	478.6	25.8	3.0	0.72	0.08	1.935

Symbols $\hat{\mu}$ and $\hat{\Delta}$ denote the estimated mean value and variation, respectively.

0.5. For greater values of κ, the product remains slightly constant, *i.e.* any reduction of distance is accompanied by an increase of mission time with the same order of magnitude. Note that for $\kappa > 0.75$, the changes are somewhat marginal.

6.8 Summary and discussion

When a team of several robots explore the same environment, undesirable phenomena are likely to occur in the absence of the coordination of their exploration actions, though they may cooperate through sharing useful information, by using the cooperation scheme proposed in chapter 5. This chapter discussed those phenomena and refined the entropy gradient based exploration method, proposed in chapter 4 for a single robot, with a suitable coordination mechanism. Its main goal is to take maximum advantage from robots' cooperation by reducing the interference among robots.

This chapter started by characterizing the problems that arise due to lack of coordination. These problems can be typified along three classes: robots sensing regions that overlap each other; constraining the robot's motion within the workspace, due to stopped robots that either block some trajectories or cause some regions to be unreachable; causing partial occlusions of the robot's sensor, due to the presence of other robots in front of it.

It was demonstrated that overcoming these problems require the robot to be sufficiently aware about the other robot's state. Therefore, the distributed architecture model for volumetric mapping, which was proposed in chapter 5, was extended so as to increase the level of awareness of each robot about the other robots' state, through minimal communication. The most important feature of the proposed coordinated exploration method is to avoid sensing overlapping regions with different robots. This problem was formulated as a mutual information minimization problem, after deriving mathematical expressions for computing the mutual information between sets of random variables.

In order to compare the team's performance when using both the coordinated and uncoordinated exploration methods, a set of volumetric mapping experiments was reported at the end of the chapter. These experiments used both mobile robots and computer simulations to compare both methods with different team sizes. Moreover, the influence of restricting the robots' traveled distance on the mission execution time was also studied.

The statistical analysis of experimental results demonstrated that coordinating the exploration significantly improves the team's performance, allowing to achieve a slightly sub-linear performance level, though the performance gain is more noticeable with two robots than with teams containing more robots. The most plausible explanation for the latter conclusion is the relatively confined dimensions of the workspace used in the experiments, wherein interference among robots is noticeably heavy. Nevertheless, further experiments carried out in larger workspaces, especially in outdoor environments, should be conducted in the future to better demonstrate the coordinated distributed architecture model proposed in this chapter, with teams of several robots.

Chapter 7

Conclusion

The essential goal of this thesis is to address the question of how to share efficiently information within a robotic system comprised of several cooperative robots. This final chapter summarizes the research reported in the previous chapters, concerning that research question. After a brief summary, it discusses the presented contributions, in the scope of their advantages and limitations, and points out perspectives on future research.

7.1 Summary

The context of this thesis is mobile robotics and, more specifically, multi-robot systems, *i.e.* robotic systems comprised of several cooperative robots that have usually the ability to move within the environment — mobile robots — and cooperate each other for the benefit of a common team's goal. Besides the overall robotics' motivation of building machines that may either substitute or assist humans in many real tasks, especially those tasks that are performed within hazardous or risky scenarios (*e.g.* search and rescue or planetary exploration), the motivation to study this family of human-made systems — cooperative multi-robot systems — are essentially based on the following potential advantages over a single robot solution: spatial distribution, parallelism, complex problems decomposition ("divide and conquer"), cost, reliability and robustness. Moreover, there are many tasks that either strictly require a team of robots due to their intrinsic characteristics (*e.g.* transporting a large object or surveillance of a wide area), or wherein a robotic collective may accomplish the mission with much better performance than a single robot (*e.g.* search for and collect items spread out along a wide area).

In spite of their advantages, cooperative multi-robot systems raise however some challenging problems that do not exist in a single robot solution, which must be properly solved in order to take real advantage from their potential. Some examples are: cooperative perception by fusing noisy observations from different robots; achieving coherent global behavior and performance with distributed control, in spite of individual robots having only a partial, sometimes inconsistent, knowledge about the task; cooperative planning by decomposing complex tasks and assigning subtasks to the individual robots; ensuring a coherent team behavior when unexpected events occur (*e.g.* robots failures, environmental disturbances, *etc.*).

This thesis addresses an issue that is transverse to many other design aspects of cooperative multi-robot systems: how to cope with the information distribution along different robots, especially when they follow a distributed control architecture. More specifically, what robots should communicate each other in order to ensure that each robot has a sufficient and consistent level of awareness. This allows the robot to cooperate in a purposive way with its teammates, so as to contribute to attain a team's global behavior that is suitable to the mission being performed.

In order to not loose the practical sense and maintain a reasonable level of abstraction in the pursued research, the problem is addressed in the specific context of building volumetric maps. This is a loosely-coupled task in the sense that it does not strictly require to be performed by more than one robot, though better performance may be attained with a multi-robot system. The task is indeed relevant to robotics, whether it is required to support other robotic operations or the purpose is the map itself.

In the former case, useful robotic tasks usually require that each robot has a representation model of the environment, in order to, for instance, be able to navigate safely and efficiently within the environment. In the latter case, there are some relevant target application wherein robots may substitute humans in the monotonous task of building detailed volumetric maps, especially in hazardous environments (*e.g.* buildings, buried utilities, gas pipes, abandoned mines, nuclear facilities, *etc.*).

Answering the main research question in the context of building volumetric maps raises other subsidiary research questions that are addressed along the thesis.

The first research question is concerned with representing the probabilistic knowledge about the map: the representation model. In this context, after comparing and discussing the possible alternatives — metric, feature-based and topological — a grid-based probabilistic volumetric model is proposed. The main reason that justify the choice of a metric representation is the ability to represent maps with fine detail, in a very intuitive way.

The main advantages of the proposed representation model are: the ability to explicitly represent the map's uncertainty through the mathematical definition of entropy; and a compact way of representing the belief about the state of a 3-D cell — a voxel — using an unimodal probability density function — a Gaussian distribution —, which can be described by *just* two parameters — its mean and its standard deviation. Due to this compactness, it is proven the effectiveness of the computational procedure for updating the voxel's belief upon new sensor measurements, which is based on very simple closed-form equations.

Another question related with the mapping process is how the robot should move within the environment, by choosing new exploration viewpoints based on its current map: the exploration problem. Since range sensors have limited range and yield noisy measurements, a robot has to survey the environment and build the map iteratively, so as to decrease gradually the map's uncertainty. The strategy should be to maximize the information gain in every sensing cycle, which, intuitively, is equivalent to use the range sensor to cover the most uncertain regions of the map.

An entropy-based formulation of the frontier-based exploration invented by Yamauchi [Yam98] is proposed for a single robot, whereby the robot's sensor is directed to frontier voxels between more explored and less explored regions, by searching voxels in the robot's neighborhood whose entropy gradient magnitude is maximum. Experiments with a mobile robot equipped with a stereo-vision system successfully validate the proposed exploration method. It is demonstrated experimentally that it allows to nicely reduce the map's uncertainty.

The aforementioned probabilistic framework is used afterwards to propose a distributed architecture model for volumetric mapping with a team of cooperative mobile robots equipped with a range sensor, whereby each robot works to improve its own map, while being altruistically committed to share useful sensory data with its teammates. In order to enable this cooperation driven by the utility of the shared information, a formal measure of information utility is developed. The utility of a range measurement is as high as the mutual information ([1]) between the current map and the new map obtained after updating the current map with that measurement.

The main research question is thus answered in the context of building volumetric maps: each robot should communicate to other robots sensory range data on the basis of the associated information utility. Experimental results obtained within experiments with two mobile robots equipped with stereo-vision demonstrated that, due to the proposed cooperation scheme, a team of robots accomplishes the mission in less time than a single

[1] As it is shown in chapter 3, mutual information is an entropy-based concept that measures the amount of information that a random variable has about another.

robot, if each robot is sufficiently selective when selecting useful sensory data to be sent to other robots.

Nevertheless, the reduction of the mission execution time due to the proposed information utility-based cooperation scheme is not so noticeable, because, besides sharing efficiently sensory data, each robot has also to be sufficiently aware of the other robots' state, so as to properly coordinate exploration actions and attain more effective cooperation. The lack of coordination yields undesirable phenomena, such as: situations wherein more than one robot senses the same map's region; or robots interfere each other, when the selected exploration viewpoint is not reachable or yields partial occlusions of the robot's sensor. The distributed architecture model is therefore refined with an exploration coordination mechanism, aiming at overcoming those problems.

The coordination method is based on additional minimal communication, which is required to let each robot know what are the selected exploration viewpoints by other robots. Its main feature is to formalize the sensing overlapping as a mutual information minimization problem. Each robot covers a set of voxels — visible map—, which in turn covers part of the map's uncertainty. The strategy is to minimize the map's uncertainty that is covered by more than one robot, *i.e.* to minimize the mutual information between the visible maps of different robots. Experimental results, obtained both with mobile robots and computer simulations, demonstrated a drastic reduction of the mission execution time due to cooperation, for teams of different sizes.

The contributions of the thesis include:

- A grid-based probabilistic model of a volumetric map, which enables to model explicitly uncertainty.

- An entropy gradient-based exploration strategy, which provides an entropy-based formulation of frontier-based exploration.

- An entropy-based measure of information utility, which robots use to select useful information to be communicated to other robots.

- A distributed architecture model for building volumetric maps with a team of cooperative mobile robots, in the absence of any centralized control.

- An exploration coordination mechanism based on the minimization of mutual information.

7.2 Discussion and directions for future research

The previous section summarized the research pursued within this thesis and presented its main contributions. Besides having advantages, the contributions of this thesis also have some limitations. After discussing these limitations, this section points out extensions and some directions for future research.

The most obvious limitation is concerned with using a metric representation model for the maps. Although providing fine detailed maps, it does not scale well with larger environments, because the map's number of cells (voxels) increases exponentially with the environment's dimension if the resolution remains constant. Nevertheless, this dimensionality problem is attenuated if the resolution of the map can be decreased by the same proportion as the environment's dimension is increased. This problem is however out of the scope of this thesis, because the most fundamental addressed question belongs to the context of how a team of mobile robots should be organized so as to perform mapping missions. Therefore, in this thesis, robotic mapping is just the application domain and not the main research purpose.

Metric maps, more specifically grid-based maps, are used herein as the representation model mainly because of its simplicity, as a means to concentrate the research effort, as much as possible, on the fundamental research question related with robots' cooperation. Moreover, because the environments' dimension has been relatively confined in the carried out experiments, the dimensionality problem has not restrained the work from obtaining useful conclusions regarding using multiple cooperative robots to build a map of an environment.

In spite of these attenuating aspects, addressing more carefully the dimensionality problem would be an important extension of the contributions proposed herein. An alternative would be, perhaps, to extend the current grid-based mapping framework with a hierarchy of representations [KB91]. This hierarchy would comprise both metric and topological representations, so as to be able to represent with high resolution local maps of more interesting regions, while being able to represent huge volumes through a higher level topological representation, whose nodes are metric maps. Another alternative would be to develop methods to extract features from dense metric data, in order to obtain a more compact and computationally efficient representation [LVH05].

A simple example of how the dimensionality problem can be attenuated with grid-based maps is to use more efficient data structures than arrays to represent a map in a computer, with less demanding memory requirements. In the course of the research reported herein, a linear memory array has been used to store the map in the computer's memory. Fig. 7.1-a shows how this structure was written in C programming language. Besides being static, this data structure is extremely inefficient concerning the required amount of memory, because it allocates a fixed array at the beginning of the program, which strictly contains one memory position for each voxel belonging to the defined volumetric grid $\mathcal{Y}$. This grid is macroscopically a parallelepiped which encompasses the whole environment being mapped (see Fig. 4.2-a, page 102).

This limitation can be better understood through an example. Fig. 7.2 depicts an example of a large-scale grid-based map with resolution $\epsilon = 0.5\ m$, which covers a volume equal to approximately $7 \times 10^3\ m^3$. The map encompasses two wider regions, which are connected by a narrower corridor.

Fig. 7.2-b shows that a significant part, equal to 23% of the volume of the blue parallelepiped encompassing the whole environment, cannot be explored, because the associated voxels are located behind the corridor's vertical walls. This situation is often encountered, because real environments rarely have the exact shape of a parallelepiped, being usually irregular. Moreover, because the exploration is gradual, there are more 25% of voxels not yet explored for the shown map's entropy level, though their state can be estimated with further exploration actions. This means that, for this example, 48% of the memory initially allocated in the computer for the map contains redundant data representing not explored voxels with maximum entropy; and approximately half of that memory (23% of the total memory) corresponds to voxels that simply will never be explored.

The previous example suggests the definition of a dynamic data structure, which divides the environment into strips. Assume, for instance, a partitioning into strips along the xx axis (orthogonal to the corridor). Each strip would be a contiguous list of voxels, whose size and first element's index could be dynamically managed.

An example of its definition in C programming language is shown in Fig. 7.1-b. Each strip along the xx axis — structure of type `StripMap` — contains the smallest list of *contiguous* voxels that encompasses every explored voxels in that strip (see an example on the right of Fig. 7.2-b). At the beginning of the mission, the map is represented as a two-dimensional array of void strips (field `n` set to zero), since every voxels are not explored. Whenever a voxel is explored for the first time, the array allocated to the associated strip (field `cells` in a structure of type `StripMap`) is dynamically re-allocated and re-arranged so as to encompass all the current explored voxels in the strip, while minimizing its size ([2]).

[2] The first element is always the explored voxel that has the smallest index along the yy axis (field `i1`

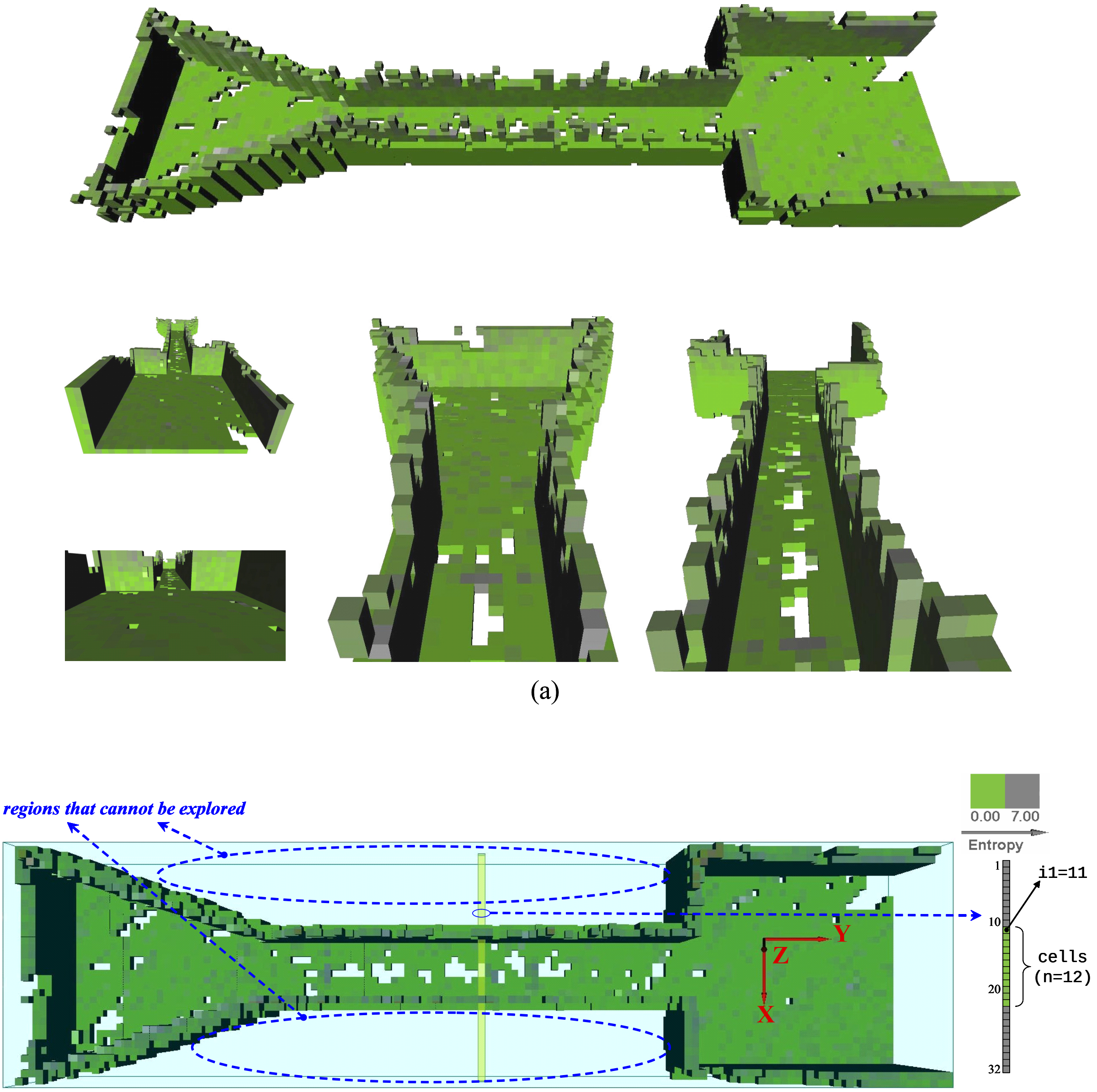
```
typedef struct s_MapCell{
  double coverage_mean; /* voxel's coverage mean */
  double coverage_std; /* voxel's coverage standard deviation */
} MapCell; /* a voxel */

const unsigned int nx = ...; /* number of voxels along xx axis */
const unsigned int ny = ...; /* number of voxels along yy axis */
const unsigned int nz = ...; /* number of voxels along zz axis */

MapCell map[nx * ny * nz]; /* the array containing the map */
```

(a)

```
typedef struct s_MapCell{
  double coverage_mean; /* voxel's coverage mean */
  double coverage_std; /* voxel's coverage standard deviation */
} MapCell; /* a voxel */

const unsigned int nx = ...; /* number of voxels along xx axis */
const unsigned int ny = ...; /* number of voxels along yy axis */
const unsigned int nz = ...; /* number of voxels along zz axis */

typedef struct s_StripMapCell{
  unsigned short int n; /* number of stored voxels for the strip */
  unsigned short int i1; /* first voxel's xx index (not used when n = 0) */
  MapCell *cells; /* array encompassing every explored voxels (0 <= n <= nx) */
} StripMap; /* a map strip along xx axis */

StripMap map[ny * nz]; /* the array containing the map */
```

(b)

Figure 7.1: Different data structures for storing the map in a computer: (a) static buffer allocated at the beginning of the program, having a memory position for each voxel; (b) a buffer of map strips, wherein each strip (in the example, a strip along the xx axis with fixed y and z coordinates) is a list of contiguous voxels represented as a dynamic buffer, whose size increases whenever new voxels are explored in the strip.



(a)

(b)

Figure 7.2: Example of a large-scale map obtained through a computer simulation: (a) different views of the map; (b) parallelepiped encompassing the whole environment and, on the right, an example of a strip along the xx axis, whose 12 explored voxels are stored in memory (all the voxels with entropy lesser than 7 bits).

For the map shown in Fig. 7.2, a static buffer requires 744 Kb of memory, while the dynamic data structure would require only 426 Kb, which represents a saving of 43% on the amount of memory, *i.e.* slightly lesser than the proportion of not explored voxels ([3]).

An important assumption of the proposed probabilistic model for the maps is to assume that random variables modeling the belief of different voxels are statistically independent. Since different cells may be updated upon the same range measurements, the belief of different voxels may present some statistical correlation and, thus, that assumption may be questionable from a theoretical point of view. Note however that, without the simplification, the probabilistic model's complexity would increase to such an extent that its practical usability would be certainly compromised. Moreover, the obtained results have demonstrated that the assumption is reasonable and not so unrealistic. Nevertheless, concerning this specific issue, a possible theoretical extension of the research would be to try to devise the equations that arise from relaxing that assumption, in order to formally evaluate its impact.

The sensor model, which is proposed to convert range measurements from a stereo-vision sensor into coverage estimates of the voxels, gives a special attention to noise in the range data, but it does neglect the bearing variance, *i.e.* noise in the measurements' angle. This noise is negligible for stereo-vision sensors or laser range scanners, whose main error is indeed present in distance information. But for other types of range sensors, whose bearing uncertainty is higher (*e.g.* sonars), the model is not realistic. Therefore, in order to properly model the latter range sensors, it would be worth including in the sensor model the bearing variance.

The problem of fusing sensory data in a common coordinates reference frame — the registration problem — is also out of the scope of this thesis, by assuming that robots are externally localized. In the performed experiments with mobile robots, a global localization scheme based on a global camera covering the robots' workspace is used. An absolute localization scheme could still be used in larger workspaces, including outdoors. Some examples of slightly more complex absolute localization schemes are: using a network of cameras inside a building, whose relative position is known *a priori*; detecting and measuring the robots' position relative to distinguishable landmarks whose localization is known *a priori*, a method which is known as trilateration; or using GPS — Global Positioning System — which uses a trilateration method wherein the landmarks are sub-

in structure `StripMap`), while the last element is the explored voxel with the highest value for that index (its index is `i1+n-1`).

[3] The comparison assumes that a variable of type `double` occupies 8 bytes, a variable of type `unsigned short int` occupies 2 bytes and a pointer occupies 4 bytes.

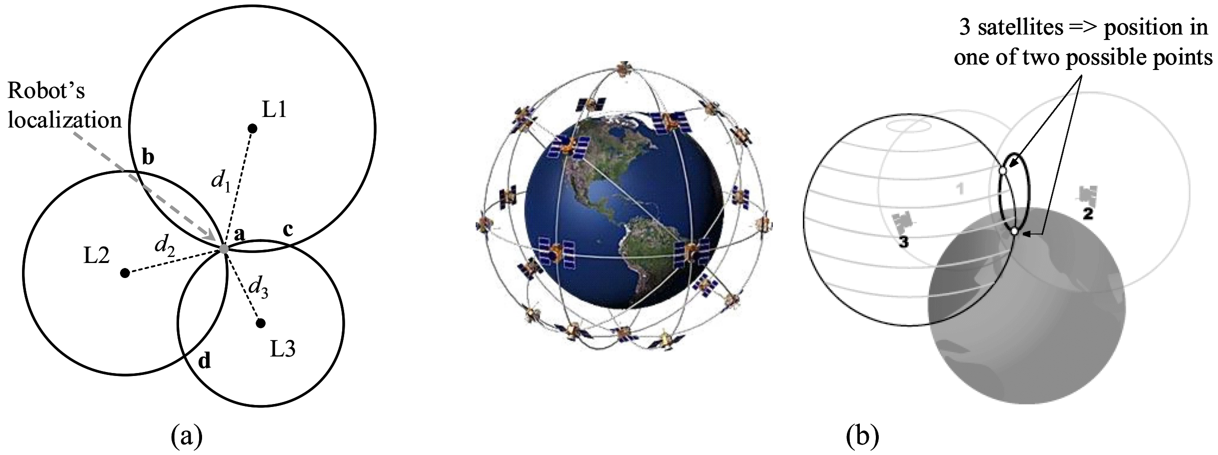
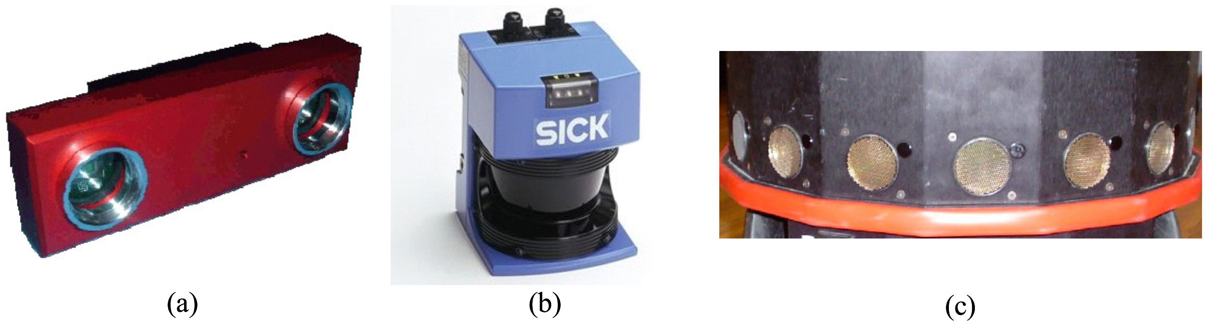


Figure 7.3: Absolute localization based on the trilateration principle. (a) The principle: if a robot is able to measure the distances to three distinguishable landmarks $L1$, $L2$ and $L3$, whose absolute localization is know *a priori*, it can determine the coordinates of the intersection point **a** where it is located. (b) The Global Positioning System — GPS — uses a constellation of 24 satellites in orbit above the Earth's surface to compute localization through the trilateration principle [Gar00].

stituted by a constellation of satellites. Fig. 7.3 illustrates the trilateration principle and shows an overview of GPS.

If these absolute localization methods are not viable or if the localization precision provided by any of these alternatives is not suitable for a given application, more complex and robust methods can be used, which fuse robot's odometry with data provided by one of those absolute sensors, in order to obtain a less uncertain localization estimate. An example are cooperative localization methods [MPS05, MR05, How05], which have been demonstrated to be more accurate than single robot localization methods [FBKT00]. Using relative observations, different robots can refine their internal beliefs based on the other robots' localization estimate and improve localization accuracy.

The aforementioned limitations are all concerned with the application domain that is used in the thesis: robotic mapping. Nevertheless there are also some limitations related with the main thesis' topic: cooperation and efficient information sharing. The main research question — how robots of a team should interact via communication in order to attain a suitable cooperative global behavior — is answered herein in the context of building volumetric maps. The proposed methods are validated through experiments with mobile robots in a relatively structured environment, inside a research laboratory. Although they have been sufficient to validate the approach and obtain important experimental results, it would be worth using it in more realistic scenarios, perhaps using teams



(a) (b) (c)

Figure 7.4: Different range sensors: (a) high resolution digital stereo-vision head STH-MDCS2 from Videre Design, with firewire interface and two 1.3 megapixel CMOS cameras [Vid05]; (b) laser scanner LMS 200-30106 from Sick AG, with 80 m maximum range, 180 deg field of view, 10 mm distance resolution and 0.25 deg angular resolution [Sic05]; (c) ring of 16 Polaroid 6500 sonar ranging modules mounted around the Scout robots shown in Fig. 4.10, page 124, with range 15 cm – 107 cm [Nom99].

comprised of many robots, not necessarily homogeneous, working in outdoors. This would require to overcome first the aforementioned problems related with representing the maps and the robots' robust localization.

An interesting extension would be also to use heterogeneous teams, having different sensory modalities. In fact, the approach proposed in this thesis is sufficient general to be used with any type of range sensors ([4]). It could still be used stereo-vision, due to its ability to easily provide dense sets of 3-D data, but it could also be used other range sensors, such as laser range scanners or ultrasonic sensors (sonars). Fig. 7.4 shows examples of these types of range sensors.

This heterogeneity could enhance the team's robustness against sensory limitations, by taking advantage from the complementary advantages of those types of range sensors. For instance, laser scanners would allow the team to sense poorly textured portions of the environment (*e.g.* flat white walls in a building), which cannot be properly sensed by stereo-vision. Although ultrasonic sensors present a lower precision, mainly due to the higher bearing variance, they can sense some structures that other types of sensors, based on light propagation, cannot (*e.g.* windowpanes in buildings, mirrors, *etc.*).

The insight provided by research reported herein could be used to tackle other relevant application domains of cooperative multi-robot systems and, thus, generalize the contributions herein proposed in the scope of building volumetric maps. This effort would

[4] The only component that, perhaps, would need to be refined to better model range sensors with different characteristics than stereo-vision sensors, would be the sensor model, especially for range sensors whose bearing variance is significant (*e.g.* sonars).

be particularly valuable for more complex tasks than robotic mapping, which require planning the mission and the decomposition of the task into sub-tasks that are afterwards dynamically assigned to the different robots. These issues — cooperative planning [BA00] and distributed dynamic task allocation [GM02] — are out of the scope of the thesis, since the mapping mission may be executed by a set of homogeneous mobile robots that do all the same: exploring the most uncertain regions of the map.

For instance, a slightly more complex collective robotic mission than building volumetric maps would be the surveillance of a building by a team of mobile robots. This kind of mission may be viewed as an extension of a robotic mapping mission, because the team still needs to build and update a map of the building, which serves as a common knowledge base that is used to coordinate the patrolling operations performed by the different robots. Besides cooperating to maintain an updated map of the environment, robots need also to cooperate so as to cover properly the environment and detect intruders as soon as possible.

The team's performance concerning the former aspect of the surveillance mission could still be measured through the map's entropy — lesser entropy means a better map — and the information utility measure proposed in this thesis remains directly applicable to the mission. But, concerning the latter aspect of the mission, another performance measure and information utility measure needs to be specified, so as to represent the uncertainty of existing an intruder in some region of the building. The word *uncertainty* allows to foresee that this latter performance component would be again measurable through an entropy, though using a different formulation. Intuitively, because that uncertainty increases gradually with the time elapsed since the last patrolling operation that covered the region being considered, this time seems to be also a relevant variable in measuring the team's performance.

Nevertheless, the information utility measure developed in this thesis for the mapping application domain is based on a more general concept also proposed in this thesis: given a specific multi-robot mission and its associated performance measure, a piece of information is as useful as it yields a high ratio between performance improvement and cost increasing. Although the information utility measure is unavoidably a concept that requires a specific formulation, tailored to the collective robotic mission being considered, that basic principle is sufficiently general to be applicable to any application domain. Moreover, the thesis provides important insights into entropy-based measures, which are unquestionably a manifestation of the usefulness of the mathematical definition of entropy for many situations wherein uncertainty needs to be quantified.

References and Bibliography

[AB98a] R. Arkin and T. Balch. Cooperative multiagent robotic systems. In D. Kortenkamp, R.P. Bonasso, and R. Murphy, editors, *Artificial Intelligence and Mobile Robots*. MIT Press, 1998.

[AB98b] R. Aylett and D. Barnes. A multi-robot architecture for planetary rovers. In *Proc. of 5th ESA Workshop on Space Robotics, ASTRA'98*, 1998.

[ABN93] R. Arkin, T. Balch, and E. Nitz. Communication of behavioral state in multi-agent retrieval tasks. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'93)*, volume 3, pages 588–594, 1993.

[AF01] T. Arbel and F. Ferrie. Entropy-based gaze planning. *Elsevier Journal of Image and Vision Computing*, 19:779–786, 2001.

[AFH+97] R. Alami, S. Fleury, M. Herrb, F. Ingrand, and S. Qutub. Operating a large fleet of mobile robots using plan-merging paradigm. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'97)*, pages 2312–2317, 1997.

[AFH+98] R. Alami, S. Fleury, M. Herrb, F. Ingrand, and F. Robert. Multi-robot cooperation in the MARTHA project. *IEEE Robotics and Automation Magazine*, 5(1):36–47, 1998.

[AHP05] Autonomous helicopter project. Robotics Institute, Carnegie Mellon University, U.S.A., Jul. 2005. [Online]. Available: http://www-2.cs.cmu.edu/afs/cs/project/chopper/www/index.html.

[AMI89] H. Asama, A. Matsumoto, and Y. Ishida. Design of an autonomous and distributed robot system: ACTRESS. In *Proc. of IEEE/RSJ Int. Workshop on Intelligent Robots and Systems (IROS'89)*, pages 283–290, Tsukuba, Japan, 1989.

[Ark89] R. Arkin. Motor schema based mobile robot navigation. *International Journal of Robotics Research*, 8(4):92–112, 1989.

[Ark92] R. Arkin. Cooperation without communication: Multi-agent schema based robot navigation. *Journal of Robotic Systems*, 9(3):351–364, 1992.

[AS97] K. Azarm and G. Schmidt. Conflict-free motion of multiple mobile robots based on decentralized motion planning and negotiation. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'97)*, volume 4, pages 3526–3533, 1997.

[Axe80a] R. Axelrod. Effective choice in the prisoner's dilemma. *Journal of Conflict Resolution*, 24:3–25, 1980.

[Axe80b] R. Axelrod. More effective choice in the prisoner's dilemma. *Journal of Conflict Resolution*, pages 379–403, 1980.

[Axe84] R. Axelrod. *The Evolution of Cooperation.* New York, Basic Books, 1984.

[BA94] T. Balch and R. Arkin. Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):27–52, 1994.

[BA98] T. Balch and R. Arkin. Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, 14(6):926–939, 1998.

[BA99] S. Botelho and R. Alami. M+: a scheme for multi-robot cooperation through negotiated task allocation and achievement. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'99)*, pages 1234–1239, 1999.

[BA00] S. Botelho and R. Alami. Robots that cooperatively enhance their plans. In *Proc. of 5th Int. Symp. on Distributed Autonomous Robotic Systems (DARS'2000)*, Knoxville, Tennessee, USA, 2000.

[Bal98] Tucker Balch. *Behavioral Diversity in Learning Robot Teams.* PhD thesis, Mobile Robot Laboratory, Georgia Institute of Technology, USA, 1998.

[Bal00] T. Balch. Hierarchical social entropy: An information measure of robot group diversity. *Autonomous Robots, Special Issue on Heterogeneous Multi-Robot Systems*, 8(3):209–237, Jun. 2000.

[Bal02] T. Balch. Taxonomies of multirobot task and reward. In T. Balch and L. Parker, editors, *Robot Teams: From Diversity to Polymorphism.* A. K. Peters Ltd., 2002.

[BB01] C. Boutilier and R. Brafman. Partial-order planning with concurrent interacting actions. *Journal of Artificial Intelligence Research*, 14:105–136, 2001.

[BB05] B. Burns and O. Brock. Single-query entropy-guided path planning. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2005)*, pages 2136–2141, Apr. 2005.

[BDF$^+$02] H.-D. Burkhard, D. Duhaut, M. Fujita, P. Lima, R. Murphy, and P. Rojas. The road to RoboCup 2050. *IEEE Robotics and Automation Magazine*, 9(2):31–38, Jun. 2002.

[BDG99] E. Bonabeau, M. Dorigo, and Théraulaz G. *Swarm intelligence: from natural to artificial systems.* Oxford University Press, 1999.

[Ben88] G. Beni. The concept of cellular robotic system. In *Proc. of IEEE Int. Symp. on Intelligent Control*, pages 57–62, 1988.

[BFT97] W. Burgard, D. Fox, and S. Thrun. Active mobile robot localization by entropy minimization. In *Proc. of* 2^{nd} *Euromicro Workshop on Advanced Mobile Robots (EUROBOT'97)*, 1997.

[BG00] E. Bonabeau and Théraulaz G. Swarm smarts. *Scientific American*, pages 72–79, Mar. 2000.

[BH00] T. Balch and M. Hybinette. Social potentials for scalable multi-robot formations. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2000)*, pages 73–80, 2000.

[BHD94] R. Beckers, O. Holland, and J. Deneubourg. From local actions to global tasks: Stigmergy and collective robotics. In *Proc. of Artificial Life IV.* MIT Press, 1994.

[BHK$^+$03] M. Buss, M. Hardt, J. Kiener, M. Sobotka, M. Stelzer, O. von Stryk, and D. Wollherr. Towards an autonomous, humanoid, and dynamically walking robot: modelling, optimal trajectory planning, hardware architecture, and experiments. In *Proc. of IEEE/RAS Int. Conf. on Humanoid Robots*, Karlsruhe, Germany, 2003.

[BK91] J. Borenstein and Y. Koren. The vector field histogram: fast obstacle avoidance for mobile robots. *IEEE Journal of Robotics and Automation*, 7(3):278–288, 1991.

[BMF$^+$00] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun. Collaborative multi-robot exploration. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2000)*, volume 1, pages 476–481, 2000.

[BMW$^+$02] F. Bourgault, A. Makarenko, S. Williams, B. Grocholsky, and H. Durrant-Whyte. Information based adaptive robotic exploration. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2002)*, pages 540–545, 2002.

[Bot00] Silvia Botelho. *Une Architecture Décisionnelle pour la Coopération Multi-Robots.* PhD thesis, LAAS/CNRS, Toulouse, France, 2000.

[Bou05] Jean-Yves Bouguet. Camera calibration toolbox for Matlab©, Feb. 2005. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc.

[BR05] E. Brunskill and N. Roy. SLAM using incremental probabilistic PCA and dimensionality reduction. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2005)*, pages 344–349, 2005.

[Bro86] R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation*, RA-2:14–23, 1986.

[BSH$^+$04] M. Beetz, T. Schmitt, R. Hanek, S. Buck, F. Stulp, D. Schröter, and B. Radig. The AGILO robot soccer team: experience-based learning and probabilistic reasoning in autonomous robot control. *Autonomous Robots*, 17(1):55–77, 2004.

[BSWL04] B. Braun, G. Starr, J. Wood, and R. Lumia. A framework for implementing cooperative motion on industrial controllers. *IEEE Trans. on Robotics and Automation*, 20(3):583–589, Jun. 2004.

[BVS02] L. Bernold, L. Venkatesan, and S. Suvarna. Multi-sensory approach to 3-D mapping of underground utilities. In *Proc. of Int. Symp. on Automation and*

Robotics in Construction, pages 525–530, NIST, Gaithersburg, Maryland, USA, 2002.

[Cas93] C. G. Cassandras. *Discrete Event Systems: Modeling and Performance Analysis.* Aksen Associates Inc. and Irwin Inc., 1993.

[CCL$^+$90] P. Caloud, W. Choi, J.C. Latombe, C. Le Pape, and M. Yim. Indoor automation with many mobile robots. In *Proc. of IEEE/RSJ Int. Workshop on Intelligent Robots and Systems (IROS'90)*, pages 67–72, Tsuchiura, Japan, 1990.

[CF04] H. Choset and D. Fox. The world of mapping. In *Proc. of WTEC Workshop on Review of United States Research in Robotics*, National Science Foundation (NSF), Arlington, Virginia, U.S.A., Jul. 2004.

[CFK97] Y. Cao, A. Fukunaga, and A. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4:1–23, 1997.

[Cho01] H. Choset. Coverage for robotics: a survey on recent results. *Annals of Mathematics and Artificial Intelligence, Kluwer*, 31:113–126, 2001.

[CKC04] L. Chaimowicz, V. Kumar, and M. Campos. A paradigm for dynamic coordination of multiple robots. *Autonomous Robots*, 17(1):7–21, 2004.

[CL85] R. Chatila and J.-P. Laumond. Position referencing and consistent world modeling for mobile robots. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'85)*, 1985.

[CL91] P. Cohen and H. Levesque. Teamwork. *Nous, Special Issue on Cognitive Science and Artificial Intelligence*, 25(4):487–512, 1991.

[CM03] J. Casper and R. Murphy. Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center. *IEEE Trans. on Syst., Man, Cybernetics B*, 33(3):367–385, Jun. 2003.

[CMKB04] J. Cortés, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Trans. on Robotics and Automation*, 20(2):243–255, Apr. 2004.

[CT91] T. Cover and J. Thomas. *Elements of Information Theory.* Wiley Series in Telecommunications. John Wiley & Sons, 1991.

[DFJN97] J. Doran, S. Franklin, N. Jennings, and T. J. Norman. On cooperation in multi-agent systems. *The Knowledge Engineering Review*, 12(3):309–314, 1997.

[DJM02] G. Dudek, M. Jenkin, and E. Milios. A taxonomy of multirobot systems. In T. Balch and L. Parker, editors, *Robot Teams: From Diversity to Polymorphism*. A. K. Peters Ltd., 2002.

[DMC96] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernatics - Part B: Cybernatics*, 26(1):1–13, 1996.

[DNC$^+$01] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans. on Robotics and Automation*, 17(3):229–241, 2001.

[DSO05] Projects of the dynamic systems and ocean robotics laboratory. Institute of Systems and Robotics, Lisbon, Jul. 2005. [Online]. Available: http://dsor.isr.ist.utl.pt/Projects/index.html.

[DTS$^+$04] M. Dorigo, V. Trianni, E. Sahin, R. Grob, T. Labella, G. Baldassarre, S. Nolfi, J.-L. Deneubourg, F. Mondada, D. Floreano, and L. Gambardella. Evolving self-organizing behaviors for a Swarm-Bot. *Autonomous Robots*, 17(2-3):223–245, 2004.

[EHBBB97] S. El-Hakim, P. Boulanger, F. Blais, and J. Beraldin. A system for indoor 3D mapping and virtual environments. In *Proc. of Videometrics V, SPIE vol. 3174*, pages 21–35, 1997.

[ER94] E. Ephrati and J. Rosenschein. Divide and conquer in multi-agent planning. In *Proc. of National Conf. on Artificial Intelligence (AAAI'94)*, pages 375–380, U.S.A., 1994.

[ER95] E. Ephrati and J. Rosenschein. A tractable heuristic that maximizes global utility through local plan combination. In *Proc. of 1st Int. Conf. on Multi-Agent Systems*, 1995.

[FBKT00] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. A probabilistic approach to collaborative multi-robot localization. *Autonomous Robots*, 8(3):325–344, 2000.

[FIN04] A. Farinelli, L. Iocchi, and D. Nardi. Multirobot systems: a classification focused on coordination. *IEEE Trans. on Syst., Man, Cybernetics B*, 34(5):2015–2028, 2004.

[FJC05] J. Folkesson, P. Jensfelt, and H. Christensen. Vision SLAM in the measurement subspace. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2005)*, pages 30–35, 2005.

[FLS02] J. Feddema, C. Lewis, and D. Schoenwald. Decentralized control of cooperative robotic vehicles: theory and application. *IEEE Trans. on Robotics and Automation*, 18(5):852–864, Oct. 2002.

[FM02] J. Fredslund and M. Matarić. A general algorithm for robot formations using local sensing and minimal communication. *IEEE Trans. on Robotics and Automation*, 18(5):837–846, Oct. 2002.

[FMP02] K. Fregene, R. Madhavan, and L. Parker. Incremental multiagent robotic mapping of outdoor terrains. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2002)*, pages 1339–1346, 2002.

[FS94] T. Fukuda and K. Sekiyama. Communication reduction with risk estimate for multiple robotic system. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 2864–2869, 1994.

[Gar00] Garmin Corporation. GPS guide for beginners, 2000. [Online]. Available: http://www.garmin.com/aboutGPS/manual.html.

[GM02] B. Gerkey and M. Matarić. Sold!: auction methods for multirobot coordination. *IEEE Trans. on Robotics and Automation*, 18:758–768, Oct. 2002.

[GMDW03] B. Grocholsky, A. Makarenko, and H. Durrant-Whyte. Information-theoretic coordinated control of multiple sensor platforms. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2003)*, pages 1521–1526, 2003.

[GMGH04] M. Ghaffari, D. Manthena, A. Ghaffari, and E. Hall. Mines and human casualties: a robotics approach toward mine clearing. In *Proc. of SPIE Intelligent Robots and Computer Vision XXI: Algorithms, Techniques, and Active Vision*, volume 5608, Philadelphia, USA, Oct. 2004.

[H2R05] Kawada Industries. Humanoid robot H2-R protoype, Jul. 2005. [Online]. Available: http://www.kawada.co.jp/ams/hrp-2/index_e.html.

[HB91] S. Hackwood and G. Beni. Self-organization sensors by deterministic annealing. In *Proc. of IEEE/RSJ Int. Workshop on Intelligent Robots and Systems (IROS'91)*, pages 1177–1183, 1991.

[HB92] S. Hackwood and G. Beni. Self-organization of sensors for swarm intelligence. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'92)*, pages 819–829, 1992.

[HBFT03] D. Hähnel, W. Burgard, D. Fox, and S. Thrun. An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2003)*, 2003.

[HJSL04] E. Hygounenc, I. Jung, P. Soueres, and S. Lacroix. The autonomous blimp project of LAAS-CNRS: achievements in flight control and terrain mapping. *Int. Journal of Robotics Research*, 23:473–511, 2004.

[Hon05] Honda. The Honda humanoid robot ASIMO, Jul. 2005. [Online]. Available: http://asimo.honda.com.

[How05] A. Howard. Multi-robot simultaneous localization and mapping using acausal particle filters. In *Proc. of Robotics: Science and Systems I*, Massachusetts Institute of Technology, U.S.A., June 8-11, 2005.

[HPS04] A. Howard, L. Parker, and G. Sukhatme. The SDR experience: experiments with a large-scale heterogeneous mobile robot team. In *Proc. of Int. Symp. on Experimental Robotics*, Singapore, 2004.

[HRW$^+$03] F. Hundelshausen, R. Rojas, F. Wiesel, E. Cuevas, D. Zaldivar, and K. Gunarsson. FU-Fighters team description 2003. In *Proc. of RoboCup International Symposium*, 2003.

[HTOA$^+$04] T. Huntsberger, A. Trebi-Ollennu, H. Aghazarian, P. Schenker, P. Pirjanian, and H. Nayar. Distributed control of multi-robot systems engaged in tightly coupled tasks. *Autonomous Robots*, 17(1):79–92, 2004.

[HV03] D. Huber and N. Vandapel. Automatic 3D underground mine mapping. In *Proc of the 4^{th} Int. Conf. on Field and Service Robotics*, July 2003.

[Jen95] N. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence*, 75:195–240, 1995.

[Jen96] N. Jennings. Coordination techniques for distributed artificial intelligence. In G. O'Hare and N. Jennings, editors, *Foundations of Distributed Artificial Intelligence*, pages 187–210. John Wiley & Sons, New York, 1996.

[JLB94] K. Jin, P. Liang, and G. Beni. Stability of synchronized control of discrete swarm structures. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'94)*, pages 1033–1038, 1994.

[Jun98] David Jung. *An Architecture for Cooperation among Autonomous Agents.* PhD thesis, Dep. of Computer Science, Univ. of Wollongong, Australia, 1998.

[JZ00] D. Jung and A. Zelinsky. Grounded symbolic communication between heterogeneous cooperating robots. *Autonomous Robots, Special Issue on Heterogeneous Multi-Robot Systems*, 8(3):269–292, Jun. 2000.

[KANM98] H. Kitano, M. Asada, I. Noda, and H. Matsubara. RoboCup: Robot world cup. *IEEE Robotics and Automation Magazine*, 5(3):30–36, 1998.

[KB91] B. Kuipers and Y.-T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, 8:47–63, 1991.

[KB00] C. Kube and E. Bonabeau. Cooperative transport by ants and robots. *Robotics and Autonomous Systems*, 30(1-2):85–101, 2000.

[KB02] Kurt Konolige and David Beymer. *SRI Small Vision System user's manual.* SRI International, Apr. 2002. Software version 2.3.

[KFO$^+$04] K. Konolige, D. Fox, C. Ortiz, A. Agno, M. Eriksen, B. Limketkai, J. Ko, B. Morisset, D. Schutz, B. Stewart, and R. Vincet. Centibots: very large scale distributed robotic teams. In *Proc. of Int. Symp. on Experimental Robotics*, Singapore, 2004.

[Kha86] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, 1986.

[KK92] J. Kapur and H. Kesavan. *Entropy Optimization Principles with Applications.* Academic Press, 1992.

[Kor92] R. Korf. A simple solution to pursuit games. In *Proc. of 11th Int. Workshop on Distributed Artificial Intelligence*, pages 183–194, 1992.

[KSF$^+$03] J. Ko, B. Stewart, D. Fox, K. Konolige, and B. Limketkai. A practical, decision-theoretic approach to multi-robot mapping and exploration. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2003)*, pages 3232–3238, 2003.

[KT01] H. Kitano and S. Tadokoro. RoboCup Rescue: a grand challenge for multi-agent and intelligent systems. *Artificial Intelligence Magazine*, 22(1):39–52, 2001.

[KZ94] C. Kube and H. Zhang. Collective robotics: From social insects to robots. *Adaptive Behavior*, 2(2):189–219, 1994.

[LAD02] J. Lobo, L. Almeida, and J. Dias. Segmentation of dense depth maps using inertial data: a real-time implementation. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2002)*, pages 92–97, Oct. 2002.

[LDWC92] J. Leonard, H. Durran-Whyte, and I. Cox. Dynamic map building for an autonomous mobile robot. *Int. Journal of Robotics Research*, 11(4):286–298, 1992.

[LH97] F.C. Lin and J.Y. Hsu. Cooperation protocols in multi-agent robotic systems. *Autonomous Robots*, 4:175–198, 1997.

[Lob02] Jorge Lobo. Inertial sensor data integration in computer vision systems. Master's thesis, Dep. of Electrical and Computer Engineering, University of Coimbra, Apr. 2002.

[LVAC99] P. Lima, R. Ventura, P. Aparício, and L. Custódio. A functional architecture for a team of fully autonomous cooperative robots. In M. Veloso and H. Kitano, editors, *RoboCup'99: Robot Soccer World Cup III.* Lecture Notes in Computer Science, Springer-Verlag, 1999.

[LVH05] J.-F. Lalonde, N. Vandapel, and M. Hebert. Data structure for efficient processing in 3-D. In *Proc. of Robotics: Science and Systems I*, Massachusetts Institute of Technology, U.S.A., June 8-11, 2005.

[Mat92a] M. Matarić. Behavior-based systems: Main properties and implications. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'92), Workshop on Architectures for Intelligent Control Systems*, pages 46–54, Nice, France, 1992.

[Mat92b] M. Matarić. Minimizing complexity in controlling a mobile robot population. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'92)*, pages 830–835, 1992.

[Mat93] M. Matarić. Kin recognition, similarity and group behavior. In *Proc. of 15th Annual Cognitive Science Conference*, pages 705–710, Boulder, Colorado, USA, 1993. Lawrence Erlbaum Associates.

[Mat94] Maja Matarić. *Interaction and Intelligent Behavior.* PhD thesis, Dep. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, USA, 1994.

[Mat95] M. Matarić. Designing and understanding adaptive group behavior. *Adaptive Behavior*, 4(1):51–80, 1995.

[Mat99] M. Matarić. Behavior-based robotics. In R. Wilson and F. Keil, editors, *MIT Encyclopedia of Cognitive Sciences*, pages 74–77. MIT Press, 1999.

[MCdDO05] L. Merino, F. Caballero, J. de Dios, and A. Ollero. Cooperative fire detection using unmanned aerial vehicles. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2005)*, pages 1896–1901, Barcelona, Spain, Apr. 2005.

[MDDW98] R. Madhavan, G. Dissanayake, and H. Durrant-Whyte. Map-building and map-based localization in an underground-mine by statistical pattern matching. In *Proc. of Int. Conf. on Pattern Recognition*, 1998.

[ME85] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'85)*, 1985.

[MFP04] R. Madhavan, K. Fregene, and L. Parker. Distributed cooperative outdoor multirobot localization and mapping. *Autonomous Robots*, 17(1):23–39, 2004.

[MLT$^+$02] R. Murphy, C. Lisetti, R. Tardif, L. Irish, and A. Gage. Emotion-based control of cooperative heterogeneous mobile robots. *IEEE Trans. on Robotics and Automation*, 18(5):744–757, 2002.

[MM96] M. Martin and H. Moravec. Robot evidence grids. Technical Report CMU-RI-TR-96-06, Robotics Institute, CMU, Pittsburgh, USA, 1996.

[MMO+98] M. Maimone, L. Matthies, J. Osborn, E. Rollins, J. Teza, and S. Thayer. A photo-realistic 3-D mapping system for extreme nuclear environments: Chernobyl. In *Proc. of IEEE/RSJ Int. Workshop on Intelligent Robots and Systems (IROS'98)*, volume 3, pages 1521–1527, 1998.

[MNS95] M. Matarić, M. Nilsson, and K. Simsarian. Cooperative multi-robot box-pushing. In *Proc. of IEEE/RSJ Int. Workshop on Intelligent Robots and Systems (IROS'95)*, volume 3, pages 556–561, 1995.

[MPG+04] F. Mondada, G. Pettinaro, A. Guignard, I. Kwee, D. Floreano, J.-L. Deneubourg, S. Nolfi, L. Gambardella, and M. Dorigo. Swarm-Bot: a new distributed robotic concept. *Autonomous Robots*, 17(2-3):193–221, 2004.

[MPS05] A. Martinelli, F. Pont, and R. Siegwart. Multi-robot localization using relative observations. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2005)*, pages 2808–2813, 2005.

[MR05] A. Mourikis and S. Roumeliotis. Performance bounds for cooperative simultaneous localization and mapping (C-SLAM). In *Proc. of Robotics: Science and Systems I*, Massachusetts Institute of Technology, U.S.A., June 8-11, 2005.

[MRLD05] J. Martins, R. Rocha, J. Lobo, and J. Dias. RAC robotic soccer small-size team: control architecture and global vision. In *Proc. of Scientific Meeting of Robótica'2005 National Robotics Festival*, pages 136–143, Coimbra, Portugal, Apr. 29, 2005.

[MS01] M. Matarić and G. Sukhatme. Task-allocation and coordination of multiple robots to planetary exploration. In *Proc. of 10th Int. Conf. on Advanced Robotics*, 2001.

[MVB04] S. Monteiro, M. Vaz, and E. Bicho. Attractor dynamics generates robot formations: from theory to implementation. In *Proc. of Int. Conf. on Robotics and Automation (ICRA'2004)*, pages 2582–2587, Apr. 2004.

[NAS05] NASA. Photo gallery of the Jet Propulsion Laboratory, California Institute of Technology, Jul. 2005. [Online]. Available: http://www.jpl.nasa.gov.

[NMS05] V. Nguyen, A. Martinelli, and R. Siegwart. Handling the inconsistency of relative map filter. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2005)*, pages 651–656, 2005.

[Nom99] Nomadic Technologies, Inc. *Nomad Scout user's manual*, Jul. 1999. v2.7.

[NSH03] A. Nuchter, H. Surmann, and J. Hertzberg. Planning robot motion for 3D digitalization of indoor environments. In *Proc. of 11th Int. Conf. on Advanced Robotics*, pages 222–227, 2003.

[ÖEH02] P. Ögren, M. Egerstedt, and X. Hu. A control Lyapunov function approach to multiagent coordination. *IEEE Trans. on Robotics and Automation*, 18(5):847–851, Oct. 2002.

[OJ96] G. O'Hare and N. Jennings, editors. *Foundations of Distributed Artificial Intelligence*. Sixth Generation Computer Technology Series. John Wiley & Sons, New York, 1996.

[OMW$^+$04] C. Olson, L. Matthies, J. Wright, R. Li, and K. Di. Visual terrain mapping for mars exploration. In *Proc. of IEEE Aerospace Conference*, volume 2, pages 762–771, Mar. 2004.

[Pap91] A. Papoulis. *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, Inc., 3rd edition, 1991.

[Par93] L. Parker. Designing control laws for cooperative agent teams. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'93)*, volume 3, pages 582–587, 1993.

[Par94] Lynne Parker. *Heterogeneous Multi-Robot Cooperation*. PhD thesis, Dep. of Electrical Eng. and Computer Science, MIT, USA, 1994.

[Par95] L. Parker. The effect of action recognition and robot awareness in cooperative robotic teams. In *Proc. of IEEE/RSJ Int. Workshop on Intelligent Robots and Systems (IROS'95)*, pages 212–219, 1995.

[Par98] L. Parker. ALLIANCE: An architecture for fault-tolerant multi-robot cooperation. *IEEE Trans. on Robotics and Automation*, 14(2):220–240, 1998.

[Par99] L. Parker. Cooperative robotics for multi-target observation. *Intelligent Automation and Soft Computing*, 5(1):5–19, 1999.

[Par00] L. Parker. Current state of the art in distributed autonomous mobile robotics. In G. Bekey and J. Barhen, editors, *Distributed Autonomous Robotic Systems 4*. Springer-Verlag, 2000.

[Par02] L. Parker. Distributed algorithms for multi-robot observation of multiple moving targets. *Autonomous Robots*, 12(3):231–255, 2002.

[Par03] L. Parker. The effect of heterogeneity in teams of 100+ mobile robots. In A. Schultz and L. Parker, editors, *Proc. of NRL Workshop on Multi-Robot Systems*, volume 2, pages 205–215, Washington, DC, USA, 2003.

[PM00] P. Pirjanian and M. Matarić. Multiple objective vs. fuzzy behavior coordination. In D. Drainkov and A. Saffiotti, editors, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, pages 235–253. Studies on Fuzziness and Soft Computing, Springer-Verlag, 2000.

[PNDW98] D. Pagac, E. Nebot, and H. Durrant-Whyte. An evidential approach to map-building for autonomous vehicles. *IEEE Trans. on Robotics and Automation*, 14(4):623–629, 1998.

[PPCC02] G. Pereira, B. Pimentel, L. Chaimowicz, and M. Campos. Coordination of multiple mobile robots in an object carrying task using implicit communication. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2002)*, pages 281–286, 2002.

[Pre05] Army Technology. Predator unmanned aerial vehicle, Jul. 2005. [Online]. Available: http://www.army-technology.com/projects/predator.

[RB02] S. Roumeliotis and G. Bekey. Distributed multirobot localization. *IEEE Trans. on Robotics and Automation*, 18(5):781–795, 2002.

[RBL$^+$05] J. Rodrigues, S. Brandão, J. Lobo, R. Rocha, and J. Dias. RAC robotic soccer small-size team: Omnidirectional drive modelling and robot construction. In *Proc. of Scientific Meeting of Robótica'2005 National Robotics Festival*, pages 130–135, Coimbra, Portugal, Apr. 29, 2005.

[RDC03] R. Rocha, J. Dias, and A. Carvalho. Assessing information utility in cooperation-based robotic systems. In *Proc. of 11th Int. Conf. on Advanced Robotics (ICAR'2003)*, pages 840–845, Coimbra, Portugal, Jul. 2003.

[RDC05a] R. Rocha, J. Dias, and A. Carvalho. Cooperative multi-robot systems: a study of vision-based 3-D mapping using information theory. To appear (accepted) in the Elsevier Science's Robotics and Autonomous Systems journal, Sep. 2005.

[RDC05b] R. Rocha, J. Dias, and A. Carvalho. Cooperative multi-robot systems for vision-based 3-D mapping using information theory. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2005)*, pages 386–391, Barcelona, Spain, Apr. 2005.

[RDC05c] R. Rocha, J. Dias, and A. Carvalho. Entropy gradient-based exploration with cooperative robots in 3-D mapping missions. In *Proc. of ICRA'2005 Workshop on Cooperative Robotics, IEEE Int. Conf. on Robotics and Automation*, Barcelona, Spain, Apr. 2005.

[RDC05d] R. Rocha, J. Dias, and A. Carvalho. Exploring information theory for vision-based volumetric mapping. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2005)*, pages 2409–2414, Edmonton, Canada, Aug. 2005.

[RDJ95] D. Rus, B. Donald, and J. Jennings. Moving furniture with teams of autonomous robots. In *Proc. of IEEE/RSJ International Workshop on Intelligent Robots and Systems (IROS'95)*, volume 1, pages 235–242, 1995.

[RM02] M. Riedmiller and A. Merke. Using machine learning techniques in complex multi-agent domains. In M. Richter I. Stamatescu, W. Menzel and U. Ratsch, editors, *Perspectives on Adaptivity and Learning*. Springer-Verlag, 2002.

[Roc01] R. Rocha. State of the art of mobile robotics in Portugal. *Robótica, Revista Técnico-Científica, Special issue on Mobile Robotics and Applications*, 43, Apr. 2001. In Portuguese.

[RR04] S. Roumeliotis and I. Rekleitis. Propagation of uncertainty in cooperative multirobot localization: analysis and experimental results. *Autonomous Robots*, 17(1):41–54, 2004.

[Saf97] A. Saffiotti. Fuzzy logic in autonomous robotics: Behavior coordination. In *Proc. of 6th IEEE International Conference on Fuzzy Systems*, pages 573–578, Barcelona, Spain, 1997.

[SB03] C. Stachniss and W. Burgard. Mapping and exploration with mobile robots using coverage maps. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2003)*, pages 467–472, 2003.

[SBV04] J. Searock, B. Browning, and M. Veloso. Segway CM-RMP robot soccer player. In *Proc. of RoboCup Int. Symp.*, Lisbon, Portugal, Jul. 2004.

[SDH+04] V. Sujan, S. Dubowsky, T. Huntsberger, H Aghazarian, Y. Cheng, and P. Schenker. An architecture for distributed environment sensing with application to robotic cliff exploration. *Autonomous Robots*, 16:287–311, 2004.

[SE05] J. Sáez and F. Escolano. Entropy minimization SLAM using stereo vision. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'05)*, pages 37–43, Barcelona, Spain, 2005.

[Seq99] João Sequeira. *Cooperation Among Robots: a Behavioural Approach Supported On Group Theory*. PhD thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa, Portugal, 1999.

[Ser05] Statistical Consulting Services. Statistics tutorial, May 2005. [Online]. Available: http://www.stats-consult.com/tutorials.html.

[SF93] K. Singh and K. Fujimura. Map making by cooperating mobile robots. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'93)*, pages 254–259, 1993.

[SGB05] C. Stachniss, G. Grisetti, and W. Burgard. Recovering particle diversity in a Rao-Blackwellized particle filter for SLAM after actively closing loops. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2005)*, pages 667–672, 2005.

[Sha49] C.E. Shannon. *The Mathematical Theory of Communication*. University of Illinois Press, 1949.

[SHB+02] T. Schmitt, R. Hanek, M. Beetz, S. Buck, and B. Radig. Cooperative probabilistic state estimation for vision-based autonomous mobile robots. *IEEE Trans. on Robotics and Automation*, 18(5):670–684, Oct. 2002.

[SHB04] C. Stachniss, D. Hahnel, and W. Burgard. Exploration with active loop-closing for FastSLAM. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2004)*, 2004.

[SHP$^+$01] P. Schenker, T. Huntsberger, P. Pirjanian, E. Baumgartner, H. Aghazarian, S. Dubowsky, and K. Iagnemma. Robotic automation for space: planetary surface exploration, terrain-adaptive mobility, and multi-robot cooperative tasks. In *Proc. of the SPIE Symp. on Intelligent Robots and Computer Vision XX*, volume 4572, Oct. 2001.

[Sic05] Sick AG. Laser measurement systems, Jul. 2005. [Online]. Available: http://www.sick.de.

[Smi80] R. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, c-29(12):1104–1113, 1980.

[SOD$^+$02] M. Saptharishi, C. Oliver, C. Diehl, K. Bhat, J. Dolan, A. Trebi-Ollenu, and P. Khosla. Distributed surveillance and reconnaissance using multiple autonomous ATVs: CyberScout. *IEEE Trans. on Robotics and Automation*, 18(5):826–836, Oct. 2002.

[Son05] Sony. QRIO humanoid robot, Jul. 2005. [Online]. Available: http://www.sony.net/SonyInfo/QRIO/top_nf.html.

[SR01] A. Saffiotti and E. Ruspini. Global team coordination by local computation. In *Proc. of European Control Conference (ECC'2001)*, Porto, Portugal, 2001.

[SRV00] P. Stone, P. Riley, and M. Veloso. The CMUnited-99 champion simulator team. In E. Pagello M. Veloso and H. Kitano, editors, *RoboCup99: RobotSoccer World Cup III*, pages 25–44. Springer-Verlag, Berlin, 2000.

[SSDNB95] J. Stankovic, M. Spuri, M. Di Natale, and G. Buttazzo. Implications of classical scheduling results for real time systems. *IEEE Computer*, 28(6):16–25, Jun. 1995.

[SSH$^+$02] R. Simmons, S. Singh, D. Hershberger, J. Ramos, and T. Smith. First results in the coordination of heterogeneous robots for large-scale assembly. In T. Balch and L. Parker, editors, *Robot Teams: From Diversity to Polymorphism*. A. K. Peters Ltd., 2002.

[SSHH04] W. Spears, D. Spears, J. Hamann, and R. Heil. Distributed, physics-based control of swarms of vehicles. *Autonomous Robots*, 17(2-3):137–162, 2004.

[SV99] P. Stone and M. Veloso. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. *Artificial Intelligence*, 110(2):241–273, 1999.

[SV00] P. Stone and M. Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots, Special Issue on Heterogeneous Multi-Robot Systems*, 8(3):345–383, Jun. 2000.

[SWG04] W.-M. Shen, P. Will, and A. Galstyan. Hormone-inspired self-organization and distributed control of robotic swarms. *Autonomous Robots*, 17(1):93–105, 2004.

[Tam97] M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.

[TFB98] S. Thrun, D. Fox, and W. Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31:29–53, 1998.

[TFBD01] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo localization for mobile robots. *Artificial Intelligence Journal*, 128(1-2):99–141, 2001.

[THF$^+$03] S. Thrun, D. Hahnel, D. Ferguson, M. Montermelo, R. Riebwel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker. A system for volumetric robotic mapping of underground mines. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'2003)*, 2003.

[Thr01] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *Int. Journal of Robotics Research*, 20(5):335–363, 2001.

[Thr02] S. Thrun. Robotic mapping: a survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artif. Intell. in New Millenium.* M. Kaufmann, 2002.

[TK04] S. Thomson and S. Kagami. Incorporating stereo vision and sonar data into 2.5 dimensional maps. In *Proc of Intelligent and Autonomous Systems*, Amsterdam, Netherlands, 2004.

[TMG04] D. Tarapore, Lungarella. M., and G. Gómez. Fingerprinting agent-environment interaction via information theory. In *Proc. of Intelligent and Autonomous Systems*, Amsterdam, Netherlands, 2004.

[Tou00] C. Touzet. Robot awareness in cooperative mobile robot learning. *Autonomous Robots*, 2:1–13, 2000.

[TPK04] H. Tanner, G. Pappas, and V. Kumar. Leader-to-formation stability. *IEEE Trans. on Robotics and Automation*, 20(3):443–455, Jun. 2004.

[UA04] P. Ulam and R. Arkin. When good comms go bad: communications recovery for multi-robot teams. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 3727–3734, 2004.

[UF93a] T. Ueyama and T. Fukuda. Knowledge acquisition and distributed decision making. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'93)*, volume 3, pages 167–172, 1993.

[UF93b] T. Ueyama and T. Fukuda. Self-organization of cellular robots using random walk with simple rules. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'93)*, volume 3, pages 595–600, 1993.

[vdL97] Jan van der Lubbe. *Information Theory*. Cambridge University Press, 1997. Translated to English by H. Hoeve and S. Gee from a Dutch version published in 1988.

[Vid05] Videre Design. 865 College Avenue, Menlo Park, CA 94025, U.S.A., Jul. 2005. [Online]. Available: http://www.videredesign.com.

[Vio95] Paul Viola. *Alignment by Maximization of Mutual Information*. PhD thesis, Massachusetts Institute of Technology, U.S.A., 1995.

[Vol99] R. Volpe. Navigation results from desert field tests of the Rocky 7 Mars Rover prototype. *Int. Journal of Robotics Research, special issue on Field and Service Robots*, 18(7), Jul. 1999.

[VSHA98] M. Veloso, P. Stone, K. Han, and S. Achim. CMUnited: a team of robotic soccer agents collaborating in an adversarial environment. *Crossroads*, 4(3), Feb. 1998.

[VSK$^+$02] R. Vidal, O. Shakernia, H. Kim, D. Shim, and S. Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE Trans. on Robotics and Automation*, 18(5):662–669, 2002.

[VUF+98] M. Veloso, W. Uther, M. Fujita, M. Asada, and H. Kitano. Playing soccer with legged robots. In *Proc. of IEEE/RSJ Int. Workshop on Intelligent Robots and Systems (IROS'98)*, Victoria, Canada, Oct. 1998.

[WCfMJT05] TWI World Centre for Materials Joining Technology. Automotive industry – photos and illustrations, Jul. 2005. [Online]. Available: http://www.twi.co.uk/j32k/unprotected/band_1/imglb015.html.

[WGD+02] T. Weigel, J.-S. Gutmann, M. Dietl, A. Kleiner, and B. Nebel. CS Freiburg: coordinating robots for successful soccer playing. *IEEE Trans. on Robotics and Automation*, 18(5):685–699, Oct. 2002.

[WM00] B. Werger and M. Matarić. Broadcast of local eligibility for multi-target observation. In G. Bekey L. Parker and J. Barhen, editors, *Distributed Autonomous Robotics Systems 4*, pages 347–356. Springer-Verlag, 2000.

[WMSFS04] M. Wilson, C. Melhuish, A. Sendova-Franks, and S. Scholes. Algorithms for building annular structures with minimalist robots inspired by brood sorting in ant colonies. *Autonomous Robots*, 17(2-3):115–136, 2004.

[Yam98] B. Yamauchi. Frontier-based exploration using multiple robots. In *Proc. of 2^{nd} Int. Conf. on Autonomous Agents*, pages 47–53, 1998.

[YAOA03] A. Yamashita, T. Arai, J. Ota, and H. Asama. Motion planning of multiple mobile robots for cooperative manipulation and transportation. *IEEE Trans. on Robotics and Automation*, 19(2):223–237, Apr. 2003.

[YP92] S. Yuta and S. Premvuti. Coordinating autonomous and centralized decision making to achieve cooperative behaviors between multiple mobile robots. In *Proc. of IEEE/RSJ Int. Workshop on Intelligent Robots and Systems (IROS'92)*, pages 1566–1574, 1992.

[YZD02] M. Yim, Y. Zhang, and D. Duff. Modular robots. *IEEE Spectrum*, pages 30–34, Feb. 2002.

[ZSAC01] Y. Zhang, M. Schervish, E. Acar, and H. Choset. Probabilistic methods for robotic landmine search. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2001)*, pages 1525–1532, Oct. 2001.

[ZSDT02] R. Zlot, A. Slentz, M. Dias, and S. Thayer. Multi-robot exploration controlled by a market economy. In *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'02)*, pages 3016–3023, 2002.

Appendix A

Mathematical proofs

A.1 Product of two Gaussian distributions is still a Gaussian

Consider a continuous random variable X and two Gaussian beliefs for this variable:

$$p_1(x) = N(\mu_1, \sigma_1) = \frac{1}{\sigma_1\sqrt{2\pi}} \exp\left[-\left(\frac{x-\mu_1}{\sqrt{2}\sigma_1}\right)^2\right],$$
$$p_2(x) = N(\mu_2, \sigma_2) = \frac{1}{\sigma_2\sqrt{2\pi}} \exp\left[-\left(\frac{x-\mu_2}{\sqrt{2}\sigma_2}\right)^2\right].$$

The product of these two probability density functions is given by

$$\begin{aligned} p(x) = p_1(x)p_2(x) &= \frac{1}{\sigma_1\sigma_2 2\pi} \exp\left[-\frac{(x-\mu_1)^2}{2\sigma_1^2} - \frac{(x-\mu_2)^2}{2\sigma_2^2}\right] \\ &= \frac{1}{\sigma_1\sigma_2 2\pi} \exp\left[-\frac{1}{2\sigma_1^2\sigma_2^2}\left(\sigma_2^2\left(x^2 - 2\mu_1 x + \mu_1^2\right) + \sigma_1^2\left(x^2 - 2\mu_2 x + \mu_2^2\right)\right)\right] \\ &= \frac{1}{\sigma_1\sigma_2 2\pi} \exp\left[-\frac{\sigma_1^2+\sigma_2^2}{2\sigma_1^2\sigma_2^2}\left(x^2 - 2\left(\frac{\mu_1\sigma_2^2+\mu_2\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right)x + \frac{\mu_1^2\sigma_2^2+\mu_2^2\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right)\right]. \end{aligned} \tag{A.1}$$

Let define the constants

$$\mu = \frac{\mu_1\sigma_2^2 + \mu_2\sigma_1^2}{\sigma_1^2+\sigma_2^2}, \tag{A.2}$$

$$\sigma = \frac{\sigma_1\sigma_2}{\sqrt{(\sigma_1^2+\sigma_2^2)}}. \tag{A.3}$$

As we shall see, these two constants are, respectively, the mean and the standard deviation of the probability density function associated with the product $p_1(x)p_2(x)$. Substituting equations (A.2) and (A.3) in (A.1), we have

$$\begin{aligned}
p(x) = p_1(x)p_2(x) &= \frac{1}{\sigma_1\sigma_2 2\pi}\exp\left[-\frac{1}{2\sigma^2}\left(x^2 - 2\mu x + \frac{\mu_1^2\sigma_2^2+\mu_2^2\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right)\right] \\
&= \frac{1}{\sigma_1\sigma_2 2\pi}\exp\left[-\frac{1}{2\sigma^2}\left((x-\mu)^2 - \mu^2 + \frac{\mu_1^2\sigma_2^2+\mu_2^2\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right)\right] \\
&= \frac{1}{\sigma_1\sigma_2 2\pi}\exp\left[\frac{\sigma_1^2+\sigma_2^2}{2\sigma_1^2\sigma_2^2}\left(\mu^2 - \frac{\mu_1^2\sigma_2^2+\mu_2^2\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right)\right]\exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right] \\
&= \frac{1}{\sigma_1\sigma_2 2\pi}\exp\left[\frac{1}{2\sigma_1^2\sigma_2^2}\left(\frac{(\mu_1\sigma_2^2+\mu_2\sigma_1^2)^2}{\sigma_1^2+\sigma_2^2} - \mu_1^2\sigma_2^2 - \mu_2^2\sigma_1^2\right)\right]\exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right] \\
&= \frac{1}{\sigma_1\sigma_2 2\pi}\exp\left(\frac{\mu_1^2\sigma_2^4 + 2\mu_1\mu_2\sigma_1^2\sigma_2^2 + \mu_2^2\sigma_1^4 - \mu_1^2\sigma_1^2\sigma_2^2 - \mu_1^2\sigma_2^4 - \mu_2^2\sigma_1^4 - \mu_2^2\sigma_1^2\sigma_2^2}{2\sigma_1^2\sigma_2^2\left(\sigma_1^2+\sigma_2^2\right)}\right)\cdot \\
&\qquad \exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right] \\
&= \frac{1}{\sigma_1\sigma_2 2\pi}\exp\left(\frac{2\mu_1\mu_2 - \mu_1^2 - \mu_2^2}{2\left(\sigma_1^2+\sigma_2^2\right)}\right)\exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right] \\
&= \frac{1}{\sigma_1\sigma_2 2\pi}\exp\left[-\frac{(\mu_1-\mu_2)^2}{2\left(\sigma_1^2+\sigma_2^2\right)}\right]\exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right]. \qquad \text{(A.4)}
\end{aligned}$$

Let define the constant

$$\beta = \sqrt{2\pi\left(\sigma_1^2+\sigma_2^2\right)}\exp\left[\frac{(\mu_1-\mu_2)^2}{2\left(\sigma_1^2+\sigma_2^2\right)}\right]. \qquad \text{(A.5)}$$

Substituting equation (A.5) in equation (A.4), we have finally

$$\begin{aligned}
p(x) = p_1(x)p_2(x) &= \frac{1}{\beta\sigma\sqrt{2\pi}}\exp\left[-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right] \qquad \text{(A.6)} \\
&= \frac{1}{\beta}.N(\mu,\sigma). \qquad \text{(A.7)}
\end{aligned}$$

This equation shows that product $p_1(x)p_2(x)$ is still a Gaussian distribution $\beta p(x)$ with parameters μ and σ, given by equations (A.2) and (A.3), respectively. The function $p(x)$ must be multiplied by the constant β, given by equation (A.5), in order to obtain a valid probability density function, *i.e.* to ensure that it sums up to one over all x.

A.2 Application examples of the mutual information definition for sets of discrete random variables

This section presents thoroughly some application examples of the definitions derived in section 6.3, page 187, concerning the computation of mutual information with sets of discrete random variables.

A.2.1 Mutual information between a set and one of its random variables

Consider a set of n discrete random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ and X_1 as one of the variables contained on it ([1]). The mutual information between $\mathcal{X}$ and X_1 is given by

$$\begin{aligned} I(\mathcal{X}; X_1) &= \sum_{i=1}^{n} I(X_i; X_1 \mid X_1, \ldots X_{i-1}) && \text{(A.8)} \\ &= I(X_1; X_1) + \sum_{i=2}^{n} I(X_i; X_1 \mid X_1, \ldots X_{i-1}) && \text{(A.9)} \\ &= H(X_1). && \text{(A.10)} \end{aligned}$$

Equation (A.8) is obtained by direct application of equation (6.5), page 188. Equation (A.9) is obtained by partitioning the sum in equation (A.8) into two terms. Then, it can be easily proven that all terms in the sum of the second term of equation (A.9) are equal to zero, which yields equation (A.10), because, accordingly with equation (3.27), page 91, entropy is self-information.

Note, for example, that, acoordingly with the definition of conditional mutual information given by equation (3.28), page 91, the first term in the sum of the second term of equation (A.9) is given by

$$\begin{aligned} I(X_2; X_1 \mid X_1) &= H(X_2 \mid X_1) - H(X_2 \mid X_1, X_1) \\ &= H(X_2 \mid X_1) - H(X_2 \mid X_1) = 0, \end{aligned}$$

and the second term of that sum is given by

$$\begin{aligned} I(X_3; X_1 \mid X_1, X_2) &= H(X_3 \mid X_1, X_2) - H(X_3 \mid X_1, X_2, X_1) \\ &= H(X_3 \mid X_1, X_2) - H(X_3 \mid X_1, X_2) = 0. \end{aligned}$$

It can be proven by induction that all those terms are null indeed.

[1] The example would be very similar if another variable $X_i \in \mathcal{X}$ different from X_1 would have been chosen.

A.2.2 Mutual information between two small sets of random variables

Consider the sets of discrete random variables $\{X_1, X_2, X_3\}$ and $\{Y_1, Y_2\}$. Using equation (6.11), page 189, the definition of conditional mutual information given by equation (3.28), page 91, and the definition of joint entropy given by equation (3.18), page 89, their mutual information can be computed as

$$\begin{aligned}
I(X_1, X_2, X_3; Y_1, Y_2) &= I(X_1; Y_1) + I(X_1; Y_2 \mid Y_1) \\
&\quad + I(X_2; Y_1 \mid X_1) + I(X_2; Y_2 \mid X_1, Y_1) + I(X_3; Y_1 \mid X_1, X_2) + I(X_3; Y_2 \mid X_1, X_2, Y_1) \\
&= H(X_1) - H(X_1 \mid Y_1) + H(X_1 \mid Y_1) - H(X_1 \mid Y_1, Y_2) + H(X_2 \mid X_1) - H(X_2 \mid X_1, Y_1) \\
&\quad + H(X_2 \mid X_1, Y_1) - H(X_2 \mid X_1, Y_1, Y_2) + H(X_3 \mid X_1, X_2) - H(X_3 \mid X_1, X_2, Y_1) \\
&\quad + H(X_3 \mid X_1, X_2, Y_1) - H(X_3 \mid X_1, X_2, Y_1, Y_2) \\
&= [H(X_1) + H(X_2 \mid X_1) + H(X_3 \mid X_1, X_2)] - [H(X_1 \mid Y_1, Y_2) \\
&\quad + H(X_2 \mid X_1, Y_1, Y_2) + H(X_3 \mid X_1, X_2, Y_1, Y_2)] \\
&= H(X_1, X_2, X_3) - [H(X_1, Y_1, Y_2) - H(Y_1, Y_2) + H(X_1, X_2, Y_1, Y_2) - H(X_1, Y_1, Y_2) \\
&\quad + H(X_1, X_2, X_3, Y_1, Y_2) - H(X_1, X_2, Y_1, Y_2)] \\
&= H(X_1, X_2, X_3) + H(Y_1, Y_2) - H(X_1, X_2, X_3, Y_1, Y_2).
\end{aligned}$$

Appendix B

Parameters used in volumetric mapping experiments

This appendix presents the values of most of the parameters related with the volumetric mapping experiments, which were carried out with the robots depicted in Fig. 4.10, page 124. Although some of them are intrinsic characteristics of the equipment, they are mostly parameters that are adjusted by the user or obtained through calibration.

B.1 Miscellaneous tables

Table B.1: Parameters of the SVS stereo algorithm.

Parameter	Value
Number of disparities	16
Horopter's offset	20
Search window size	11
Confidence level threshold	22
Left/Right filter	on

Table B.2: Parameters of the sensor model related with the standard deviation for the stereo-vision range sensors depicted in Fig 4.11, page 126. They are used in equations (4.29) and (4.30), page 111.

Parameter		Value	
σ_{min}	Standard deviation at null distance	-0.06	mm
ζ	First derivative with distance	3.75×10^{-3}	
τ	Damping ratio with the distance to the obstacle	2	m

Table B.3: Parameters of the Scout robot from Nomadics Technologies, Inc.

Parameter	Value		
Diameter	385	mm	
Height	350	mm	
Maximum speed	101.6	$cm.s^{-1}$	
Maximum acceleration	203.2	$cm.s^{-2}$	
Configured maximum speed	50.8	$cm.s^{-1}$	(*)
Configured maximum acceleration	76.2	$cm.s^{-2}$	(*)
Safety speed / speed before stopping	5.08	$cm.s^{-1}$	(*)
Stopping distance	20	cm	(*)
Maximum angular speed	302.4	$deg.s^{-1}$	(*)
Angular speed at the end of rotation	30.2	$deg.s^{-1}$	(*)
Angular stopping distance	35	$deg.$	(*)
Collision avoidance critical distance	200	mm	(*)
Collision avoidance safety distance	500	mm	(*)
Sonars' range	$15 \ldots 107$	cm	
Time between two sonars are fired	0.1	s	(*)

(*) Value configured in the volumetric mapping software.

Table B.4: Main characteristics of the stereo vision sensors depicted in Fig 4.11, page 126. All the values but the resolution and the baseline were obtained through calibration with the SVS software.

Parameter	**Value**	
Resolution, image pair	160 x 120 pixels	
Resolution, individual camera	320 x 240 pixels	
	STH-V2	*STH-V3*
Baseline $[mm]$	70	85
Focal length $[mm]$	7.542	4.380
Aspect ratio	0.984	0.916
Radial distortion, 2nd order	6.225×10^{-3}	25.847×10^{-3}
Camera center [pixels]	$\begin{bmatrix}151.404 & 124.933\end{bmatrix}^T$	$\begin{bmatrix}146.992 & 58.821\end{bmatrix}^T$

Table B.5: Parameters used in the experiments carried out in April 2005. The table only presents those parameters that are not specific to the robots or to their stereo-vision sensors, which are presented in previous tables of this appendix.

Parameter	**Value**
Voxel's edge ϵ $[m]$	0.1
Volumetric grid size (x,y,z) [number of voxels]	(43,53,7)
Discrete entropy, number of histogram bins b	128
Stopping criteria H_{th} (entropy threshold) [bits]	3×10^4
Gaussian's initial standard deviation for a voxel	10.0
Initial map's entropy $H(\mathcal{C} \mid \mathcal{M}_0)$ [bits]	11.167×10^4
Stereo-vision sensor's minimum distance to obstacles $[m]$	1
Team size n	$\{1 \ldots 10\}$
Min. inform. utility of a comm. measurement I_{min} $(n > 1)$	0.0145
Max. number of comm. measurements $\max(s_k)$ $(n > 1)$	2500
Exploration, jump distance with null gradient $[m]$	1.5
Uncoordinated exploration strategy	
Surveying fixed neighborhood radius ε $[m]$	1
Coordinated exploration strategy	
Surveying mutual information scale factor ξ [bits]	5.0
Cost sensitivity coefficient κ	$\left\{0, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\right\}$

Table B.6: Parameters used in the experiments carried out in July 2004. The table only presents those parameters that are not specific to the robots or to their stereo-vision sensors, which are presented in previous tables of this appendix.

Parameter	**Value**
Voxel's edge ϵ $[m]$	0.1
Volumetric grid size (x,y,z) [number of voxels]	(42,44,10)
Discrete entropy, number of histogram bins b	128
Stopping criteria H_{th} (entropy threshold) [bits]	2.65×10^4
Gaussian's initial standard deviation for a voxel	10.0
Initial map's entropy $H(\mathcal{C} \mid \mathcal{M}_0)$ [bits]	12.936×10^4
Stereo-vision sensor's minimum distance to obstacles $[m]$	1
Team size n	$\{1, 2\}$
Min. inform. utility of a comm. measurement I_{min} $(n > 1)$	$\{0, 0.00723, 0.0145, 0.074, 0.152\}$
Max. number of comm. measurements $\max(s_k)$ $(n > 1)$	$\{500, 1000, 1750, 2500, 5000, \infty\}$
Exploration, jump distance with null gradient $[m]$	1.5
Exploration, fixed neighborhood radius ε $[m]$	0.3

B.2 Spatial transformation from the robot's sensor to the robot's platform

The transformation ${}^{R}\mathbf{T}_{C} = \left[{}^{R}\mathbf{R}_{C} \middle| {}^{R}\mathbf{t}_{C}\right]_{3\times4}$, which is used in equation (4.56), page 131, to transform coordinates expressed on the right camera's reference frame $\{C\}$ to the robot's reference frame $\{R\}$ (see Fig. 4.14), was obtained through the calibration procedure presented in section 4.7.4, page 129. Observe in Fig. 4.10, page 124, the placement of the stereo vision sensor on each robot. The left robot is equipped with the stereo rig STH-V2 and the right robot is equipped with the stereo rig STH-V3.

For the stereo rig STH-V2, the translation vector is

$$ {}^{R}\mathbf{t}_{C} = \begin{bmatrix} 100 & -35 & 380 \end{bmatrix}^{T}, \tag{B.1}$$

the rotation matrix is

$$ {}^{R}\mathbf{R}_{C} = \begin{bmatrix} -0.058891 & -0.217779 & 0.974220 \\ -0.998177 & -0.000083 & -0.060358 \\ 0.013226 & -0.975998 & -0.217377 \end{bmatrix} \tag{B.2}$$

and the transformation matrix is

$$ {}^{R}\mathbf{T}_{C} = \begin{bmatrix} -0.058891 & -0.217779 & 0.974220 & 100 \\ -0.998177 & -0.000083 & -0.060358 & -35 \\ 0.013226 & -0.975998 & -0.217377 & 380 \end{bmatrix}. \tag{B.3}$$

Given equation (B.2), and accordingly with equation (4.60), page 130, the stereo rig's tilt angle towards the floor is roughly $\alpha = 13\ deg$.

For the stereo rig STH-V3, the translation vector is

$$ {}^{R}\mathbf{t}_{C} = \begin{bmatrix} 120 & -42.5 & 435 \end{bmatrix}^{T} \tag{B.4}$$

the rotation matrix is

$$ {}^{R}\mathbf{R}_{C} = \begin{bmatrix} -0.037071 & -0.113945 & 0.992795 \\ -0.999297 & -0.001296 & -0.037463 \\ 0.005555 & -0.993486 & -0.113816 \end{bmatrix} \tag{B.5}$$

and the transformation matrix is

$$ {}^{R}\mathbf{T}_{C} = \begin{bmatrix} -0.037071 & -0.113945 & 0.992795 & 120 \\ -0.999297 & -0.001296 & -0.037463 & -42.5 \\ 0.005555 & -0.993486 & -0.113816 & 435 \end{bmatrix}. \tag{B.6}$$

Given equation (B.5), and accordingly with equation (4.60), page 130, the stereo rig's tilt angle towards the floor is roughly $\alpha = 7 \ deg$.

These transformation matrices assume that coordinates are always expressed in millimeters.

Appendix C

Computing range data with the SVS stereo engine

The SVS — Small-Vision System — is a stereo engine from SRI International for computing range data from stereo images [KB02]. The version 2.3c of the software was used in the experiments reported in chapters 4, 5 and 6 to measure distances through the stereo rigs presented in section 4.7.2, page 125. The SVS software implements an area correlation algorithm for computing range from stereo images. See Fig. C.1 for an example of a depth map yielded by the SVS engine.

Consider a stereo-vision system with two pinhole cameras ([1]). Given the left and right images depicted in Fig. C.1-a,b, the software performs an image warp for rectifying them. This rectification effectively rotates the images about their centers of projection, in order to establish an ideal stereo setup, wherein the two cameras have parallel optical axes and horizontal epipolar lines (see Fig. C.2).

Accordingly with the epipolar geometry, given a stereoscopic vision system comprising n cameras, any point and its associated projections in the image planes of the n cameras belong to a common plane: the epipolar plane. The intersection of this plane with the cameras' image planes are the epipolar lines. Ensuring that in a binocular system (two cameras) the two epipolar lines are horizontal is crucial for the SVS's correlation algorithm, since it looks for matches along horizontal scan lines. This constraint simplifies very much the correspondence problem, *i.e.* associating pixels in both cameras that represent the projection of the same point in space, because it ensures that any point in space is projected in the same row of both cameras.

The stereo pair depicted in Fig. C.1-c,d is obtained after applying the rectification

[1] See section D.1, page 265, for a brief introduction about the pinhole camera model.

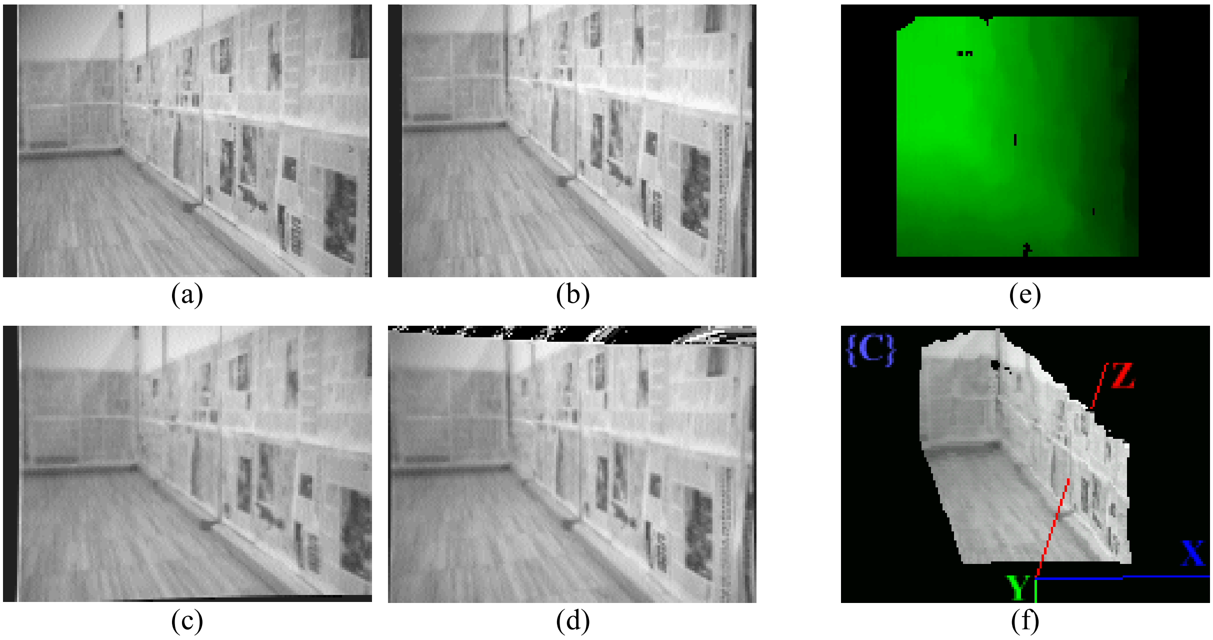


Figure C.1: Example of a depth map provided by stereo-vision: (a) left camera image; (b) right camera image; (c) left camera image after rectification; (d) right camera image after rectification; (e) disparity map; (f) 3-D reconstruction relative to the right camera coordinates reference frame $\{C\}$.

procedure on the stereo pair of Fig. C.1-a,b. Note that the two images are now aligned correctly with respect to each other. Then, the SVS stereo engine is able to compute the disparity map depicted in Fig. C.1-e, which contains depth information: it associates depth with each pixel in an image and a correlated pixel in the other image; lighter regions have higher depth.

If the stereo-vision system is properly calibrated, the SVS stereo engine is able to use the disparity map to compute the coordinates $[x, y, z]^T$ for each pixel of the right image and, thus, to obtain the 3-D reconstruction depicted in Fig. C.1-f. This graph projects each pixel in space given its 3-D coordinates relative to the right camera's coordinates reference frame $\{C\}$.

The SVS stereo engine provides the user with C/C++ libraries containing sets of functions that can be used to: acquire and display video from a binocular stereo-vision system in real time; calibrate the stereo-vision, so as to properly accomplish the aforementioned stereo pair rectification; eliminate low probability stereo matches due to lack of image texture and filter occluded regions, that only appear in one of the images, through a consistency check; compute disparity maps; and compute range data from a stereo pair and obtain a 3-D reconstruction. Most of these functionalities are exercised in a stand-

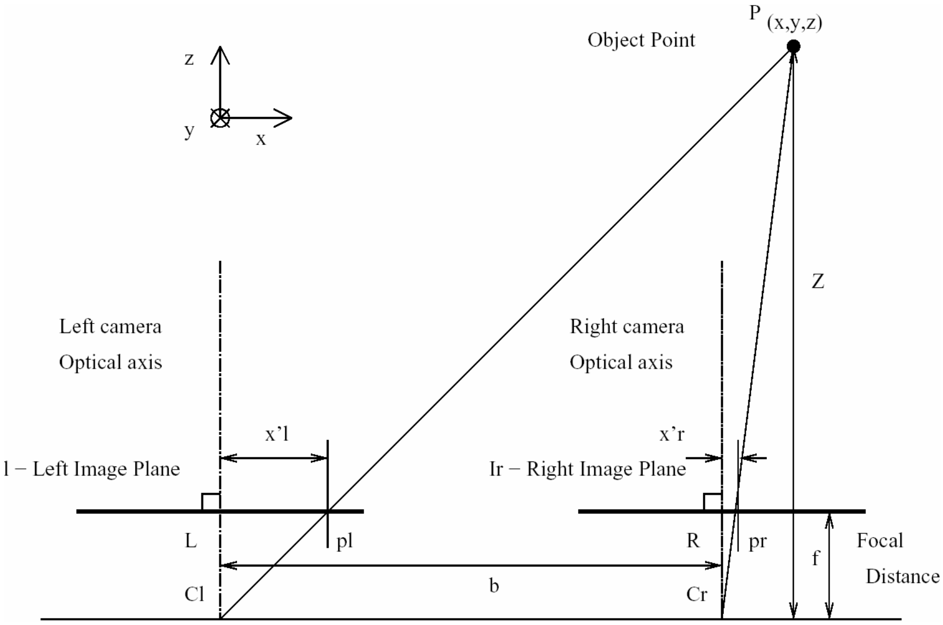


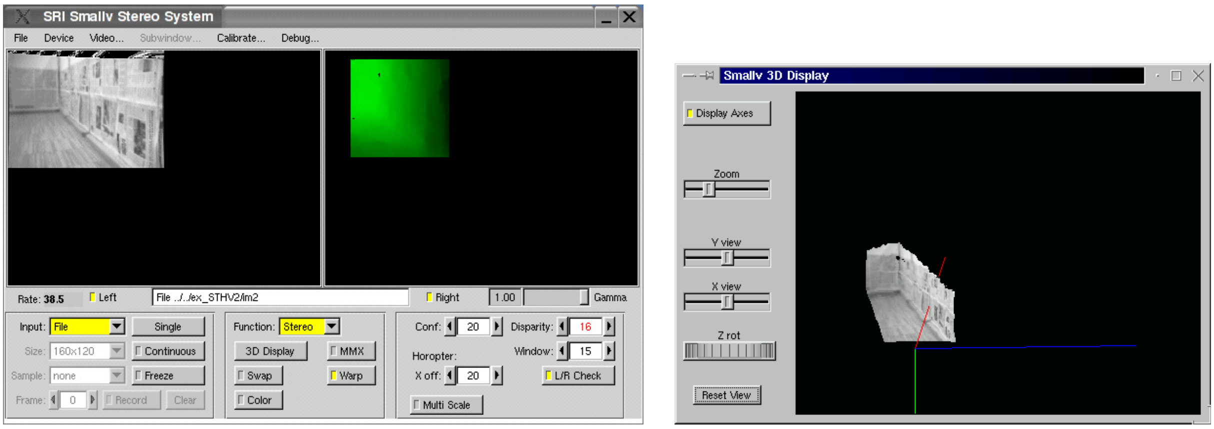
Figure C.2: Binocular stereo-vision system with front-parallel geometry (figure reproduced from [LAD02]). Given the system's geometry, the cameras' focal length f and the baseline b, the coordinates $[x, y, z]^T$ of any point P in space are computed through *triangulation* upon its projection in both cameras (x'_l and x'_r are the projections' coordinates along the xx axis).

alone graphical application, which can be used by the user to get insight about using the provided libraries. The main windows of the application are depicted in Fig. C.3.

Before using a given stereo-vision system to compute range data through the SVS stereo engine, it must be properly calibrated. This calibration comprises *intrinsic calibration* and *extrinsic calibration*. While the former one deals with the properties of the individual cameras, such as lens distortion and differing focal lengths for the two cameras, the latter one deals with the spatial relationship of the cameras to each other, including imperfections due to non-parallel optical axes and non-horizontal epipolar lines.

The SVS stereo engine provides the user with an automatic calibration procedure based on a nonlinear optimization algorithm; Fig. C.4-a depicts the SVS calibration window. In order to calibrate the stereo-head, the user has to provide SVS with five image pairs of a planar checkerboard. Fig. C.4-b depicts the set of image pairs that were used to calibrate the stereo-vision sensor STH-V2 shown in Fig. 4.11-a. When choosing these image pairs, the goal is to include views that differ both in translation and rotation, so as to ensure that the calibration procedure converges ([2]).

[2] Finding an image set that ensures the calibration convergence is sometimes a tedious task, especially with low resolution cameras such as those cameras depicted in Fig. 4.11.



SRI Smallv Stereo System

Rate: 38.5

■ Left

File ../../ex_STHV2/im2

■ Right

1.00 Gamma

Input: File

Size: 160x120

Sample: none

Frame: 0

□ Continuous

□ Freeze

□ Record

Function: Stereo

□ MMX

□ Swap

■ Warp

□ Color

Conf: 20

Disparity: 16

Window: 15

Horopter:

X off: 20

■ L/R Check

□ Multi Scale

Smallv 3D Display

■ Display Axes

(a) (b)

Figure C.3: SVS — Small Vision System — stereo-vision engine from SRI International: (a) main window of the software application wherein the user can configure the video acquisition, tune the stereo correlation algorithm's parameters and visualize the stereo image pair or, alternatively, one of the images (left) and the disparity map (right); (b) window showing a 3-D reconstruction of the most recent acquired stereo image pair.

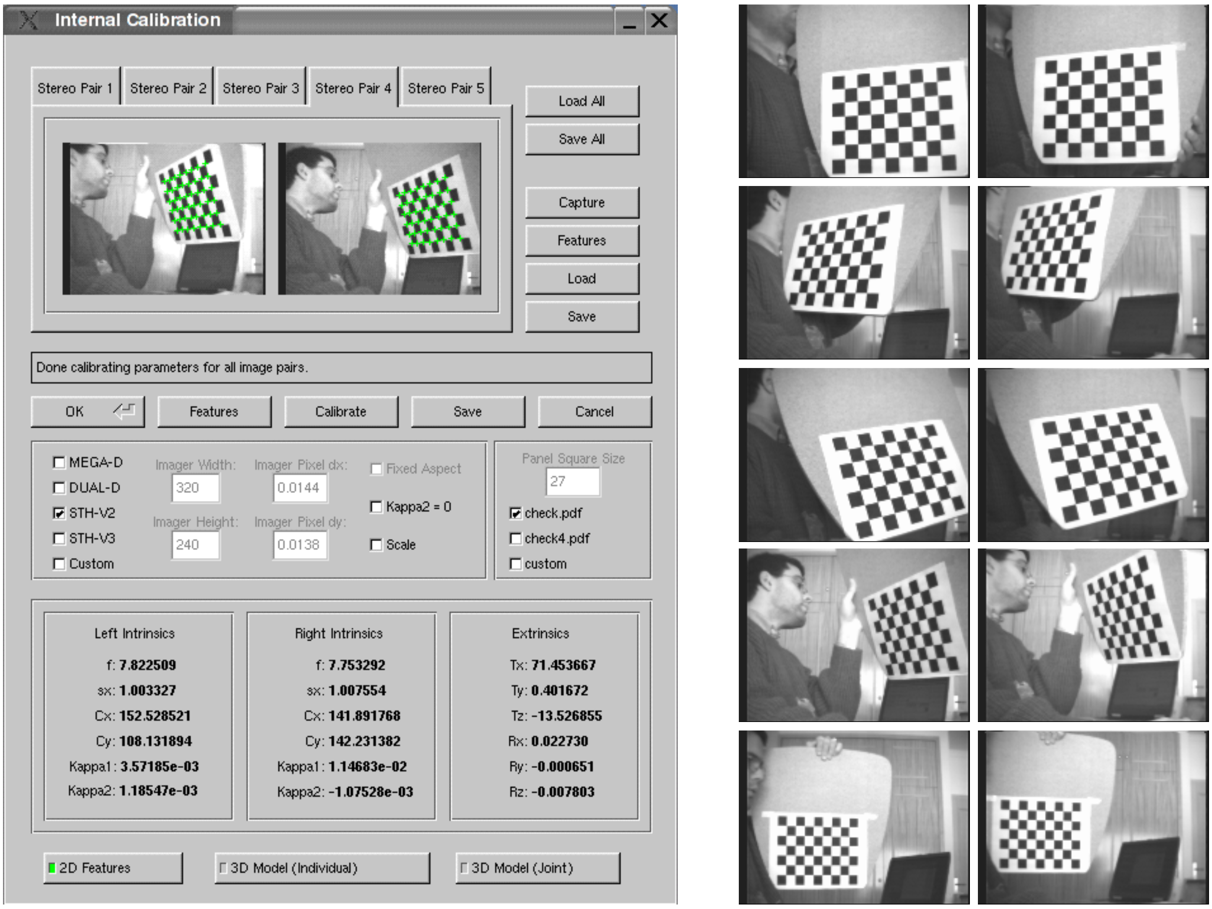
After choosing a set of image pairs, the calibration procedure starts by finding the image features in each of the image pairs. These features are the corners of the checkerboard's squares, which are depicted in Fig. C.4-a with (+) for the fourth stereo pair. Then, the calibration algorithm may be started and, at the end, if it successfully converges, the calibration parameters are written to a text file and are shown in the half-bottom of the calibration window (see Fig. C.4-a). The information stored in this text file must be loaded before performing the aforementioned image warp procedure, in order to properly initialize the stereo algorithm. After the successful calibration, the stereo-vision sensor can be used as a range sensor for getting 3-D range data (see Fig. C.1 for an example).

C.1 Parameters of the stereo algorithm

There are some parameters that influence the behavior of the stereo algorithm of SVS. They are the number of disparities, the horopter's offset, the search window size and the confidence level.

The *disparity* associated with a given 3-D point in the scene is the offset between its projections on the image planes of both cameras of the binocular stereo-vision system, which is directly related to the distance of the point normal to the image planes ([3]). This

[3] This statement assumes that the image planes of both cameras are embedded within the same plane,



Done calibrating parameters for all image pairs.

☐ MEGA-D
☐ DUAL-D
☑ STH-V2
☐ STH-V3
☐ Custom

Imager Width: 320
Imager Height: 240
Imager Pixel dx: 0.0144
Imager Pixel dy: 0.0138

☐ Fixed Aspect
☐ Kappa2 = 0
☐ Scale

Panel Square Size: 27
☑ check.pdf
☐ check4.pdf
☐ custom

Left Intrinsics	Right Intrinsics	Extrinsics
f: **7.822509**	f: **7.753292**	Tx: **71.453667**
sx: **1.003327**	sx: **1.007554**	Ty: **0.401672**
Cx: **152.528521**	Cx: **141.891768**	Tz: **-13.526855**
Cy: **108.131894**	Cy: **142.231382**	Rx: **0.022730**
Kappa1: **3.57185e-03**	Kappa1: **1.14683e-02**	Ry: **-0.000651**
Kappa2: **1.18547e-03**	Kappa2: **-1.07528e-03**	Rz: **-0.007803**

(a)

(b)

Figure C.4: Calibration of a stereo-vision sensor: (a) calibration window of the SVS stereo engine; (b) set of stereo image pairs used to calibrate the stereo-vision sensor STH-V2 shown in Fig. 4.11-a, page 126 (one stereo pair per row).

means that images taken from different viewpoints "see" the object at different positions (pixels) and the offset is called disparity. The disparity is inverse proportional to the distance and is proportional to the cameras' focal length and to the stereo-rig's baseline (see Fig. C.2).

Due to the inverse relationship between disparity and distance, most of the change in disparity takes place in the first several meters. Furthermore, the smallest change in range that is discernable by the stereo geometry given a fixed change in disparity, *i.e.* the *range resolution*, is a function of the range itself: at closer ranges, the resolution is much better than at farther ranges; the range resolution gets worst as the square of the range ([4]). Obviously the images' resolution, *i.e.* the pixel size, has also a strong impact on the range resolution, so that smaller pixel sizes (higher images' resolution) yield better range resolution.

Even rectifying the images, it may not be possible to match every object in the scene. The *number of disparities* is associated with the search range for matching pixels from both images. Changing the disparity search size affects the time it takes to process stereo. For instance, a search space of 32 pixels takes about twice as long as a search space of 16 pixels.

Given a disparity search size, the range of objects that can be successfully measured is restricted to a 3-D volume that is covered by the search range of the stereo algorithm. This 3-D volume is denoted as the *horopter*. Those objects cannot be too close or too farther to the sensor. The smallest distance is associated with the highest disparity and the farthest distance is associated with the lowest disparity. For instance, if the number of disparities is equal to 16, the highest distance yields a disparity equal to 0 and the smallest distance yields a disparity equal to 15.

The *horopter's offset* allows to vary the placement of the horopter, *i.e.* to place it closer or farther to the sensor. This offset may be also useful to compensate the fact of the optical axes of the two cameras not being exactly parallel.

The SVS stereo algorithm is local search method, *i.e.* it is an area correlation algorithm which tries to find corresponding elements between the two images. To do this, it uses correlation to compare small patches, or windows, from both images. The *search window size* specifies the size of each patch being compared and presents an important tradeoff between correlation success and signal-to-noise ratio.

i.e. they are co-planar like in Fig. C.2. In this figure, the disparity along the xx axis is the offset $x'_l - x'_r$.

[4] Note that focal length and baseline, which are intrinsic parameters of the stereo-rig, also have an influence on the resolution. They have an inverse influence on the resolution, so that larger baselines and focal lengths make the range resolution better.

Small windows are more likely to be similar in images taken from different viewpoints, but larger windows increase the signal-to-noise ratio, especially in textureless areas. Moreover, there is also a tradeoff related with disparity image spatial resolution: large windows tend to "smear" foreground objects, *i.e.* the object appears larger in the disparity map than in the original input image. This means that smaller windows have a poorer signal-to-noise ratio but allows to match smaller objects.

Like every vision algorithms, stereo algorithms are error-prone. Errors result from noisy video signals — random errors — and from the difficulty of matching non-textured, or regularly textured, image areas — systematic errors.

SVS implements a texture filter that assigns to each pair of matched patches a *confidence level*, *i.e.* a probability value for the matching validity. Given a *confidence level threshold*, which the user can adjust accordingly with the application requirements, the stereo algorithm reject matchings whose confidence level is less than the threshold.

Besides the texture filter, SVS implements a left/right filter that eliminate errors due to correlation windows covering areas with very different disparities or discontinuities. The filter's name is because it is implemented through consistency tests involving the comparison of regions from the left and right images.

The standalone application of SVS, shown in Fig. C.3-a, contains a set of control buttons to tune all the aforementioned stereo parameters. In the specific SVS software application that was programmed to carry out mapping experiments with the stereo-vision sensors mounted on the robots of Fig. 4.10, page 124, those stereo parameters were empirically set to the values presented in Table B.1, page 249. The example shown in Fig. C.1 uses this set of parameters.

Appendix D

Global localization with a color camera

In equation (4.57), page 129, the transformation given by matrix ${}^{W}\mathbf{T}_{R} = \left[{}^{W}\mathbf{R}_{R} \middle| {}^{W}\mathbf{t}_{R} \right]_{3\times4}$ is not rigid and is given by some localization scheme. Because the robot's motion is restricted to the floor plane, *i.e.* a plane parallel to plane xy, the goal of the localization system is just to determine the robot's absolute position $\mathbf{x} = [x, y, z]^T \in \mathbb{R}^3$ and the yaw angle θ giving the robot's heading, on the global reference frame $\{W\}$ (see Fig. 4.14, page 131). Moreover, the coordinate z is constant and is known *a priori* by the localization system. Given the values of x, y, z and θ, the localization system transforms coordinates from the robot's reference frame $\{R\}$ to the global reference frame $\{W\}$ using the transformation matrix

$$
{}^{W}\mathbf{T}_{R} = \left[{}^{W}\mathbf{R}_{R} \middle| {}^{W}\mathbf{t}_{R} \right]_{3\times4} = \left[\begin{array}{ccc|c} \cos\theta & -\sin\theta & 0 & x \\ \sin\theta & \cos\theta & 0 & y \\ 0 & 0 & 1 & z \end{array} \right], \qquad \text{(D.1)}
$$

wherein the rotation matrix ${}^{W}\mathbf{R}_{R}$ is a pure rotation about zz axis of θ radians.

During the mapping experiments reported in chapters 4, 5 and 6, which were carried out in laboratory, a global localization scheme based on a global camera was used. It uses the RGB analog color camera from Toshiba depicted in the bottom-right of Fig. 4.16-a, page 134, which is denoted hereafter as the global camera.

The global camera's video signal is acquired through a Matrox Meteor frame grabber PCI card, which is installed on a desktop PC running the Microsoft Windows 98 operating system (see the bottom-left of Fig. 4.16-a). In order to localize the robots within the workspace, the graphical localization server software, which is depicted in Fig. 4.16-b, was programmed to detect and track colored markers on the top of the robots depicted in Fig. 4.10, page 124. The colors of the markers were chosen in such a way that they could not be easily encountered in other objects cluttering the environment.

The position of each robot is determined upon a blue rectangular shape located on the back-top of the robot's platform. Additionally, each robot has more two rectangular shapes, near to its center and in the front, respectively, whose color is unique: it is pink for the robot on the left of Fig. 4.10 and green for the robot on the right. The purpose of these two additional markers is twofold: firstly, due to their unique color, they allow to identify the robot; secondly, when combined with the blue marker's position, they allow to compute the robot's orientation (yaw angle θ).

Due to their location, although there are some poses wherein one of the markers (pink or green, depending on the robot) is occluded by the robot's stereo-vision sensor, at least one of the markers always appears in the global camera's image. When both of the markers appear in the image, the marker in the front is chosen to compute the robot's orientation, because the accuracy increases with the distance to the blue marker.

The localization software executes a sequence of operations in real-time, which includes: acquiring a color image; converting it from the RGB — Red-Green-Blue — color space to the HSV — Hue-Saturation-Value — color space; performing a color segmentation algorithm to localize pixels belonging to the colors of interest — blue, pink and green; clustering segmented pixels in regions of interest, *i.e.* the regions representing the colored markers; and, finally, using the centroid of the detected colored regions to compute the robots' localization in the global coordinates frame $\{W\}$.

The computation burden associated with the conversion from RGB to HSV is justified by the fact that the latter color space is more robust against variable illumination conditions. Moreover, by using the HSV color space those colors of interest can be easily defined using just the hue and saturation dimensions and a training procedure; the value dimension, which is related with luminance, is not used.

For each one of the three colors of interest, and using 5 bits to encode each color dimension between 0 and 31, a color of interest is defined by a range of values in the hue and saturation dimensions. This classification, which is presented in detail in Table D.1, page 265, is used during color segmentation to decide whether a given pixel of an acquired image belongs or not to those colors of interest.

Although the frame grabber may provide higher resolutions, the resolution 384 x 288 pixels is used in order to restrict the computation burden and ensure a reasonable frame rate. The PC, which is based on a Pentium-II MMX, 200 MHz, with 256 Mb of RAM, is able to perform the localization with a maximum frame rate equal to 2.5 Hz. Since this

Table D.1: Color classes used in the absolute localization with a color camera, expressed in the HSV — Hue-Saturation-Value — color space.

Color class	**Hue**		**Saturation**	
	min.	*max.*	*min.*	*max.*
Blue	14	19	15	21
Pink	20	28	4	14
Green	5	12	6	22

frame rate is somewhat low, robots use odometry most of the time, using the localization server to reset periodically odometry errors.

The localization server works as a TCP/IP server that can accept localization queries from the robots, which act as TCP/IP clients. Whenever a given robot needs to know its absolute localization, it establishes a connection to the server and asks for its localization; then, the server answers by sending the most recent localization estimate.

D.1 Pinhole model of the camera used to localize the robots

The pinhole model is generally used in computer vision to model the behavior of a camera [Lob02]. It is a simplified model, which limits the incidence of light rays onto a surface, enabling the formation of an inverted image of the world (see Fig. D.1). A small opening (the pinhole) is the center of projection of the image and all rays of light forming the image pass through this point.

The distance from the image plane to the center of projection is denoted as the focal length f. The optical axis is normal to the image plane and contains the center of projection, defining the viewing direction of the camera. The image center is the intersection of the optical axis with the image plane. The image is formed by projecting the 3-D scene onto a surface, thus loosing depth information. This mapping can be modeled by a perspective transformation, whereby each point in space is projected onto the image plane. To avoid image inversion, the image plane is usually considered to be in front of the center of projection, forming a non-inverted image plane (mathematical image plane).

Given the localization in the image of the two colored markers of a given robot, the localization server is able to recover the depth information from the image and determine

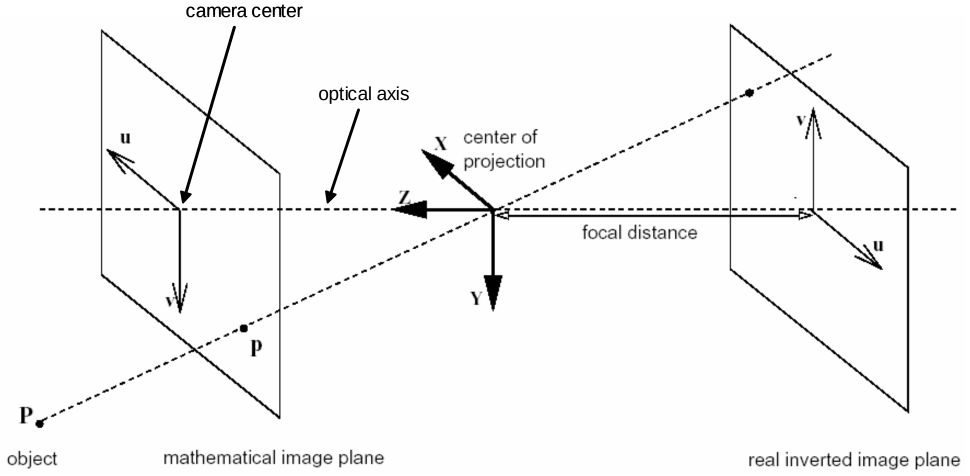


Figure D.1: Pinhole camera model. Figure reproduced from [Lob02].

the robot's pose, because the robot's motion is restricted to a plane parallel to the plane xy, *i.e.* the coordinate z is constant and is known *a priori*. Therefore, it is possible to compute the 3-D coordinates in space of each pixel in the image.

Consider the global camera reference frame $\{G\}$ and the coordinates in space $\mathbf{x}_G = [x_G, y_G, z_G]^T$, expressed on $\{G\}$, of a pixel (u, v) in the image, which is expressed as the vector $\mathbf{x}_P = [u, v, 1]^T$. A simplified pinhole linear model of the camera [Bou05] is given by

$$\mathbf{x}_N = \begin{bmatrix} \frac{x_G}{z_G} \\ \frac{y_G}{z_G} \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \mathbf{x}_P = \mathbf{A}^{-1}\mathbf{x}_P, \tag{D.2}$$

wherein $\mathbf{x}_N$ are the point's normalized coordinates, *i.e.* $\mathbf{x}_G = z_G \cdot \mathbf{x}_N$. The vector $\mathbf{f} = [f_x, f_y]^T$ is the focal length expressed in units of horizontal and vertical pixels ([1]). The vector $\mathbf{c} = [c_x, c_y]^T$ is the camera center (principal point) expressed in pixels.

The pixel's space coordinates expressed on the global reference frame $\{W\}$ is given by

$$\mathbf{x} =^W \mathbf{R}_G \cdot \left(z_G \cdot \mathbf{x}_N +^W \mathbf{t}_G\right) =^W \mathbf{R}_G \cdot \left(z_G \cdot \mathbf{A}^{-1}\mathbf{x}_P +^W \mathbf{t}_G\right), \tag{D.3}$$

wherein ${}^W\mathbf{R}_G$ and ${}^W\mathbf{t}_G$ are, respectively, the 3×3 rotation matrix and the 3-D translation vector from $\{G\}$ to $\{W\}$. In order to use directly equation (D.3) to compute the point's space coordinates $\mathbf{x}$, the value of the coordinate z, which is known *a priori*, can be used

[1] The components f_x and f_y are usually very similar.

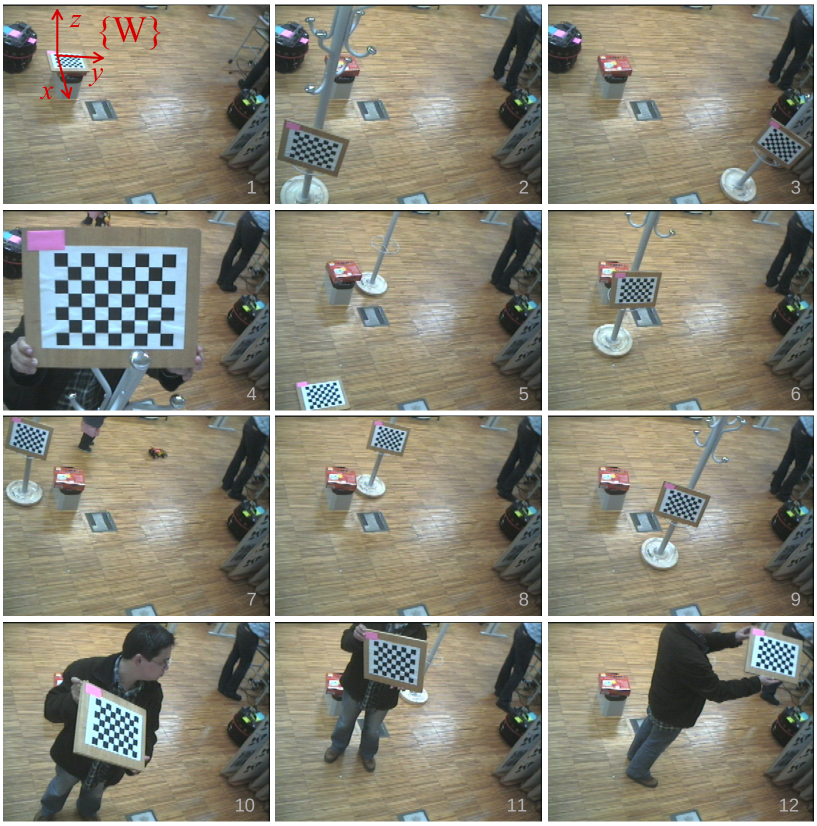


Figure D.2: Image set used in the calibration of the global camera.

to compute z_G as

$$z_G = \frac{z - \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \cdot {}^{W}\mathbf{R}_G \cdot {}^{W}\mathbf{t}_G}{\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \cdot {}^{W}\mathbf{R}_G \cdot \mathbf{x}_N}. \tag{D.4}$$

The camera's model used by the localization application is slightly more complex than the model described by equations (D.2), (D.3) and (D.4), because it also models the camera's non-linear radial and tangential distortions [Bou05]. Nevertheless, these phenomena were not included in the equations above, so as to simplify their understanding.

D.2 Camera calibration

Before using the camera's model to compute pixels' coordinates in space, both intrinsic and extrinsic parameters of the camera must be properly calibrated. While intrinsic parameters — focal length **f**, center coordinates **c** and other distortion coefficients — are related with the intrinsic characteristics of the device, extrinsic parameters — rotation

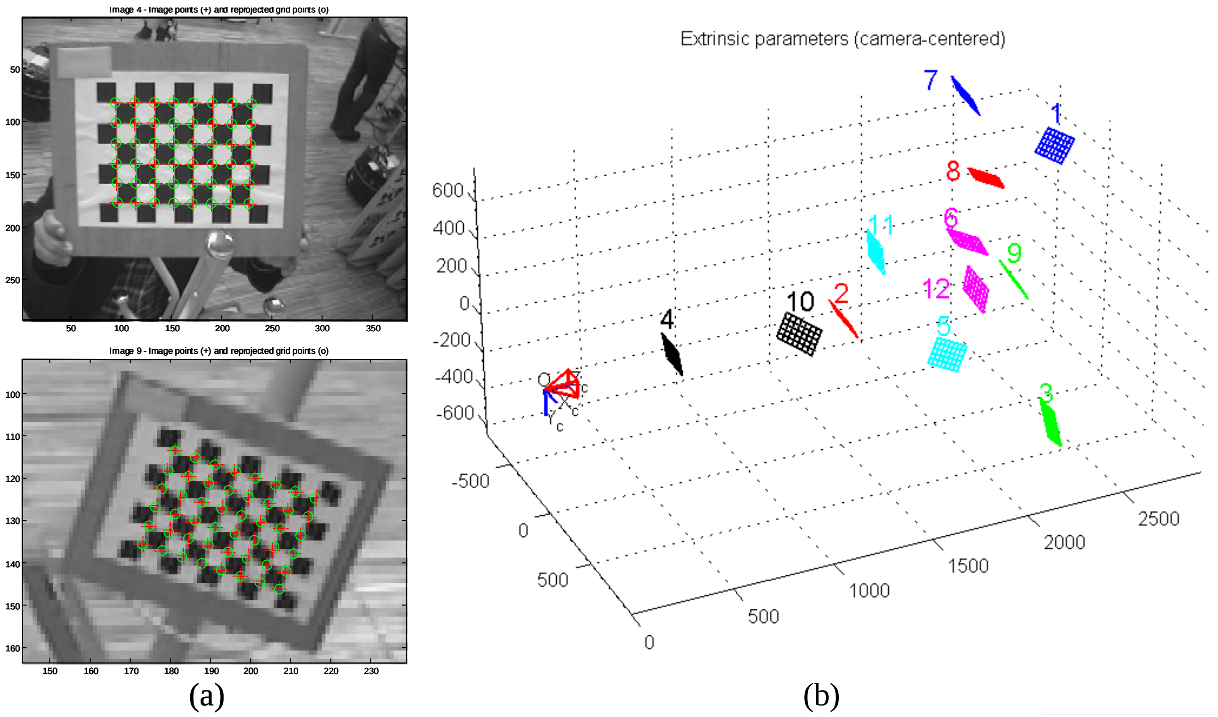


Figure D.3: Calibration of the global camera: (a) two of the images used in the calibration (image 4 and zoom of image 9 from Fig. D.2) showing the points that were processed (+) and their reprojection after calibration (o); (b) 3-D reconstruction of the acquired targets during calibration (checkerboard in different positions) relative to the camera's optical center.

matrix ${}^{W}\mathbf{R}_G$ and translation vector ${}^{W}\mathbf{t}_G$ — establish the spatial transformation between the camera reference frame $\{G\}$ and the global reference frame $\{W\}$.

The camera calibration toolbox for Matlab© [Bou05], which was readily available, was used to calibrate the camera after providing the calibration software with the image set depicted in Fig. D.2, containing different viewpoints of a checkerboard whose characteristics were known *a priori*.

Fig. D.3-a shows two of the images — images 4 and 9 — used in the calibration and the position of the points that were processed, *i.e.* the corners of the checkerboard's squares, and their reprojection after calibration. As it can be observed, the calibration errors are almost imperceptible, even for the image 9 wherein the checkerboard is represented with a much lower resolution. Fig. D.3-b represents a 3-D reconstruction of every checkerboard's positions associated with the images used in the calibration.

The detailed camera's parameters obtained through this calibration procedure are presented in Table D.2. Note that the external coordinates reference frame associated with the image 1 of Fig. D.2 was chosen to be the global reference frame $\{W\}$, *i.e.* its

Table D.2: Main characteristics of the color camera used for absolute localization. The parameters' values were obtained through the camera calibration toolbox for Matlab© [Bou05].

Parameter	**Value**
Intrinsic	
Resolution [pixels]	384 x 288
Focal length $\mathbf{f}$ [pixels]	$\begin{bmatrix} 448.935 & 449.898 \end{bmatrix}^T$
Radial distortion, 2nd order	-0.129308
Radial distortion, 4th order	-0.021539
Tangential distortion parameters	$p_1 = 4.43 \times 10^{-4}$, $p_2 = -9.064 \times 10^{-3}$
Camera center $\mathbf{c}$ [pixels]	$\begin{bmatrix} 162.737 & 147.095 \end{bmatrix}^T$
Extrinsic	
Rotation matrix ${}^W\mathbf{R}_G$ $[mm]$	$\begin{bmatrix} 0.297475 & 0.757544 & -0.581064 \\ 0.918031 & -0.059843 & 0.391966 \\ 0.262159 & -0.650035 & -0.713251 \end{bmatrix}$
Translation vector ${}^W\mathbf{t}_G$ $[mm]$	$\begin{bmatrix} -565.506248 & -490.705543 & 2984.873938 \end{bmatrix}^T$

extrinsic parameters determined the transformation between the coordinates of a pixel in the image and its spatial coordinates expressed in $\{W\}$.

Appendix E

Parameters of the consume mission case study

Table E.1: Main parameters for the *consume* mission experiments described in section 5.2.1.2, page 155.

Parameter	**Value**
Workspace	
Length x Width	50 m x 50 m
Robots	
Number of robots	$\{1,\ 2,\ 3,\ 4\}$
Axle length	0.5 m
Velocity (maximum)	1.4 $m.s^{-1}$
Acceleration (linear)	0.9 $m.s^{-2}$
Sensor range	2 m
Communication range	$\{0,\ 20\}$ m
Communication cost ($ccom$)	0.3125 bit^{-1}
Items	
Number of items	30
Ray	0.5 m
Consumption time (t_c)	100 s
Spacing	2 m
Minimum distance to initial pose	10 m
Obstacles	
Workspace coverage	$\{0,\ 5,\ 10\}$ %
Ray	$1 \ldots 4$ m
Spacing	3 m
$\mathbf{x} \in \mathcal{X}$	$\mathbf{cstate}(\mathbf{x})$
$\{Wander,\ Move\_To\_Help\}$	1 s^{-1}
$\{Acquire,\ Acquire\_To\_Help\}$	0.8 s^{-1}
$\{Consume,\ Help\_Consume\}$	2 s^{-1}
Performance weights	
$\{\alpha,\ \beta,\ \gamma,\ \delta\}$	$\{10^{-10},\ 0.4,\ 0.2,\ 0.4\}$

Appendix F

Inferential statistics and tests of hypotheses

This appendix gives the reader some background about inferential statistics, covering concepts such as Gaussian and t-Student probability distributions, level of significance, confidence intervals and hypothesis testing. This knowledge is used in section 6.7, page 204, to obtain useful conclusions from the statistical analysis of experimental results.

Most of the information presented in this appendix was extracted from [Ser05] and assumes the reader has some basic background about probability theory and descriptive statistics. If this is not the case, some bibliography covering these topics should be consulted before reading the appendix (*e.g.* [Pap91]).

By the end of the appendix, it is also presented detailed data about the tests of hypothesis referred in section 6.7.1.1, page 208.

F.1 Inferential statistics

Descriptive statistics is used to summarize or describe a sample extracted from a population, whereas *inferential statistics* is used to infer population's properties by generalizing the behavior observed in samples extracted from the population.

F.2 Gaussian distribution

The Gaussian distribution, also known as normal distribution, is defined by the equation (3.38), page 94. It is the most ubiquitous probability distribution in statistics, because

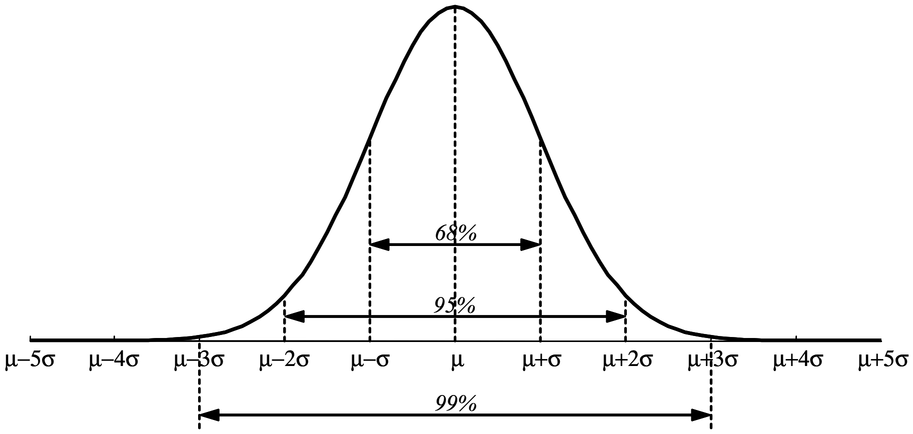


Figure F.1: Gaussian distribution: a non-skewed distribution around the mean μ, with standard deviation σ; 68% of the samples fall inside the interval $\mu \pm \sigma$, 95% of the samples fall inside the interval $\mu \pm 2\sigma$ and 99% of the samples inside in the interval $\mu \pm 3\sigma$.

it provides a convenient statistical model for many phenomena for which the random variable presents a non-skewed distribution around the mean (see Fig. F.1). Moreover, if the goal is to estimate the population mean μ from the mean value $\overline{x}$ of a set of samples extracted from that population, the Central Limit Theorem [Pap91] dictates that, if the sample's size is large enough, the sample's mean is a normally distributed random variable around μ.

If a random variable with a Gaussian distribution is sampled, accordingly with the curve represented in Fig. F.1, a sample falls inside the interval $\mu \pm \sigma_x$ with a probability equal to 68%, inside the interval $\mu \pm 2\sigma_x$ with a probability equal to 95%, and inside the interval $\mu \pm 3\sigma$ with a probability equal to 99% (see Fig. F.1).

F.3 Confidence intervals for a population mean

Although random selection imposes that all members of a population have an equal opportunity to be chosen, it does not guarantee that they are represented proportionally in a sample. For this reason, even when the sampling is properly random, the sample's characteristics (*e.g.* mean value and standard deviation) are usually different from the population's characteristics. This discrepancy is called *sampling error*.

F.3.1 Large samples

Consider a population with mean μ and standard deviation σ. Suppose that the goal is to infer the population mean μ from samples with size n, extracted from that population, by studying the distribution of the sample mean. The sample mean is computed as

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \tag{F.1}$$

and the sample standard deviation is computed as

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2}. \tag{F.2}$$

The Central Limit Theorem [Pap91] ensures that, if n is sufficiently large, the distribution of the sample mean tends to a Gaussian distribution with mean μ, being the standard deviation an inverse function of the sample size, as

$$\sigma_x = \frac{\sigma}{\sqrt{n}}. \tag{F.3}$$

This quantity is denoted as the *standard error of the mean*.

Since the population's standard deviation σ is generally unknown, the standard error of the mean is usually estimated through the sample standard deviation, as

$$s_x = \frac{s}{\sqrt{n}}. \tag{F.4}$$

The standard error of the mean allows to make inferences about the population mean because the proportions under the curve of the sampling distribution of means can be assumed equal to the proportions under the curve of the normal distribution. This assumption is valid when the sample size n is sufficiently large (typically for $n > 50$).

Given the Gaussian distribution of the sample mean $N(\overline{x}, s_x)$, accordingly with the Gaussian distribution definition, the population mean falls inside the interval $\overline{x} \pm s_x$ with a probability equal to 68%, inside the interval $\overline{x} \pm 2s_x$ with a probability equal to 95%, and inside the interval $\overline{x} \pm 3s_x$ with a probability equal to 99% (see Fig. F.1). Any of these intervals is denoted as a *confidence interval* for the population mean, with a given confidence level. Those are the confidence intervals for the population mean, with a confidence level equal to 68%, 95% and 99%, respectively ([1]).

[1] The presented critical values are approximate. For instance, for a confidence level equal to 95%, the critical value is more rigorously equal to 1.96 and the confidence interval is more rigorously equal to $\overline{x} \pm 1.96s_x$. The critical value, for a given confidence level, is usually obtained by consulting a table of critical values for a standardized Gaussian distribution (null mean and standard deviation equal to 1), in any statistics textbook.

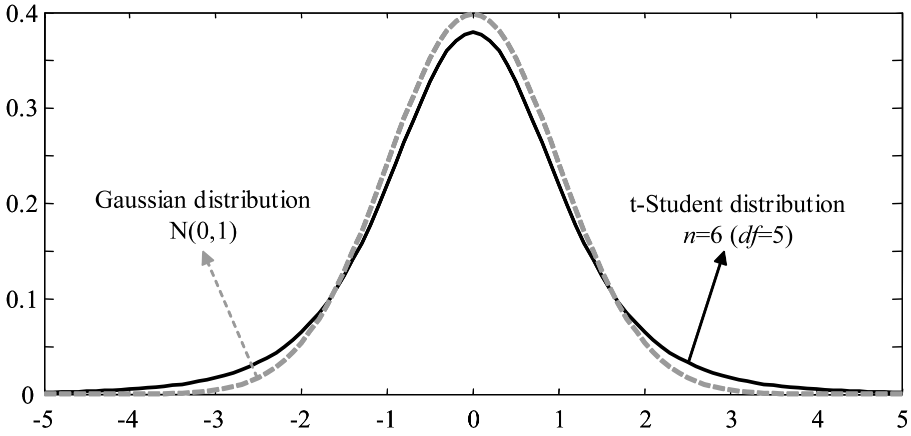


Figure F.2: *t*-Student versus Gaussian distribution: the t-Student distribution is more spread out than the Gaussian distribution and depends on the number of degrees of freedom *df*, which is computed from the sample size n as $df = n - 1$. The lesser *df* is, more spread out is the *t*-Student distribution.

F.3.2 Small samples: the *t*-Student distribution

Often, the assumption of large samples is not viable. With smaller samples, the distribution of the sample means tends to be more spread out. The *t*-Student distribution (see Fig. F.2) accounts for this effect and it depends on the sample size n through the number of degrees of freedom *df*, given by $df = n - 1$. Obviously, it approaches the Gaussian distribution if $n \to \infty$. Therefore, the *t*-Student distribution substitutes the Gaussian distribution when making inferences with small samples. The confidence interval for a given confidence level is slightly different from the one that would be obtained with the Gaussian distribution, being slightly wider.

The confidence level is associated with a given level of significance α, and is equal to $1 - \alpha$. The confidence interval for a given confidence level $1 - \alpha$ is given by

$$\overline{x} \pm t_{\alpha/2} \cdot s_x = \overline{x} \pm t_{\alpha/2} \cdot \left(\frac{s}{\sqrt{n}} \right),$$

wherein $t_{\alpha/2}$ is the two-tailed critical value of the *t*-Student distribution with $df = n - 1$ degrees of freedom, for a significance level equal to α (see Fig. F.3).

F.4 Hypothesis testing

If two different samples are compared, two different outcomes must be considered, concerning the mean of the populations from which they have been drawn: either the two

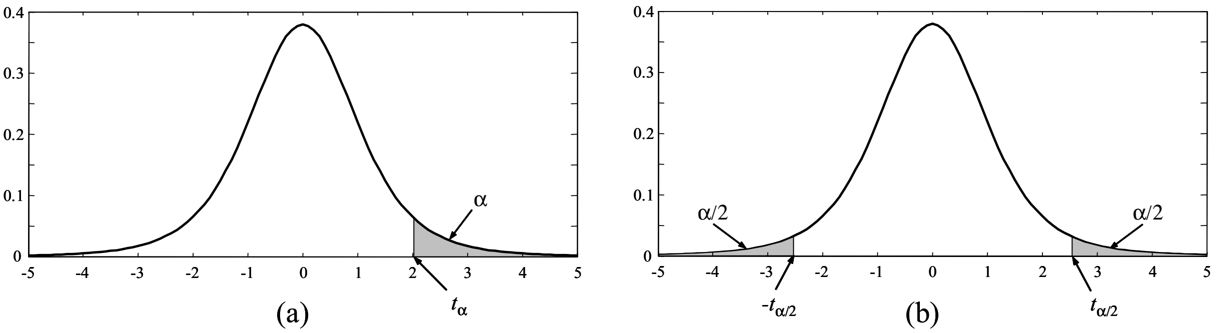


Figure F.3: Critical value of the t-Student distribution for a given confidence level $1 - \alpha$ (level of significance α): (a) one-tailed critical value t_α; (b) two-tailed critical value $t_{\alpha/2}$. In both cases, the sum of shaded areas in the tails is equal to α. In this example, $\alpha = 0.05$ and the t-Student distribution has $df = 5$ degrees of freedom.

samples represent truly different populations, or the differences are due to chance occurrence and they were obtained from the same population. The latter outcome is denoted as the *null hypothesis* H_0, whereas the former outcome is denoted as the *alternate hypothesis* or *research hypothesis* H_a.

The *null hypothesis* generally represents a theory about the values of a population (*e.g.* population mean μ), which the researcher usually wants to reject. The *alternate hypothesis* represents a theory that contradicts the null hypothesis, which the researcher usually wants to accept when sufficient statistical evidence exists to establish its truth.

Besides the two aforementioned hypothesis, any statistical test of hypothesis includes a test statistic, a rejection region and some assumptions. The test statistic is used to decide whether to reject the null hypothesis H_0. Given a confidence level, the rejection region defines the numerical values of the test statistic for which there is sufficient statistical evidence to reject the null hypothesis H_0. The assumptions define in what conditions the test can be applied.

F.4.1 Small-sample test for comparing the mean of two populations

The t-Student test of hypothesis for comparing the mean of two sampled populations is used when the test cannot be done using Gaussian distributions because of the small sample size.

Consider two samples from each population with mean $\overline{x}_1$ and $\overline{x}_2$, standard deviation

s_1 and s_2, and size n_1 and n_2, respectively. The sample mean is computed through equation (F.1) and the standard deviation is computed through equation (F.2).

F.4.1.1 One-tailed test

Given a constant Δ representing the difference between the two populations means ([2]), the hypotheses for a one-tailed test are stated as

$$H_0 : \quad \mu_1 - \mu_2 = \Delta, \tag{F.5}$$

$$H_a : \quad \mu_1 - \mu_2 > \Delta, \tag{F.6}$$

being the test statistic computed as

$$t = \frac{\overline{x}_1 - \overline{x}_2 - \Delta}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}. \tag{F.7}$$

Given a confidence level $1 - \alpha$, the rejection region is

$$t > t_\alpha, \tag{F.8}$$

wherein t_α is the associated one-tailed critical value (see Fig. F.3-a) of the t-Student distribution having

$$df = \frac{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}{\left(\frac{s_1^2}{n_1}\right)^2 \left(\frac{1}{n_1 - 1}\right) + \left(\frac{s_2^2}{n_2}\right)^2 \left(\frac{1}{n_2 - 1}\right)} \tag{F.9}$$

degrees of freedom ([3]).

If the critical value t_α falls in the rejection region, there is sufficient statistical evidence to reject the null hypothesis H_0 and the alternate hypothesis H_a can be accepted. Otherwise, the null hypothesis H_0 cannot be rejected, and nothing can be said about the alternate hypothesis H_a.

F.4.1.2 Two-tailed test

A two-tailed test is non-directional, because the hypotheses do not specify which mean is greater; they only specify whether they are equal. The hypotheses for a two-tailed test are stated as

$$H_0 : \quad \mu_1 = \mu_2, \tag{F.10}$$

$$H_a : \quad \mu_1 \neq \mu_2, \tag{F.11}$$

[2] Usually, the constant Δ is set to zero. In this case, the hypotheses are $H_0 : \mu_1 = \mu_2$ and $H_a : \mu_1 > \mu_2$.

[3] Since the value computed through equation (F.9) is generally not integer, it is rounded to the nearest integer.

being the test statistic computed as

$$t = \frac{\overline{x}_1 - \overline{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}. \tag{F.12}$$

Given a confidence level $1 - \alpha$, the rejection region is

$$t > t_{\alpha/2}, \tag{F.13}$$

wherein $t_{\alpha/2}$ is the associated two-tailed critical value (see Fig. F.3-b) of the t-Student distribution having the number of degrees of freedom computed through equation (F.9).

If the critical value $t_{\alpha/2}$ falls in the rejection region, there is sufficient statistical evidence to reject the null hypothesis H_0 and the alternate hypothesis H_a can be accepted. Otherwise, the null hypothesis H_0 cannot be rejected, and nothing can be said about the alternate hypothesis H_a.

F.4.2 Errors in hypothesis testing

Hypothesis testing have always two possible outcomes: reject or accept the null hypothesis. Both decisions are subject to some risk of being incorrect (see Table F.1). If the null hypothesis H_0 is rejected when it is true, a type I error is committed. Conversely, if H_0 is accepted when it is false, a type II error is committed.

The probability of committing a type I error is equal to the level of significance α. The *level of significance* can thus defined as the maximum risk tolerated by the researcher of rejecting the null hypothesis H_0 when it is indeed true. The compliment $1 - \alpha$ of the level of significance is the *confidence level*. Typical values for $1 - \alpha$ are 95% or 99%, *i.e.* $\alpha = 0.05$ or $\alpha = 0.01$, respectively.

The probability of committing a type II error, *i.e.* the probability of failing to reject the null hypothesis H_0, is denoted as β. The compliment $1-\beta$ is denoted as the *statistical power*. For instance, if $\beta = 0.2$ there is a chance of 20% of failing to reject H_0 when it is indeed false. Since β is usually unknown, when there is no statistical evidence to reject

Table F.1: Possible outcomes of a statistical test of hypothesis.

		Truth	
		H_0 is true	H_0 is false
Outcome	Reject H_0	Type I Error (α)	Correct
	Accept H_0	Correct	Type II Error (β)

the null hypothesis H_0, the tests of hypotheses do not prescribe to accept H_0 in that case and, thus, reject the alternate hypothesis H_a, because the researcher cannot evaluate the risk associated with the decision. Therefore, if there is no sufficient statistical evidence to reject the null hypothesis, the test of hypothesis is inconclusive.

F.5 Comparison of the uncoordinated and coordinated exploration methods

This section of the appendix presents the detailed data about the tests of hypothesis that led to the conclusions presented in section 6.7.1.1, page 208, about the comparison between the uncoordinated and coordinated exploration methods for building volumetric maps with teams of cooperative mobile robots.

Table 6.1, page 206, presents the values of the mission execution time by using both the uncoordinated and coordinated exploration methods. The mean of the former variable is denoted in this section as $t^u_{k_{max}}$, while the mean of the latter one is denoted as $t^c_{k_{max}}$ ([4]). An argument of these variables indicates the team size (*e.g.* $t^c_{k_{max}}(2)$, for 2 robots).

The small-sample statistical one-tailed test for comparing the mean of two populations was used to test statistically some hypotheses about the team's mission execution time when using both exploration methods, based on the experimental results contained in table 6.1, page 206. This type of test of hypothesis is described in section F.4.1, page 277, in this appendix.

Table F.2 presents the detailed data about all the tests of hypotheses that were computed to obtain the conclusions presented in section 6.7.1.1, page 208. Each row presents the research (alternate) hypothesis H_a, the value of the test statistic t, the number of degrees of freedom *df* of the t-Student distribution and, for three different confidence levels $1 - \alpha$, the critical value t_α of the t-Student distribution and the associated conclusion yielded by the test of hypothesis. In each test, the sample size n_1 and n_2 of both samples is always equal to 5. Note that there are cases wherein the test is inconclusive with a given confidence level, being however conclusive for a lower confidence level. These are tests having a lower statistical evidence.

[4] The superscripts "u" and "c" stand for *uccoordinated* and *coordinated*, respectively. In the case of a single robot, the superscript is not used because, obviously, it does not make sense to talk about coordination with just one robot.

Table F.2: Statistical tests of hypotheses for comparing both the uncoordinated and coordinated exploration methods.

			$1-\alpha=99\%$		$1-\alpha=95\%$		$1-\alpha=90\%$	
Research hypothesis H_a (*)	t	*df*	$t_{(0.01)}$	Conclusion	$t_{(0.05)}$	Conclusion	$t_{(0.1)}$	Conclusion
$t_{k_{max}}(1) > t^u_{k_{max}}(2)$	4.916	7	2.998	Accept H_a	1.895	Accept H_a	1.415	Accept H_a
$t_{k_{max}}(1) > t^c_{k_{max}}(2)$	12.631	8	2.896	Accept H_a	1.860	Accept H_a	1.397	Accept H_a
$t_{k_{max}}(1) - t^c_{k_{max}}(2) > 3600$	2.307	8	2.896	Inconclusive	1.860	Accept H_a	1.860	Accept H_a
$t_{k_{max}}(1) > t^u_{k_{max}}(3)$	3.067	5	3.365	Inconclusive	2.015	Accept H_a	1.476	Accept H_a
$t_{k_{max}}(1) > t^c_{k_{max}}(3)$	16.445	7	2.998	Accept H_a	1.895	Accept H_a	1.415	Accept H_a
$t_{k_{max}}(1) > t^u_{k_{max}}(4)$	0.753	5	3.365	Inconclusive	2.015	Inconclusive	1.476	Inconclusive
$t_{k_{max}}(1) - t^u_{k_{max}}(4) > -1800$	3.030	5	3.365	Inconclusive	2.015	Accept H_a	1.476	Accept H_a
$t^u_{k_{max}}(4) - t_{k_{max}}(1) > -1820$	1.549	5	3.365	Inconclusive	2.015	Inconclusive	1.476	Accept H_a
$t^u_{k_{max}}(5) > t^u_{k_{max}}(2)$	6.634	6	3.143	Accept H_a	1.943	Accept H_a	1.440	Accept H_a
$t^c_{k_{max}}(2) > t^c_{k_{max}}(4)$	1.464	8	2.896	Inconclusive	1.860	Inconclusive	1.397	Accept H_a
$t^c_{k_{max}}(2) > t^c_{k_{max}}(3)$	0.777	7	2.998	Inconclusive	1.895	Inconclusive	1.415	Inconclusive
$t^c_{k_{max}}(2) - t^c_{k_{max}}(3) > -600$	2.693	7	2.998	Inconclusive	1.895	Accept H_a	1.415	Accept H_a
$t^c_{k_{max}}(3) - t^c_{k_{max}}(2) > -900$	2.097	7	2.998	Inconclusive	1.895	Accept H_a	1.415	Accept H_a
$t^c_{k_{max}}(3) > t^c_{k_{max}}(4)$	0.354	4	3.747	Inconclusive	2.132	Inconclusive	1.533	Inconclusive
$t^c_{k_{max}}(3) - t^c_{k_{max}}(4) > -1800$	2.675	4	3.747	Inconclusive	2.132	Accept H_a	1.533	Accept H_a
$t^c_{k_{max}}(4) - t^c_{k_{max}}(3) > -2100$	2.354	4	3.747	Inconclusive	2.132	Accept H_a	1.533	Accept H_a
$t^u_{k_{max}}(2) > t^c_{k_{max}}(2)$	5.148	7	2.998	Accept H_a	1.895	Accept H_a	1.415	Accept H_a
$t^u_{k_{max}}(3) > t^c_{k_{max}}(3)$	4.350	5	3.365	Accept H_a	2.015	Accept H_a	1.476	Accept H_a
$t^u_{k_{max}}(4) > t^c_{k_{max}}(4)$	5.458	5	3.365	Accept H_a	2.015	Accept H_a	1.476	Accept H_a
$t^u_{k_{max}}(4) > t^c_{k_{max}}(2)$	4.750	5	3.365	Accept H_a	2.015	Accept H_a	1.476	Accept H_a
$t^u_{k_{max}}(3) > t^c_{k_{max}}(2)$	3.758	6	3.143	Accept H_a	1.943	Accept H_a	1.440	Accept H_a

(*) The null hypothesis H_0 is obtained from the research hypothesis H_a by substituting ">" for "=".

Errata

Page	Position	Correction
4	footnote	Replace “robot’s velocity” with “robot’s displacement”.
13	2nd paragraph	Insert “with a method” after “equipped with a range sensor”.
43	4th paragraph	Replace “discriminating obstacles from other obstacles” with “discriminating robots from other obstacles”.
76	4th paragraph	Replace “do not defined” with “did not define”.
119	5th paragraph	Replace “unoccupied or explored” with “unoccupied or unexplored”.
120	3rd paragraph	Replace “are assumed top be” by “are assumed to be”.
209	last paragraph	Replace “$\kappa = 0.5$” with “$\kappa = 0.25$”.