Chapter 2

PROJECT DIVA: GUIDANCE AND VISION SURVEILLANCE TECHNIQUES FOR AN AUTONOMOUS AIRSHIP

Alexandra Moutinho\(^1\), Luiz Mirisola\(^2\), José Azinheira\(^1\) and Jorge Dias\(^2\)

\(^1\)IDMEC/Instituto Superior Técnico, TULisbon
Av. Rovisco Pais, 1049-001 Lisbon, Portugal
\(^2\)ISR - Institute of Systems and Robotics, University of Coimbra, Portugal

Abstract

Unmanned Aerial Vehicles have a wide spectrum of potential civilian applications as observation and data acquisition platforms. Most of the aerial surveillance applications require low altitude, low speed platforms. The vehicle should ideally be able to hover above an area, allow long duration studies, take-off and land vertically without the need for runways. For such a scenario, lighter-than-air (LTA) vehicles are often better suited than airplanes and helicopters. This chapter introduces the Portuguese airship project named DIVA, and the approaches currently under development and that are being implemented in the areas of robotic integration, navigation and guidance, and vision based surveillance. The airship platform is described, along with an overview of the architecture developed, to document the practical experience gained in this field. Looking at the typical autonomous mission objective, we present a control approach for airship path-tracking, covering the whole flight envelope from hover to the normal cruise flight. An asymptotically stable backstepping controller is designed from the airship nonlinear dynamics and kinematics. Some practical issues are then considered and the control law is improved to take into account input saturations and wind disturbances, maintaining its asymptotic stability for a bounded wind estimation error. The presented simulation results illustrate the controller performance during a full realistic mission that covers all the usual tasks: vertical take-off and landing, stabilization and route path-tracking. Wind disturbances are also included. In vision systems used in aerial robotics, inertial and earth field magnetic sensors can provide valuable data.

\(^1\)E-mail address: moutinho@dem.ist.utl.pt
\(^2\)E-mail address: lgm@isr.uc.pt
\(^3\)E-mail address: jraz@dem.ist.utl.pt
\(^4\)E-mail address: jorge@isr.uc.pt
about the observer ego-motion, as well as an absolute orientation reference. Here, the inertial orientation measurements are used to compensate the rotational degrees of freedom in two different computer vision tasks: first, inertial data is used to project images on a leveled plane, relaxing the demands on interest point matching algorithms when performing image mosaicing; second, in the rotation-compensated, pure translation case, full homographies are reduced to planar homologies, and the heights over the ground plane on two views are calculated more accurately. Visual odometry for the airship 3D trajectory is performed by calculating the focus of expansion during the motion. These results are important for context of vision-based aerial mapping and visual surveillance. A technique is also presented that allows visual tracking of dynamic objects moving in the ground. Experimental results are presented, illustrating the approach and some achievements.

1. Introduction

Besides their use as military surveillance platforms, Unmanned Aerial Vehicles (UAVs) have a wide spectrum of potential civilian applications as observation and data acquisition platforms. They can be utilized in several environmental monitoring applications related to biodiversity, ecological and climate research and monitoring. Inspection oriented applications cover different areas such as mineral and archaeological prospecting, agricultural and livestock studies, crop field prediction, and use surveys in rural and urban regions, and also inspection of manmade structures such as pipelines, power transmission lines, dams and roads. UAV gathered data can also be used in a complementary way to acquiring information obtained by satellites, balloons, manned aircraft or on ground. Detailed UAV applications scenarios and road maps are presented in [1] and [2] from civil and military perspectives, respectively.

Most of the applications before cited have profiles that require maneuverable low altitude, low speed airborne data gathering platforms. The vehicle should ideally be able to hover above an area, present extended airborne capabilities for long duration studies, take-off and land vertically without the need of runway infrastructures, have a large payload to weight ratio, among other requirements. For this scenario, lighter-than-air (LTA) vehicles are better suited than airplanes and helicopters [3], mainly because: they derive the largest part of their lift from aerostatic, rather than aerodynamic forces; they are safer and, in case of failure, present a graceful degradation; they are intrinsically of higher stability than other platforms.

In this context, the Portuguese Project DIVA - Dirigível Instrumentado para Vigilância Aérea - was proposed. DIVA focuses on the establishment of the technologies required to substantiate autonomous operation of unmanned robotic airships for environmental monitoring and aerial inspection missions. This includes sensing and processing infrastructures, control and guidance capabilities, and the ability to perform mission, navigation, and sensor deployment planning and execution.

Other important researches related to outdoor autonomous airships in the world at this moment are the AURORA Project [4] in Brazil, sharing a partnership with the DIVA Project, the Lotte Project [5] at Germany, the French projects at LAAS-CNRS [6, 7], and LSCUniversité d'Evry [8]. In the USA there is a partnership between the projects of STWing-SEAS [9] of University of Pennsylvania and the EnviroBLIMP at CMU.

Aiming at the autonomous airship goal, aerial platform positioning and path-tracking should be assured by a control and navigation system. Such a system needs to cope with the highly nonlinear and underactuated airship dynamics, ranging from hovering flight (HF) to cruise or aerodynamic flight (AF). Hovering flight is defined here as a flight in low airspeed condition. In addition, the abrupt and continuous transition between the HF and AF in the dynamics, and the different use of actuators necessary within each region, makes that a very difficult issue to be dealt with by the control scheme.

Basically, two main approaches can be considered for the automatic control and navigation system of an airship. The first one relies on the linear control theory to design individual compensators to satisfy closed-loop specifications, based on linearized models of the airship dynamics. One important result of the linearization approach is the separation of two independent (decoupled) motions: the motion in the vertical plane, named longitudinal, and the motion in the horizontal plane, named lateral. Following this approach, experimental results were obtained for the AURORA airship for path following through a set of pre-defined points in latitude/longitude, along with an automatic altitude control [10].

Also based on a linearized airship model, Wimmer et al. [5] introduced a robust controller design method to compensate for the lack of knowledge about the airship airship dynamic behavior and model parameters. The decoupled longitudinal and lateral control systems both consist of an inner $H_{in}$-controller for the dynamics and an outer SISO P- or PI-controller for the remaining states. Experimental results are shown therein for the pitch and velocity control. We remark that, as far as the authors are aware, both experimental results (from Lotte and AURORA Projects) on automatic control for outdoor airships are the only ones reported in the literature at this moment. For the lateral control problem, an alternative $H_{in}$-approach for the airship heading control is proposed in [11], and a $H_2$/$H_{in}$ approach for the design of a lateral PD-PI controller for the AURORA airship is proposed in [12]. Other works in the AURORA Project following this methodology of linear based controllers can be found in [3, 10, 13].

The second approach for the airship automatic control system consists on the search for a single global control scheme covering all the aerodynamic range, such that the different flight regions, from HF to AF, are considered inside a sole formulation. For security reasons, as well as simplicity and flexibility, a global nonlinear control is more interesting than a linearized and decoupled one. At present, Backstepping is the main nonlinear approach under investigation for the AURORA airship [14]. Other important nonlinear approaches for the UAV control are the Sliding Mode technique [15, 16] and Dynamic Inversion (or Feedback Linearization) [17, 18].

In the Backstepping approach [19], a Lyapunov-based technique, by formulating a scalar positive function of the system states and then choosing a control law to make this function decrease, we have the guarantee that the nonlinear control system thus designed will be asymptotically stable, and still robust to some unmatched uncertainties. In a previous work [14], a Backstepping control strategy for the stabilization of the AURORA airship has been proposed. It introduced a synthetic modeling of the airship dynamics, resulting in an original formulation of the system kinematics and dynamics with an appropriate change of variables allowing the application of Backstepping techniques for the design of the UAV stabilization control. The saturation of the control signals was also considered in the design, since at low airspeeds, or when in hovering state, the airship is usually underactuated. In
order to cope with limitations due to reduced actuation, the idea of Teel [20] is followed. It uses a nonlinear combination of saturation functions of linear feedbacks that globally stabilizes a chain of integrators. The problem of actuator limits is of fundamental interest for this kind of underactuated systems, making this a theoretical study closely related to practical applications. In addition, a guidance strategy was proposed to deal with the airship lateral underactuation in face of wind disturbances.

Based on this work [14], we now present a Backstepping controller applied to the DIVA airship model, and where the kinematics are described by the Euler angles in place of the quaternions used previously. Also, the control law applicability range is broadened so it is valid over the entire flight envelope and not only for stabilization purposes.

Other successful applications of the Backstepping approach for UAV control can be found in [6, 21, 22]. A Backstepping technique has been proposed by the LAAS/CNRS autonomous blimp project [6, 7]. The global control strategy studied is obtained by switching between four sub-controllers, one for each of the flight phases considered. Each controller is however still based on linearized models of the airship, which leads once again to the separate control of the longitudinal and lateral motions. Also considering the nonlinear control techniques, Beji et al. [23] introduced a Backstepping tracking feedback control for ascent and descent flight maneuvers, where the objective is to stabilize the airship engine around trimmed flight trajectories. The treatment of the actuators saturation is also considered in the Backstepping design of Freeman and Praly [24], so that the boundedness of the control signals and its derivative is propagated through each step of the recursive design. Finally, Metin et al. [25] follow the same idea for an UAV with orientation limits.

The trajectory of a mobile observer can be recovered from images of a planar surface using interest point matching and the well known planar homography model. The recovered homography matrix is then decomposed into the rotational and translational motion parameters, yielding four possible solutions among which two can not be immediately discarded. Various geometric constraints have been proposed to recover the right motion. This was already performed for an airship using clustering and blob-based interesting point matching algorithms and building a image mosaic to improve the trajectory estimate [26].

The trajectory of an UAV can also be recovered by tracking known fixed targets on the ground, what requires modifying the environment [27]. With an on-board stereo camera, the height over the ground plane can be recovered with efficient sparse stereo techniques, although subject to limitations in range related to the stereo baseline size [28].

Aerial vehicles have been utilized to produce 3D maps of the ground using a variety of different sensors. For example, stereo images taken by a remotely controlled blimp were used to build a dense 3D map of the ground surface, in the form of a DEM (Digital Elevation Map), and also performing localization by tracking the position of automatically detected landmarks on the ground [29]. Stereo imagery has also been combined with other vision techniques such as color segmentation [30]. Airborne range sensing devices such as laser range finders or radars have also been extensively used to build 3D maps actually exploited in domains such as geology [31]. Statistical techniques such as Markov Random Fields have been applied on such DEMs to reduce noise, in various levels including pixel-to-pixel graphs and larger segmented regions [32].

This chapter introduces the Portuguese DIVA airship project, and the approaches currently under development and that are being implemented in the areas of robotic integration, navigation and guidance, and vision based surveillance.

Looking at the typical autonomous mission objective, and after this introductory section, the remaining parts of this chapter are organized as follows.

Section 2 presents the DIVA airship, and lists the requirements for the on-board and ground station systems, providing a development road-map for the experimental platform utilized to obtain the current results. An overview of the system developed so far is also described.

Section 3 is dedicated to the guidance and control objectives. It describes the airship dynamics and kinematics under constant wind and presents a control approach for airship path-tracking, covering the whole flight envelope from hover to the normal cruise flight. An asymptotically stable backstepping controller is designed from the airship nonlinear equations. Some practical issues are then considered and the control law is improved to take into account input saturations and wind disturbances, maintaining its asymptotic stability for a bounded wind estimation error. The control allocation problem is focused as well as the reference shaping to deal with the airship underactuation. It presents simulation results illustrating the controller performance during a full realistic mission that covers all the usual tasks: vertical take-off and landing, stabilization and route path-tracking. Wind disturbances are also included.

Section 4 is dedicated to visual perception. Trajectory recovery from images is discussed, exploiting the inertial orientation measurements to separate rotational and translational components, and using a pure translation movement model. The approach followed uses a monocular camera and does not require artificial targets on the ground. Next, rotation compensated images projected on the horizontal ground plane are further exploited to build a coarse Digital Elevation Map (DEM), performing 3D mapping from monocular aerial images and with a very fast process after pixel correspondences are found. Finally, section 5 stresses the concluding remarks.

2. DIVA Prototype

The DIVA is a nonrigid airship (see Fig. 1(a)) with 9.47m long, 1.97m diameter and with a volume of 18m^3. Its payload capacity is approximately 147kg, and the maximum speed attained is around 70km/h.

The airship actuators input is given by $U = \{T_x, \delta_x, T_D, T_y, \delta_x, \delta_e, \delta_r\}^T$ (see Fig.1(b)), where $T_x$ is the total main propellors thrust, $\delta_x$ is the vectoring angle, $T_D$ is the difference between right and left thrust, $T_y$ is the stern propeller lateral thrust, and $\delta_x, \delta_e, \delta_r$ are the tail surfaces deflections, corresponding to aileron, elevator and rudder, respectively. The aileron input $\delta_x$ is generated through the opposite deflection of each of the fins yielding a rolling moment.

2.1. Technical Characteristics of the Airship

This section details the technical characteristics of the system which collected the data used on the experiments shown in section 4. The system is composed by an embedded system and a ground station, which communicate through a wireless link.
2.1.1. Embedded System

Operational Requisites

- Periodically, the embedded system must read data from all sensors and transmit the data to the ground station, with a period compatible with the vehicle dynamics.

- The embedded system must integrate at least one camera, capturing images with sufficient resolution and frame rate. A modern camera with automatic gain adjustment is necessary as illumination conditions change often during the flight.

- The embedded system must store all telemetry data, besides transmitting them to the ground station, to avoid losing data if the data link is lost. Camera images also are stored (the heaviest burden to the CPU and storage system).

- If DGPS (Differential GPS) corrections are needed, the embedded system must receive DGPS data from the ground station via a reliable data link. GPS readings when DGPS data is intermittent can often “jump” between corrected and uncorrected states, which may be worse than having no DGPS correction at all. A separate low speed link for DGPS data may be advisable.

- Payload weight is a severe limitation for airships. Thus, energy consumption must be minimized to decrease battery weight.

Safety Characteristics

- The embedded system must have a Remote Control (RC) receiver and allow a human pilot to manually pilot the airship.

- There must be a device electrically independent of the CPU to read the servo commands from the RC receiver, able to continue working in case of CPU malfunctioning. It receives commands from the RC receiver, and relays them to the servo motors and to the CPU to be read and stored.

- The embedded system must be sufficiently resistant to vibration and tilting to resist the flight and motor-induced vibration. Vibration isolation (lightweight) may be necessary. Vibration-resistant data storage is needed to store imagery.

2.1.2. Ground Station

This section presents the technical characteristics of the ground station.

Data Collection And Storage

- Although the embedded system stores all state variables, the data may not be recoverable due to accidents or malfunctioning. Therefore, the ground station should receive and store data from the embedded system.

- The ground station must be easily reconfigurable if there is a change in the data format sent by the embedded system (e.g., if a new sensor is added).

- The stored data must be easily converted to a format readable by commercial mathematical software such as MATLAB® (e.g., an ASCII format).

User Interface: Vehicle Safety And Monitoring

The ground station must monitor the vehicle state not only to detect hazardous situations (safety issues), but also to avoid useless flights and waste of time in case of malfunctioning. The human pilot and the algorithm developers should determine which variables should be monitored.

- Monitor all critical state variables, i.e., the ones which are essential to the flight safety or data recording, like tachometers, GPS, camera status, etc. The user interface must show the state of the most critical variables clearly and continuously (but not necessarily the actual numeric value of all of them), indicating critical failures with alarms (red lights and/or sounds).

- This monitoring must be active before take-off, so that the ground station operator can abort the flight if a critical system is not operating.

2.2. Overview of Architecture

This section provides an overview of the system architecture developed and utilized to obtain the datasets used on this chapter, as well as parts still in development. Remotely piloted flights were performed with telemetry and image recording with the on-board hardware architecture shown in Fig. 2. The images and sensor data were time-stamped immediately after reading with the same CPU clock.

The C++ embedded software includes a main loop to read, store and transmit sensor data, and threads to capture and store images. CORBA middleware [33] is used for the data transmission, avoiding manual data marshaling and allowing fast reconfiguration when the telemetry format is changed, as when a new sensor is added.
CPU The data sets used on this chapter were collected with a M570-BAB Board; VIA EDEN 600MHZ processor; PC104+; 512MB RAM. Nevertheless, an upgrade is necessary to be able to store images at a higher frame rate and to transmit images to the ground.

Inertial System (Xsens MTB-9 system) Outputs direct readings from its internal sensors (magnetometer, accelerometer and inclinometer), and also filtered, driftless absolute orientation. Low weight (35g).

GPS Receptor (Garmin GPS35 12 channel GPS unit) Low weight GPS receptor and antenna integrated in a single mouse-sized package.

Wind Sensor It is mounted on the airship nose and measures barometric altitude, the angle of attack, sideslip angle and wind speed. It is not yet calibrated.

RPM Sensor Measures the rotation speed of the motors, by counting the number of interruptions on an Infra Red light signal that is cut by the propeller movement.

Switch Board The servo-motors on the flaps and motors accelerators are commanded by Pulse Width Modulation (PWM) signals sent by the human pilot via RC. The Switch Board reads the signals from the Radio Control Receiver, transmits their values in digital form to the CPU, for recording, and retransmits PWM signals to the servo motors. In the automatic flights, the same board may switch to transmit commands sent from the CPU to the servo motors.

Besides the on-board system, there is also support equipment on the ground (see Fig. 4). The human pilot commands the airship with a standard aero model Remote Control Unit, sending PWM signals to command the servo-motors on the flaps and motor accelerators. Also, a laptop connected to a Wireless Access Point receives, stores and displays telemetry data from the airship.

3. Guidance and Control

This section is dedicated to the guidance and control problem of the DIVA airship. After presenting its dynamic model, a backstepping path-tracking controller is designed. Con-

---

Figure 2. The embedded system of the DIVA project.

Figure 3 shows hardware components used in the DIVA airship, some of which are described below:

(a) Xsens MTB-9 IMU  (b) Garmin GPS35  (c) Wind sensor on the airship nose

(d) The embedded system.

Figure 3. Hardware components of the DIVA prototype.
3.1. Airship Dynamic Model

This section describes the dynamic and cinematic equations of the DIVA airship. It also presents an estimator for the unknown wind disturbances.

3.1.1. Airship Dynamics

For the derivation of the mathematical model of the DIVA airship flight dynamics, based on the AURORA airship model, the following aspects were taken into account [34, 35]: (i) the model considers the airship virtual masses and inertias due to the large volume of air displaced by the airship; (ii) the airship motion is referenced to a system of orthogonal axes fixed to the vehicle (Fig. 5) whose origin is the Center of Volume (CV), assumed to coincide with the gross Center of Buoyancy (CB); (iii) the airship is assumed to be a rigid body, so that aerodynamic effects are ignored.

![Figure 5. Airship local reference frame.](image)

Let \{i\} represent the inertial frame (which, for simplicity, is considered coincident with the geographical North-East-Down (NED) frame), \{f\} be the body-fixed coordinate frame, and \( S \in SO(3) := \{ S \in R^{3 \times 3} : SS^T = I, \det(S) = +1 \} \) be the rotation matrix from \{i\} to \{f\} frame. In the \{f\} frame, the airship linear and angular velocities are given by \( v \in R^3 \) and \( \omega \in R^3 \), respectively. Let us recall the dynamic equation as it was deduced in [35], using the formulation of Newton second law expressed in the local frame \{f\}:

\[
M \ddot{v} = F_k + F_w + F_g + F_p + F_a
\]

where \( M \in R^{6 \times 6} \) is the generalized mass and inertia matrix, \( V = [v^T, \omega^T]^T \in R^6 \) is the inertial velocity in the local frame, and the generalized forces appearing in the right hand side are respectively the kinematic, wind induced, gravity, propulsion and aerodynamic forces.

If we assume the wind as constant in the earth frame, with linear inertial velocity \( w \) and without angular component, the local linear velocity \( v \) may be written as the sum of the air velocity \( v_a \), which is the relative velocity of the vehicle in the air flow (associated with the airspeed), and the wind velocity in local frame \( v_w \):

\[
v = v_a + v_w = v_a + Sw
\]

(2)

Defining the air velocity state as \( x = [v^T, \omega^T]^T \in R^6 \), the airship dynamic equation may finally be written in a compact form as a function of the air velocity [36]:

\[
M \ddot{x} = -\Omega_6 M \dot{x} + E_6 S g + F + f
\]

(3)

where \( \Omega_6 \triangleq \text{diag}(\Omega_3, \Omega_3) \in R^{6 \times 6} \), and the synthetic matrix notation is used for the cross-product \( \Omega_3 = \omega \times \in R^{3 \times 3} \).

Also, \( g \) is the gravity vector in the inertial frame, and \( E_6 \triangleq \begin{bmatrix} \omega^T & 0 \end{bmatrix} \), where \( \omega \) is the airship scalar mass and \( m_w \) is its weighting mass. The forces vector corresponds to \( F + f = \begin{bmatrix} F_p + F_a \\ T_p + T_a \end{bmatrix} \) separating \( F = F(x) \), for the state only depending part, and \( f \) for the actuation or control force input, with an aerodynamic part in \( (F_a, T_a) \), and a propulsion part in \( (F_p, T_p) \).

3.1.2. Airship Kinematics

Let us define the airship position vector \( \eta = [p^T, \phi]^T \in R^6 \) as being composed by its Cartesian coordinates \( p \in R^3 \) in the \{i\} frame, and the angular Euler attitude \( \phi \in R^3 \). The kinematics involves the transformation between velocity and position [37]:

\[
\dot{p} = S^T v
\]

(4a)

\[
\dot{\phi} = R \omega
\]

(4b)

with the coefficient matrix \( R \in R^{3 \times 3} \) relating the Euler angles with their derivatives and the angular rates.

The position derivative is related to the airship velocity as:

\[
\eta = \begin{bmatrix} \dot{p} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} S^T & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} S^T & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} v_a + Sw \\ \omega \end{bmatrix}
\]

(5)

We may write (5) as:

\[
\dot{\eta} = \begin{bmatrix} S^T & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} v_a \\ \omega \end{bmatrix} + \begin{bmatrix} I \\ 0 \end{bmatrix} \omega
\]

(6)

with \( I \) the identity matrix, or:

\[
\dot{\eta} = J \dot{x} + B \omega
\]

(7)
where \( J \triangleq \begin{bmatrix} S^T & 0 \\ 0 & R \end{bmatrix} \in \mathbb{R}^{6 \times 6} \) and \( B \triangleq I, 0 \)\( ^T \in \mathbb{R}^{6 \times 3} \).

The system may then be expressed by the following equations describing the dynamics, kinematics and constant wind:

\[
\begin{align*}
\dot{z} &= Kx + M^{-1} (E_g Sg + F + f) \quad (8a) \\
\dot{\eta} &= Jx + Bw \\ 
\dot{w} &= 0 \quad (8c)
\end{align*}
\]

where \( K \triangleq -M^{-1} \Omega M \in \mathbb{R}^{6 \times 6} \) is linearly dependent of the angular velocity \( \omega \), whereas \( M \) is constant or slowly varying with altitude (since the inertia terms depend on the air density).

The kinematical derivative relations satisfy the equations:

\[
\begin{align*}
\dot{S} &= -\Omega S \\ 
\dot{R} &= -RR^{-1}R \\ 
\dot{J} &= JH
\end{align*}
\]

with \( H \triangleq \begin{bmatrix} \Omega & 0 \\ 0 & -R^{-1}R \end{bmatrix} \in \mathbb{R}^{6 \times 6} \).

### 3.1.3. Wind Estimator

Since the wind disturbance is unknown, an estimator may be built based on (8b)-(8c). As assumed earlier, the wind input is not affecting the angular position part in (6), and therefore only the cartesian position \( p \) of the airship should be considered:

\[
\dot{p} = S^T v_a + w \quad (10)
\]

The estimator states may then be \((\hat{p}, \hat{w})\), and its dynamics may be chosen as:

\[
\frac{d}{dt} \begin{bmatrix} \hat{p} \\ \hat{w} \end{bmatrix} = \begin{bmatrix} S^T v_a \\ 0 \end{bmatrix} + \begin{bmatrix} L_p & I_s \\ L_w & 0 \end{bmatrix} \begin{bmatrix} p - \hat{p} \\ w - \hat{w} \end{bmatrix} \quad (11)
\]

leading to a dynamics of the estimation error \( \epsilon \) obtained from (10)-(11) and given by:

\[
\dot{\epsilon} = \frac{d}{dt} \begin{bmatrix} p - \hat{p} \\ w - \hat{w} \end{bmatrix} = \begin{bmatrix} -L_p & I_s \\ -L_w & 0 \end{bmatrix} \epsilon = A_{\epsilon} \epsilon \quad (12)
\]

where the two constant matrices \((L_p, L_w)\) are chosen so that \( A_{\epsilon} \) be Hurwitz.

Then there exists a positive definite symmetric matrix \( P_{\epsilon} > 0 \) such that:

\[
\frac{d}{dt} (\epsilon^T P_{\epsilon} \epsilon) = -\epsilon^T Q_{\epsilon} \epsilon \quad (13)
\]

where the matrix \( Q_{\epsilon} > 0 \) is symmetric positive definite, and chosen in a block diagonal form. The matrix \( P_{\epsilon} \) is the solution of the Riccati equation:

\[
A_{\epsilon}^T P_{\epsilon} + P_{\epsilon} A_{\epsilon} = -Q_{\epsilon} = \begin{bmatrix} Q_p & 0 \\ 0 & Q_w \end{bmatrix} \quad (14)
\]

with \( Q_p \) and \( Q_w \) diagonal matrices, so that the Lyapunov function of the estimator and its derivative are:

\[
\begin{align*}
W_\epsilon &= \epsilon^T P_{\epsilon} \epsilon \\
\dot{W}_\epsilon &= -\epsilon^T Q_{\epsilon} \epsilon = -\epsilon^T Q_p \epsilon - \epsilon^T Q_w \epsilon
\end{align*}
\]

where the estimation errors are:

\[
\begin{align*}
\hat{p} &= p - \hat{p} \\
\hat{w} &= w - \hat{w}
\end{align*}
\]

The asymptotically stable wind estimator \((\hat{W}_\epsilon < 0)\) will then assure that the estimation errors (17)-(18) will converge to zero.

### 3.2. Path-Tracking Controller Design

#### 3.2.1. Backstepping Design Approach

Let us consider a generic control problem with output \( y \). We first define two auxiliary outputs involving the output \( y \) and its derivative \( \dot{y} \):

\[
\begin{align*}
y_1 &= ay + \dot{y} \\
y_2 &= \dot{y}
\end{align*}
\]

where \( a \) is a positive scalar to be used as design parameter. It is easily seen that when both auxiliary outputs are taken to the origin, the regulation of the main output \( y \) is then achieved.

A tentative Lyapunov function may be:

\[
W_0 = \frac{1}{2} y_1^T y_1 + \frac{1}{2} y_2^T y_2
\]

Its derivative is:

\[
\dot{W}_0 = y_1^T \dot{y}_1 + y_2^T \dot{y}_2 = (ay + \dot{y})^T (ay + \dot{y}) + y^T \dot{y} = (ay + 2\dot{y})^T (ay + \dot{y}) - ay^T \dot{y}
\]

If the control is chosen in order to give:

\[
ay + \dot{y} = -\Lambda (ay + 2\dot{y})
\]

where \( \Lambda = A^T \) is a positive definite matrix, then the derivative:

\[
\dot{W}_0 = -(ay + 2\dot{y})^T \Lambda (ay + 2\dot{y}) - ay^T \dot{y}
\]

will clearly be negative definite and the system will be globally asymptotically stable.
3.2.2. Backstepping Design Applied to Path-Tracking

We shall now proceed applying the control design described in the previous section to the path-tracking problem.

Let us assume a point \( p_r \) with a constant ground velocity \( v_r \) is to be tracked with constant attitude along a rectilinear path AB (see Fig. 6):

\[
p_r = S_r^T v_r
\]

where \( S_r \in \mathbb{R}^{3\times3} \) is the constant transformation matrix from the inertial frame to the desired path.

![Figure 6. Air velocity reference estimation (2D).](image)

As the wind velocity \( v_w \) is considered, the desired air velocity \( v_{ar} \) may be deduced from the desired ground velocity \( v_r \). Moreover, since the airship is to align itself with this air velocity, we have a reference for the attitude given by the transformation \( S_{ar} \) from the inertial frame to the air velocity \( v_{ar} \), which is described by the desired attitude vector \( \phi_{ar} \).

This leads to the reference position \( \eta_r = [y_r^T, \phi_{ar}^T]^T \).

The derivative of this reference position is:

\[
\dot{\eta}_r = \begin{bmatrix}
S_r^T v_r \\
0
\end{bmatrix} = J_r x_r
\]

where the reference velocity state is \( x_r = \begin{bmatrix} v_r \\ 0 \end{bmatrix} \in \mathbb{R}^6 \) and \( J_r = \begin{bmatrix} S_r^T & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{6\times6} \).

Note that, although we have assumed a rectilinear reference path, the approach may also be extended to the cases where the reference path varies slowly, with negligible derivatives when compared to the state derivative.

Let us now consider a candidate Lyapunov function similar to (20):

\[
W_1 = \frac{1}{2} y_1^T y_1 + \frac{1}{2} y_2^T y_2
\]

where the output auxiliary variables \( y_1 \) and \( y_2 \) are again derived from the output \( y \) and its derivative \( \dot{y} \), but where \( y = \eta - \eta_r \) is the position tracking error:

\[
\begin{align*}
\dot{y}_1 &= \dot{y} + \ddot{y} = a(\eta - \eta_r) + \dot{\eta} - \eta_r = a(\eta - \eta_r) + J x + B w - J_r x_r \\
\dot{y}_2 &= \ddot{y} - \dot{\eta}_r = J x + B w - J_r x_r
\end{align*}
\]

and where equations (8b) and (25) were used.

The derivative of the tentative Lyapunov function (26) is then:

\[
\dot{W}_1 = y_1^T \ddot{y}_1 + y_2^T \ddot{y}_2 = (a y + \dot{y})^T (a y + \ddot{y}) + y^T \ddot{y} = (a y + 2 \ddot{y})^T (a y + \ddot{y}) - a y^T y
\]

If the control is chosen such that:

\[
a y + \ddot{y} = -\Lambda (a y + 2 \ddot{y})
\]

or:

\[
a (J x + B w - J_r x_r) + J \dot{x} + J H x = -\Lambda (a (\eta - \eta_r) + 2(J x + B w - J_r x_r))
\]

where we used:

\[
y_2 = \ddot{y} = J \dot{x} + J H x
\]

this leads to the control law:

\[
J \dot{x} = -\Lambda (a (\eta - \eta_r) + 2(J x + B w - J_r x_r)) - J H x - a (J x + B w - J_r x_r)
\]

As wind is estimated, the suggested control law is:

\[
J \dot{x} = -\Lambda (a (\eta - \eta_r)) - J H x - (a I + 2\Lambda (J x + B w - J_r x_r))
\]

and (29) should be rewritten as:

\[
a y + \ddot{y} = -\Lambda (a y + 2 \ddot{y}) + (a I + 2\Lambda) B \ddot{w}
\]

Introducing \( y_0 = \begin{bmatrix} \lambda \end{bmatrix} y_0 + \lambda \ddot{y} \), and defining \( G \in \mathbb{R}^{6\times3} \) such that \( G \cong (\frac{\lambda}{\lambda^*} \Lambda^{-1} + I) B \), the tentative Lyapunov derivative appears as:

\[
\dot{W}_1 = -\Lambda y_0^T (y_0 - 2G \ddot{w}) - a y_2^T y_2
\]

or, completing the squares:

\[
\dot{W}_1 = -\Lambda(y_0 - G \ddot{w})^T (y_0 - G \ddot{w}) + \Lambda \ddot{w}^T G^T G \ddot{w} - a y_2^T y_2
\]

If we now consider a corrected tentative Lyapunov function with the wind estimator:

\[
W = W_1 + W_e
\]

the derivative \( W \) may be written using (36) and (16) as:

\[
W = -\Lambda(y_0 - G \ddot{w})^T (y_0 - G \ddot{w}) - a y_2^T y_2 - \ddot{w}^T (Q_\ddot{w} - \Lambda G^T G) \ddot{w}
\]
which is definite negative if:

$$Q_w - \Lambda G^T G > 0$$  \hspace{1cm} (39)$$

The control law may be deduced from equations (8a) and (33), leading to:

$$f = M(\dot{z} - Kz) - E_p S g - F$$  \hspace{1cm} (40)$$

$$\dot{z} = -aJ^{-1}(\eta - \eta_t) - HZ - J^{-1}\Lambda_2^2 (Jx + B\hat{w} - J_t x_r)$$  \hspace{1cm} (41)$$

where $\Lambda_2^2 = (aI + 2A)$. The force control input is then given by:

$$f = -M[\begin{bmatrix} A_1 (Jx + B\hat{w} - J_t x_r) + B_t (\eta - \eta_t) + \Gamma_1 x_t \end{bmatrix} - E_p S g - F$$  \hspace{1cm} (42)$$

with $A_1 = J^{-1}\Lambda_2^2$, $B_t = aJ^{-1}A_1$ and $\Gamma_1 = H + K$, resulting in an asymptotically stable closed-loop system.

However, the force control input, as it is, may result in excessively high demands for a real system subject to input constraints. In the next section the control solution (42) will be adapted to deal with this matter.

3.2.3. Control Design with Saturation Constraints

In order to include saturation limits into the control design, let us rewrite equation (29), corresponding to a second derivative demand:

$$\ddot{y} = -a\dot{y} - \Lambda (a\dot{y} + 2\dot{y}) = -aI + 2A) \dot{y} - a\Lambda y = -\Lambda_2^2 \dot{y} - \Lambda_1 y$$  \hspace{1cm} (43)$$

with $A_1 = aA$ and $\Lambda_2^2$ as defined in the previous section.

Defining the second Lyapunov function as:

$$W_2 = \frac{1}{2} y_2^T y_2$$  \hspace{1cm} (44)$$

with, as before, $y_2 = \dot{y}$, its derivative may be expressed as:

$$\dot{W}_2 = y_2^T \ddot{y} = -\dot{y}^T (\Lambda_2^2 \dot{y} + \Lambda_1 y) = -\dot{z}_2^T (z_2 + z_1)$$  \hspace{1cm} (45)$$

where $z_1 = \Lambda_2^2 \dot{y}$ and $z_2 = \dot{z}_2 \dot{y}$. Writing (43) as function of $z_1$ and $z_2$ yields:

$$\ddot{y} = -\Lambda_2 (z_2 + z_1)$$  \hspace{1cm} (46)$$

Before proceeding, we will now define linear saturation as well as its properties, and provide an important theorem used in the proof of stability of the saturated control.

**Definition 1.** As a particular case and extension of the linear saturation definition proposed by Teel [20], let us introduce the element-wise nondecreasing saturation function $\sigma : \mathbb{R}^n \rightarrow \mathbb{R}^n$, defined by a vector $m$ of $n$ positive values $m_i$, with $m_i > r > 0$, and such that:

$$\forall z \in \mathbb{R}^n, \sigma [z] = \Sigma z$$  \hspace{1cm} (47)$$

where the diagonal matrix $\Sigma$ is defined by:

$$|z_i| < m_i \Rightarrow \Sigma_i = 1$$

$$|z_i| \geq m_i \Rightarrow \Sigma_i = \frac{m_i}{|z_i|}$$  \hspace{1cm} (48)$$

Properties. It may easily be verified that the definition yields the following properties [20]:

$$\forall \zeta \in \mathbb{R}^n, \sigma [\zeta] > 0$$

$$\forall z \in \mathbb{R}^n, |\sigma [z]| \leq R$$

$$|z| < r \Rightarrow \sigma [z] = z$$  \hspace{1cm} (49)$$

where $|z| = \sqrt{z^T z}$ is the norm of vector $z$ as defined in $\mathbb{R}^n$.

**Theorem 1.** If two saturations $\sigma_1$ and $\sigma_2$ are defined, such that $R_1 < \frac{1}{2} r_2$, then:

$$\forall (z_1, z_2) \in \mathbb{R}^n, |z_2| > \frac{1}{2} r_2 \Rightarrow \sigma_2^T \sigma_2 [z_2 + \sigma_1 [z_1]] > 0$$  \hspace{1cm} (50)$$

**Proof.** Since $|z_2| > \frac{1}{2} r_2$ and $|\sigma_1 [z_1]| \leq R_1 < \frac{1}{2} r_2$, one can write the orthogonal projection of the saturated vector $\sigma_1 [z_1]$ on $z_2$ as:

$$\sigma_1 [z_1] = \lambda_1 z_2 + v_1$$  \hspace{1cm} (51)$$

where $|\lambda_1| < 1$, $z_2^T v_1 = 0$, and $|\lambda_1 z_2 + v_1| < \frac{1}{2} r_2$.

Then:

$$z_2^T \sigma_2 [z_2 + \sigma_1 [z_1]] = z_2^T \sigma_2 [(1 + \lambda_1) z_2 + v_1]$$

$$= z_2^T \sigma_2 ((1 + \lambda_1) z_2 + v_1)$$

$$= (1 + \lambda_1) z_2^T \sigma_2 z_2 > 0$$  \hspace{1cm} (52)$$

We can now proceed and introduce the second derivative (46) saturated demand:

$$\ddot{y}_s = -\Lambda_2 \sigma_2 [z_2 + \sigma_1 [z_1]]$$  \hspace{1cm} (53)$$

From Theorem 1, if $|z_2| > \frac{1}{2} r_2$, then $\dot{W}_2 = -\sigma_2^T \sigma_2 [z_2 + \sigma_1 [z_1]]$ will be negative definite for saturations $\sigma_1$ such that $|\sigma_1 [z_1]| \leq R_1 < \frac{1}{2} r_2$.

Since the saturated system is asymptotically stable, after a time $T_2$ the variable $z_2$ will enter the linear zone of its saturation and remain inside of it, namely with $|z_2| < \frac{1}{2} r_2$.

After time $T_2$ the saturated demand will be equal to:

$$\ddot{y}_s = -\Lambda_2 (z_2 + \sigma_1 [z_1])$$  \hspace{1cm} (54)$$

Introducing (54) into (28) yields:

$$\dot{W}_s = (a\dot{y} + 2\ddot{y})^T (a\dot{y} + \ddot{y}) - a\ddot{y}^T \ddot{y}$$

$$= (a\dot{y} + 2\ddot{y})^T (a\dot{y} - \Lambda_2 (z_2 + \sigma_1 [z_1])) - a\ddot{y}^T \ddot{y}$$

$$= (a\dot{y} + 2\ddot{y})^T (a\dot{y} - \Lambda_2 z_2 - \Lambda_2 \sigma_1 [z_1]) - a\ddot{y}^T \ddot{y}$$

$$= (a\dot{y} + 2\ddot{y})^T (a\dot{y} - (a \dot{z}_2 + 2A) \dot{y} - \Lambda_2 \sigma_1 [z_1]) - a\ddot{y}^T \ddot{y}$$

$$= -(a\dot{y} + 2\ddot{y})^T \Lambda (2\dot{y} + \Lambda^{-1} \Lambda_2 \sigma_1 [z_1]) - a\ddot{y}^T \ddot{y}$$  \hspace{1cm} (55)$$
Using the definition of the saturation $\sigma_1 \{ z_s \} = \Sigma_2 z_1 = \Sigma_2 A^{-1} a \Lambda y$, from (55) we get:

$$W_{ts} = -(2y + ay)T \Lambda (2y + \Lambda^{-1} A \Sigma_2 A^{-1} a \Lambda y) - ayT y$$  \hspace{1cm} (56)$$

Two scenarios are now possible: (i) $z_1$ is not saturated, in which case $\Sigma_1 = I$, resulting in $W_{ts} < 0$; or (ii) $z_1$ is saturated and $\Sigma_1 = \begin{bmatrix} m_{z_1} \\ a \end{bmatrix}$, let us further analyze this case.

Taking $z_0 = 2A^{1/2} y$, $s = aA^{1/2} \Sigma_2 y$, and $Z = \Sigma_1^{-1}$, we have:

$$W_{ts} = -(z_0 + Z s)^T (z_0 + s) - ayT y$$  \hspace{1cm} (57)$$

If we consider the decomposition of the vectors in their components, $z_0 = [z_1]$, $e = [s_1]$ and also the diagonal matrix $Z = [\lambda_1]$ with elements $\lambda_1 = \frac{|z_1|}{|e|} \geq 1$, then:

$$(z_0 + s)^T (z_0 + Z s) = \sum_i (z_1 + s_1)(e_1 + \lambda_1 s_1)$$  \hspace{1cm} (58)$$

Noting that $s$ and $z_0$ have behaviors similar to, respectively, $z_1$ and $z_2$, and that $z_2$ is in its linear zone and converging, we can that after some time $|z_1| < |s_1|$, and then $z_1 \approx e/s_1$ with $|e| < 1$, so that:

$$(z_0 + s)^T (z_0 + Z s) = \sum_i (\mu_i s_1 + s_1)(\mu_i s_1 + \lambda_i s_1)$$  \hspace{1cm} (59)$$

$$= \sum_i (s_1^2 (\mu_i + 1)) (\mu_i + \lambda_i)$$  \hspace{1cm} (60)$$

which shows that the term is positive, making the result of the sum also positive. Therefore, $W_{ts}$ is also negative definite.

To include the input forces limitations into the control law design, let us consider the desired demand is a saturated one, $\dot{y} = \dot{y}_d$. From (31) and (53) we obtain:

$$J \ddot{x} + J H x = -A_2 \sigma_2 \left[ A_2 (J x + B w - J_2 x) + \sigma_1 A_2^{-1} A_1 (\eta - \eta_r) \right]$$  \hspace{1cm} (61)$$

or, solving for $\dot{x}$:

$$\dot{x} = -J^{-1} A_2 \sigma_2 \left[ A_2 (J x + B w - J_2 x) + \sigma_1 A_2^{-1} A_1 (\eta - \eta_r) \right] - H x$$  \hspace{1cm} (62)$$

Substituting now (62) into (40) leads to the control law:

$$f_s = -M (J^{-1} A_2 \sigma_2 \left[ A_2 (J x + B w - J_2 x) + \sigma_1 A_2^{-1} A_1 (\eta - \eta_r) \right] + \Gamma_1 x) - E_p S g - F$$  \hspace{1cm} (63)$$

Again, as the wind is estimated, the control law that considers the force input saturations is finally given by:

$$f_s = -M (J^{-1} A_2 \sigma_2 \left[ A_2 (J x + B \omega - J_2 x) + \sigma_1 A_2^{-1} A_1 (\eta - \eta_r) \right] + \Gamma_1 x) - E_p S g - F$$  \hspace{1cm} (64)$$

where $\sigma_1$ and $\sigma_2$ are the velocity saturation matrices obtained from (64) with $f_s$ corresponding to the input force maximum values related to the actuators limits (see section 3.3.1.1), and that satisfy the condition $R_1 < \frac{1}{2} r_2 < |z_2|$. This control law will lead to an asymptotically stable closed-loop system as long as the estimation error is bounded according to (39).

With respect to the above control law, it is important to remark that although the reference velocity $\dot{x}_r$ is a groundspeed, the feedback velocity $\dot{x}$ is an airspeed (calculated from the actual groundspeed and the wind estimation).

### 3.3. Control Implementation

The control law (64) solves the airship path-tracking problem in the presence of constant translational wind while taking into account the limitations of the demanded forces input. However, this control law cannot be directly fed into the system, and needs to be adapted, as: (i) the airship is an underactuated vehicle, as detailed in the following; (ii) the position and velocity references may be shaped to reduce the consequences of saturations; (iii) the airship actuators are not directly usable as force inputs and a conversion or control allocation is to be applied in order to compute the real actuators inputs.

The airship actuation system may be split into two sets:

- force inputs that are available from two stroke engines, on each side of the gondola, with vectoring capability ranging from -30\(^\circ\) to +120\(^\circ\). The propellers provide a complementary lift to oppose the weighting mass, as well as a forward thrust controlling the longitudinal speed; when a differential input is added between the two propellers (meaning different rotations for the left and right engines), they also provide torque to control the rolling motion near hover; finally, a stern lateral thruster may be necessary to provide yaw control at low airspeeds, although it has not been used in the DIVA airship standard configuration;

- surface deflections of the tail (in the range of -25\(^\circ\) to +25\(^\circ\)) which, in the presence of a minimum airspeed provide torque inputs mostly for the control of the pitching and yawing motions. However, when the air is perfectly still and no wind is available, the hover control is reduced to the use of the force inputs only.

Thus, the airship real actuators input corresponds to $U = [T_X, \delta_{\alpha}, T_D, T_Y, \delta_e, \delta_r]^T$, as described in section 2.

Before presenting the proposed solution to the control allocation problem, it is important to remark some features of the airship dynamics and actuation system. The airship dynamics is highly nonlinear and underactuated, with a very different behavior as the airspeed varies from the hovering or low airspeed flight (HF) to the cruise or aerodynamic flight (AF) [10, 34, 38]. The abrupt and continuous transition between the HF and AF in the dynamics implies a different use of actuators for each situation. For AF, the most important actuators are the propellers thrust and the aerodynamic elevator/rudder control surfaces, whereas for HF the effective actuators are the propellers total thrust and vectoring, differential propulsion, and the stern thruster when available.

#### 3.3.1. Control Allocation

**Force Inputs**

As it was stressed above, the relation between actuators and control inputs depends on the flight region:

- In the low airspeed region, the tail surfaces have reduced authority since the action from the surface deflections is a function of the dynamic pressure and varies as the square of the airspeed $V_t$, according to the aerodynamic characteristics of the airship [37]. This leaves the airship to be controlled by the force inputs only. The two main propellers correspond to 3 inputs ($T_X, \delta_{\alpha}, T_D$) - total thrust, vectoring angle,
and differential thrust - providing longitudinal and vertical force, pitching and rolling torques. If available, the tail lateral thruster adds one input \((T_V)\), providing a side force and a yawing torque. These force actuators are slightly influenced by the airspeed but may be considered as independent in a first step.

- In aerodynamic flight, the vectoring angle is no longer necessary, leaving the airship with a reduced vertical force. The maneuvering is mostly accomplished by the tail fins. The surface deflections correspond to the three standard inputs of aileron, elevator and rudder deflections \((\delta_a, \delta_e, \delta_r)\), which mostly correspond to torque inputs, keeping the airship with reduced lateral force input.

As stated above, although it may have up to 7 actuator inputs to control 6 forces (3 forces and 3 torques), the airship is indeed an underactuated system (particularly in hovering) due to the limitations in the controllability. In order to reduce the influence of these limitations, a solution adopted in the control design was to add the tuning parameters \((\alpha, \Lambda)\), so as to decrease the closed-loop frequency, searching for a slower solution, that would be more robust to the unmodelled and approximate dynamics, as well as input saturations.

**Conversion From Forces to Airship Inputs**

The relation from actuators to force inputs may then be established in an approximated approach neglecting the actuators dynamics, using the airspeed measurement and resolving the possible redundancies according to the usual operation of the airship [10] (the airship aerodynamic angles also have their effect, but they may be neglected in a first step, assuming small angles):

\[
U = U(f, V_t) \tag{65}
\]

where \(U = [T_X, \delta_a, T_D, \delta_e, \delta_r]^T \in \mathbb{R}^6\) is the real actuators input, the force vector is represented by \(f = [f_a, f_e, f_p, f_o, f_t]^T \in \mathbb{R}^5\), and \(V_t\) is the true airspeed. In the present case, the input \(U\) is computed as solution of the system composed by the 6 equations below, in agreement with the DIVA airship model:

\[
\begin{align*}
    f_a &= T_X \cos(\delta_a) + k_1 \delta_a \\
    f_e &= -k_2 \delta_e \\
    f_p &= T_X \sin(\delta_a) + k_3 \delta_a \\
    f_o &= -T_X \sin(\delta_e) + k_4 \delta_e \\
    f_t &= k_5 \delta_r \\
    f_p &= T_X \sin(\delta_e) + k_6 \delta_r
\end{align*} \tag{66}
\]

where \((k_2, k_3)\) are geometrical constants of the airship, and \(k_1(V_t)\) are second order polynomials expressing the airspeed depending authority of the tail deflections.

**3.3.2. Adapted Control Law to Deal with Underactuation**

As referred, the present configuration of the DIVA airship actuators results in an underactuated system. At very low airspeeds we reach the worst-case scenario, with the airship being uncontrollable due to the lack of authority from the control surfaces. The implementation of the proposed control law assumes this situation is not reached, therefore requiring that the true airspeed does not drop below a minimum, \(V_t > V_{min} = 2m/s\) (note that it is quite realistic in outdoor conditions to assume a wind intensity above this level).

Even if this limit is respected, the airship may still be underactuated as the transversal forces available are too small. This means that a straightforward correction of eventual lateral and vertical position errors might lead to a saturated inputs request. In the following, we adapt the control law (64) to deal with this scenario, obtaining a faster error correction with smoother input requests.

Consider the approximated kinematic relations:

\[
\begin{align*}
    \dot{\psi} &\approx V_t \psi \\
    \dot{\theta} &\approx -V_t \theta
\end{align*} \tag{67} \tag{68}
\]

where \(\psi\) and \(\theta\) are the pitch and yaw Euler angles that describe the airship orientation, used here in place of the quaternions for simplicity. Equations (67)-(68) allow us to relate the airship orientation with its lateral and vertical positions.

Consider now the airship is to track a rectilinear path with orientation \((\psi_r, \theta_r)\) and has lateral and vertical errors respectively \(y\) and \(z\). The angular errors are defined as:

\[
\begin{align*}
    \Delta \psi &= \psi - \psi_r \\
    \Delta \theta &= \theta - \theta_r
\end{align*} \tag{69} \tag{70}
\]

Due to the airship underactuation, if we try to independently correct the position and angular errors, depending on their magnitude, we will probably have input saturation. However, if we consider the relation between position and attitude, we may consider instead the following expressions:

\[
\begin{align*}
    \Delta \psi' &= \psi - \psi_r - k_y y \\
    \Delta \theta' &= \theta - \theta_r - k_z z
\end{align*} \tag{71} \tag{72}
\]

where the constants \(k_y\) and \(k_z\) are dependent of the airspeed \(V_t\). This means we will postpone the angular corrections and use them to annul the position errors first. The angular references (converted first to quaternions, with the roll reference \(\phi_r = 0\) used in \(\eta_r\) in the control law (64) will then be:

\[
\begin{align*}
    \psi'_r &= \psi_r + k_y y \\
    \theta'_r &= \theta_r + k_z z
\end{align*} \tag{73} \tag{74}
\]

**3.4. Simulation Results**

In order to test the nonlinear behavior of the presented control law, representative simulation tests were performed using the fully 6-DOF nonlinear model simulation environment developed in the DIVA Project. This simulator was built based on the AURORA experience and results [12]. The simulation case presented here concerns a complete airship mission to be implemented, starting with a vertical take-off, a path-tracking with two semicircles of 200m diameter, airship stabilization for ground hover, and finally a vertical landing (see Fig. 7).

The airship starts in position \((N_t, E_t, h_t) = (-30, -20, 0) m\) and is to go up 5m to the initial reference point \(p_{ref} = (N_{ref}, E_{ref}, h_{ref}) = (-30, -20, 5) m\) so as to be stable.
and ready to start the mission. From this point, the vertical take-off begins, finishing with an approach to the first point of the horizontal path-tracking at \( p_3 = (N_{r3}, E_{r3}, h_{r3}) = (0, 0, 50)m \). A square reference with a 200m side was adapted with two half-circles so as to provide a smooth reference with a continuous derivative. Although with this solution we do not always have a straight line reference, with a groundspeed reference of 7m/s and a 200m circle radius, the approximation is quite acceptable since the yaw rate is fairly small. Obviously, the angular velocity reference must be adapted to the case. When reaching the point \( p_3 = (N_{r3}, E_{r3}, h_{r3}) = (-100, 0, 50)m \), the path-tracking gives place to the airship stabilization at the coordinates \( p_3 = (N_{r3}, E_{r3}, h_{r3}) = (-30, -20, 50)m \), preparing it for vertical landing at \( p_4 = (N_{r4}, E_{r4}, h_{r4}) = (-30, -20, 1)m \).

For the take-off and landing segments, some constraints are also given along with the position reference coordinates. Regarding the groundspeed, the airship is to ascend vertically from \( p_3 \) at 0m/s and finish the approximation to \( p_1 \) at 7m/s, having a rate limit of 0.5m/s². The stabilization process will start at \( p_3 \) at 7m/s and finish at \( p_3 \) at 0m/s. The landing at \( p_4 \) is required also at 0m/s (vertical descent). Concerning the altitude, the airship has a ascent/descent rate limit of, respectively, 1.5 and -0.5m/s.

In order to test the proposed controller robustness to unmodelled wind disturbances, a 3D 2m/s turbulence was added to the constant 4m/s wind blowing from northwest at 20°.

The airship position coordinates (north \( N \), east \( E \), and altitude \( h \)) are represented in Fig. 8. The vertical take-off and landing are well perceived in Fig. 8(a), as well as the path-tracking performance. Figure 8(b) compares the real coordinates with the references provided, allowing to identify the more problematic mission point, namely during the airship descent (see Fig. 9). The remaining noticeable errors correspond to instantaneous references changes before stabilization, which the airship smoothly corrects.

Figure 9 describes the airship north-east coordinates and heading during the mission. The preferential alignment with the wind during take-off and landing is well recognized, whereas along the tail wind segment the airship appears as slightly crabbing.

The airship velocities are depicted in Fig. 10. The ground velocity components are represented in Fig. 10(a). The longitudinal groundspeed \( u \) mostly follows the reference that varies between 0m/s for take-off and landing, and 8m/s during the path-tracking. Along the circular segments, the errors are more noticeable due to the change of the wind incidence angle while the airship is turning. The lateral velocity \( v \) is also mostly influenced by the circular segments and during the tail wind segment. For the vertical velocity \( w \), the two steps of, respectively, -1.5 and 0.5m/s corresponding to the take-off and landing vertical motion are also easy to recognize.

The airspeed and aerodynamic angles can be seen in Fig. 10(b). During the whole mission, the airspeed \( V_A \) varies significantly, from values around 4m/s (above the set limit of 2m/s - see section 3.3.2.) up to 12m/s. The airship covers a wide flight envelope, from hover to the aerodynamic flight, crossing the troublesome transition region between the two. The sideslip angle \( \beta \) and the angle of attack \( \alpha \) vary between ±10° and, as expected, their
behavior is correlated with $u$ and $w$ respectively.

(a) Groundspeed: longitudinal $u$, lateral $v$ and vertical $w$ (reference - dash, real values - solid).

(b) Aerodynamic variables: airspeed $V_t$, sideslip angle $\beta$ and angle of attack $\alpha$.

![Figure 10. Airship ground and air velocities](image)

The saturated forces computed by (64) may be seen in Fig. 11, with the forces $f_x$, $f_y$ and $f_z$ represented in Fig. 11(a) and the moments $M_x$, $M_y$ and $M_z$ in Fig. 11(b).

(a) Saturated forces $f_x$, $f_y$ and $f_z$.

(b) Saturated moments $M_x$, $M_y$ and $M_z$.

![Figure 11. Saturated forces and moments request.](image)

As justified before, these forces, which in this example show no saturation, have to be converted into airship actuators inputs. These are described in Fig. 12, with the longitudinal actuators elevator $\delta_e$, total thrust $T_X$ and vectoring angle $\delta_v$ in Fig. 12(a) and the lateral ones, aileron $\delta_a$, rudder $\delta_r$ and differential thrust $T_D$, in Fig. 12(b). The noise levels in Fig. 12 are justified by the high value of turbulence considered in the simulation.

The elevator $\delta_e$ shows a higher demand at the beginning of the mission, corresponding to the ascent where the vertical rate is higher, while the rudder $\delta_r$ has a higher command during the curves and with tail wind, due to a lower airspeed. The vectoring angle $\delta_v$ is responsible for the airship lift when the airspeed $V_t$ is too low to provide the necessary aerodynamic lift: if we compare the $\delta_v$ and the $V_t$ graphics, this correlation is obvious.

![Figure 12. Airship actuators input.](image)

This mission was defined to be representative and illustrative of a realistic behavior. It clearly represents a challenge for the automatic control system, as (i) the dynamics varies from the hovering to the aerodynamic flight during the path-tracking, (ii) we have a wind input with different incidence angles (as the trajectory is circular) and also stochastic gust, and (iii) the mission includes vertical maneuvers. These simulation results show that the approach is a strong and robust tool for the design of a single global control scheme for such an underactuated airship and surely for other Unmanned Vehicles.

4. Multi-Sensing for Mapping and Surveillance

The airship is equipped with several sensing devices where vision plays an important role. The information provided by the vision sensor is complemented with other sensing devices towards a complete and robust mapping and surveillance system. In this chapter, vision systems are rigidly coupled with Inertial Measurement Units (IMUs), which complement it with sensors providing direct measures of orientation relative to the world NED frame, such as magnetometers and inclinometers (that measure gravity components).

A novel calibration technique [39] finds the rigid body rotation between the camera and IMU frames. The camera orientation in the world is then obtained rotating the IMU orientation measurement. The approximation of the rotational degrees of freedom should allow faster processing or the use of simpler movement models in computer vision tasks. For example, it can be explored to improve robustness on image segmentation and 3D structure recovery [40, 41].

Sections 4.1. and 4.2. aim on exploiting the calibrated camera-inertial system in two other domains. With direct measurement of camera orientation, rotational and translational components of camera motion can be separated. Therefore, simpler movement models can be assumed, resulting in increased performance or accuracy. Images obtained from a camera on-board the DIVA airship (see Fig. 13) are used in all experiments. Note that the blimp envelope is transparent to GPS signals, thus the GPS receiver can be safely mounted.
over the gondola.

Figure 13. The DIVA airship, with details showing the vision-inertial system and GPS receiver mounted on the gondola.

4.1. Vision-Based Trajectory Recovery and Mapping

In this section we discuss trajectory recovery from images, presenting the results in section 4.1.2. Previous results [42] have shown that the pure translation model yields more accurate height estimation that the usual homography model into a controlled laboratory environment with hand-measured ground truth. The vertical component of the airship trajectory can also be recovered this way. The other two horizontal components of the trajectory are recovered by estimating the Focus Of Expansion (FOE), and using the known vertical component to resolve scale.

Furthermore, only one solution is recovered by the process shown in this section, as opposed to the four solutions for the rotational and translational parameters recovered by the homography model. Often two of these are potentially viable, and geometric constraints are used to recover the right motion. This last step is not necessary here once the orientation and FOE are directly measured.

Experimental Platforms and Calibration

The hardware used on-board the airship is shown in Fig. 13 and described in section 2.2. The camera is a Point Gray Flea [43], which is rigidly mounted with the inertial and magnetic sensor package Xsens MT9-B [44]. During an experimental flight, images with resolution of 1024 × 768 pixels were captured at 2 fps. The camera is calibrated, its intrinsic parameter matrix K is known, and f is its focal length. Its optical center is indicated by C.

The rotation between the camera and the inertial frames is calculated by a novel calibration process [39, 45], which is available as a Matlab® toolbox for download [46]. Two examples of calibration images are shown in Fig. 14, where a chessboard is placed in the vertical position, so that its vertical lines provide an image-based measurement of the gravity direction to be registered with the gravity measurements provided by the accelerometers. The same images are also used to calibrate the camera with the Camera Calibration Toolbox [47].

Figure 14. Two examples of calibration images used to calibrate the camera-inertial system.

A Virtual Leveled Plane

The camera inertial calibration outputs the constant rotation between the camera and the IMU frames, supposing that both are rigidly mounted together. The absolute camera orientation is thus provided directly by the IMU absolute orientation measurements. Therefore, the images can be projected on entities defined on the absolute NED frame, such as a virtual horizontal plane (with normal parallel to the gravity), at a distance f (the focal length) below the camera center, named virtual leveled plane (see Fig. 15). Projection rays from 3D points to the camera center intersect this plane, projecting the 3D point into the plane. This projection corresponds to the image of a virtual camera with the same optical center as the original camera, but with optical axis coincident with the gravity vector, named here virtual downwards camera. Therefore the principal point of the virtual downward camera is the direction of gravity, named as the nadiral point N. See [42, 48] for details.

Figure 15. The virtual leveled plane concept.

4.1.1. Planar Homographies and Homologies

In computer vision, given an image pair, pixel correspondences are used to recover the relative camera poses corresponding to these two images, i.e., the rotation between both camera frames and the translation between the camera centers, albeit the translation is retrieved only up to scale. If the images are taken from a moving observer, it is possible to reconstruct the observer trajectory from a sequence of such relative poses.
This section defines and explores a pure translation model for planar scenes to achieve more accurate measurements of the camera height and to recover the camera trajectory from an image sequence. First the vertical component of the trajectory will be recovered, followed by the other two horizontal components.

The simplest case is when the 3D plane is parallel to the image plane. To simulate cameras under pure translation, and with image plane parallel to the horizontal plane, the image sequence is projected on the ground plane as exposed above using the IMU orientation measurements. The camera is allowed to move freely but the ground plane is required to be horizontal. Interesting point algorithms find pixel correspondences on the rotation-compensated images - the SURF algorithm [49] is used through this chapter.

**Definition of Homography and Planar Homology**

Consider a 3D plane imaged in two views. Consider also a set of pixel correspondences belonging to that plane in the form of pairs of pixel coordinates \((x, x')\), corresponding to the projection of the same 3D point into each view. An homography represented by a \(3 \times 3\) matrix \(H\) relates these two sets of homogeneous pixel coordinates such that \(x' = Hx\). The homography can be recovered from pixel correspondences, and it is related to the 3D plane normal \(n\), the distance from the camera center to the plane \(d\), and to the relative camera poses represented by the two camera projection matrices \(P = [I | 0]\) and \(P' = [R|t]\), by:

\[
\lambda H = \lambda \begin{pmatrix} R - tn^T/d \end{pmatrix}
\]

(75)

where \(R\) is rotation matrix and \(t\) a translation vector [50]. The scale module \(\lambda\) of the recovered homography \(H_x = \lambda H\) is the second largest singular value of \(H_x\), and the correct signal can be recovered by imposing a positive depth constraint. Then, the matrix \(H\) can be decomposed into \(R, n,\) and \(t/d\) [51]. The translation magnitude is not recovered, only the ratio \(t/d\). In the translation-only case, \(R = I\) and the homography becomes a planar homology.

A planar homology \(G\) is a planar perspective transformation that has a line of fixed points (the axis), and another fixed point outside of the axis (the vertex)\(^3\). The axis is the image of the vanishing line of the plane (the intersection of the 3D plane and the plane at infinity), and the vertex is the epipole, or the FOE. Among the properties of homologies [50, 52], we recall:

- Lines joining corresponding points intersect at the vertex;
- The cross-ratios defined by the vertex, a pair of corresponding points, and the intersection of the line joining this pair with the axis, have the same value \(\mu\) for all points;
- The homology matrix \(G\) may be defined from its axis, vertex, and the cross-ratio \(\mu\) by:

\[
G = I + (\mu - 1) \frac{v^v}{v^v a}
\]

(76)

where \(v\) is the vertex, and \(a\) the axis line.

\(^3\)A fixed point is a point \(x\) such that \(Gx = x\), i.e., a point that is not changed by the transformation.

**Homology for 3D Planes Parallel to Image Plane**

If the 3D plane is parallel to the image planes and the axis is the infinite line \(a = (0, 0, 1)^T\), then (76) becomes:

\[
G = \begin{bmatrix}
1 & 0 & (\mu - 1) \cdot v_z \\
0 & 1 & (\mu - 1) \cdot v_y \\
0 & 0 & \mu
\end{bmatrix}
\]

(77)

where \(v_z, v_y\) are the inhomogeneous image coordinates of the vertex \(v = (v_z, v_y, 1)\). The cross-ratio \(\mu\) depends only of the relative depths of the 3D plane on the two views. To analyze this relation, we recall from Arnaspg’s paper [53] that the relative scene depth of two points equals the reciprocal ratio of the image plane distances to the vanishing point of their connecting line.

Taking two images of the same 3D point \(X\) under pure translation, and defining \(Z\) and \(Z'\) as the depth of \(X\) in the first and second views, and \(x\) and \(x'\) as its respective image coordinates, as in Fig. 16, we have:

\[
\frac{Z'}{Z} = \frac{\text{dist}(x, v)}{\text{dist}(x', v)}
\]

(78)

where \text{dist} means Euclidean distance on the image. Therefore, the relative depth of the same point in two views is calculated from image measures.

The relation between scene depths and image distances is valid for every single point, and it only requires an image of the same point in two views, and the FOE. However, if a 3D plane is parallel to the image planes, all points in the plane have the same depth, and are transferred between the two views by the same homology.

Figure 16. A pair of cameras under pure translation imaging the same 3D point.

Therefore, the homology calculation involves many corresponding pixel pairs, and thus is potentially more stable than an image measure involving just one pair. Going back to the relation between the homology parameter \(\mu\) and the depth of the 3D plane, applying
equation (77) allows us to find:

\[ x' = Gx = \left[ \frac{x_x + v_x - x_x}{\mu}, \frac{x_y + v_y - x_y}{\mu}, 1 \right] \]  

(79)

where \( x = (x_x, x_y, 1)^T \). Now, by calculating \( |x - x'| = |x - Gx| \), we relate this difference with \( |x - v| \), in image coordinates:

\[
|x - x'| = \left[ (x_x - \frac{x_x}{\mu} + \frac{v_x}{\mu})^2 + (x_y - \frac{x_y}{\mu} + \frac{v_y}{\mu})^2 \right]^{1/2} \]

\[
= \left[ \left( (x_x - v_x)(1 - \frac{1}{\mu}) \right)^2 + \left( (x_y - v_y)(1 - \frac{1}{\mu}) \right)^2 \right]^{1/2} \]

\[
= \left[ ((x_x - v_x))^2 + ((x_y - v_y))^2 \right] \left( 1 - \frac{1}{\mu} \right)^{1/2} \]

and then:

\[
|x - x'| = |x - v| \left( 1 - \frac{1}{\mu} \right)^{1/2} \]

(81)

As \( x, x', v \) are collinear, \( |x - x'| + |v - x'| = |x - v| \). So, from (81) we find the image distances of equation (78) and we can update that equation as:

\[
\frac{|Z'|}{Z} = \frac{dist(x, v)}{dist(x', v)} = \mu 
\]

(82)

Therefore, the relative depth of the plane is equal to \( \mu \), a parameter of the homography matrix. This relation agrees with the known fact that given the homography matrix induced by a 3D plane in two views, the relative distance between the camera centers and the plane is equal to the determinant of the homography [54, 55].

This is valid for homographies (correctly scaled), thus also for planar homographies. From equation (77), note that \( \det(G) = \mu \), and as the distance between the camera center and the plane is the depth of the plane, (82) is again verified.

**Calculating Relative Depth for Horizontal Planes**

This section describes the process to calculate the depth ratio, in two views related by a pure translation, of a 3D plane parallel to the two image planes. This process exploits (82) in a practical implementation.

First, the images are projected on the virtual leveled plane, and pixel correspondences are established. Then an initial FOE estimate \( v_0 \) is obtained from the pixel correspondences using a robust linear estimation with RANSAC (Random Sample Consensus) [56] followed by an optimization step [57]. The FOE estimation also excludes outliers from the set of corresponding pixel pairs, as not to hamper the final optimization.

From the pixel correspondences and the FOE estimate, an initial estimate \( \mu_0 \) of the cross-ratio parameter \( \mu \) is obtained by measuring and averaging for all pairs of corresponding pixels the ratios of image distances to the FOE as in (82).

Given the initial estimates \( v_0 \) and \( \mu_0 \), an optimization routine minimizes the projection error of the pixel correspondences when projected by the homology \( G(v, \mu, a = [0, 0, 1]^T) \), finding improved estimates for \( v \) and \( \mu \). The error metric is Sampson distance, also used to estimate full homographies [50]. The optimization is performed by the Levenberg-Marquardt algorithm, using the same implementation used to estimate full homographies [58], but parameterized by the homology parameters \( v \) and \( G \). As it is advisable to over-parameterize the optimization [50], \( v \) is considered as a 3-element vector, which is normalized once its final value is known. The relative depth is the determinant of \( G \), i.e., \( \mu \).

Figure 17 summarizes this process. Notice that there is no need to project all the image on the virtual plane, but only the coordinates of the pixel correspondences. Sensor data could provide directly an initial FOE estimate. The initial \( \mu \) estimate is trivial, and the final optimization roughly takes as much time as the optimization necessary to calculate an homography. Therefore, potentially this process can be fast enough for robotic applications.

**Reconstructing the Complete Relative Pose**

The relative depth between the two virtual cameras has been recovered in section 4.1.1. As the rotation is compensated, the virtual cameras may be represented by \( P = [I|0] \) and \( P' = [I|t] \). The relative height corresponds to the \( z \) component of the vector \( t \), although its scale depends on the height of the first camera.

The FOE, that is already calculated by the above process, is the direction of the other two components of \( t \), although it does not indicate the scale. The correct scale of translation must come from another independent measurement. Nevertheless, the scale of the \( x \) and \( y \) components given by the FOE may be calculated in function of the altitude component.

Given the FOE \( v = (v_x, v_y, 1)^T \), the nadiral point \( N = (N_x, N_y, 1)^T \), and the camera intrinsic parameters focal length \( f \) and pixel size \( dp_x \) and \( dp_y \) in the \( x \) and \( y \) directions, the vector \( t \) is calculated as a function of its vertical component \( t_x \), as:

\[
t = \begin{bmatrix} (v_x - N_x)dp_x + t_x \\ (v_y - N_y)dp_y + t_x \\ t_x \end{bmatrix} \]

(83)

This relation is derived from the similar triangles shown in Fig. 18 for the \( x \) component, and a similar relation exists for the \( y \) component. The figure omits the change of coordinates...
(N_x) and units (dpz) that must be applied to v_x. Therefore, the trajectory of a mobile observer may be reconstructed up to scale, as in the usual homography model, by adding the relative pose vectors over an image sequence.

\[ t_x = \frac{f}{v_x} \]

Figure 18. Finding the scale of translation from the difference on height.

4.1.2. Trajectory Recovery Results

Visual Odometry: Heights for an UAV

This experiment uses images taken by the remotely controlled airship of Fig. 13 carrying the IMU-camera system and a Global Positioning System (GPS) receiver, flying over a planar area. The GPS measured height is shown in Fig. 19 compared against visual odometry based on the \( \mu \) value of homologies calculated for the sequence of images using the process described in section 4.1.1.

\[ \text{UAV Trajectory from Relative Heights and FOE} \]

As exposed on section 4.1.1., for an image pair projected on the leveled plane, the camera projection matrices may be written as \( P = [I/0] \) and \( P' = [J/1] \), as the rotation has been compensated. Therefore, by recovering \( t \) for each image pair on the sequence, the UAV trajectory can be reconstructed by adding the sequence of translation vectors.

Given an image pair \( I_i, I_{i+1} \), the vertical component of the vector \( t \) is calculated from the heights computed above as \( t_y = h_{i+1} - h_i \). The other two components are given by the direction of the FOE. As the FOE is only a direction without scale, the scale of \( t \) must be given by the known altitude component (83).

During flight, the airship height changes very little between consecutive frames. Thus, the lines connecting corresponding pixels are almost parallel. Therefore, when calculating the FOE from the correspondences, the direction recovered may be the direction opposite to the real FOE. To solve this ambiguity, the measured FOE is compared with the vehicle heading indicated by the IMU. If the FOE points are behind the vehicle, the measurement is inverted, i.e., \( v = -v_{\text{measured}} \). The airship is never moving backwards in our data sets.

The trajectory is thus reconstructed for the same UAV data set used above. The height data are the same as in Fig. 19, but the other two dimensions are interpolated for the images where the homology calculation failed (indicated with red squares in Fig. 20).

The blue squares show the trajectory reconstructed by adding up all translation vectors, with no filtering applied. The pink diamonds show the smoother trajectory reconstructed after applying a Kalman filter on the translation measurements. Both 2D and 3D plots of the same data are provided.

4.2. Digital Elevation Map from Image Correspondences

In this section we further explore the relation between scene depths and image ratios with the FOE for pixel correspondences [53]. For rotation compensated images projected on the horizontal ground plane, scene depth indicates height, and is used to build a coarse Digital Elevation Map (DEM) grid, performing 3D mapping from monocular aerial images.

4.2.1. Calculating Height For Each Pixel Correspondence

Recall that (78) is valid for each individual corresponding pixel pair. Therefore, if an image contains regions above the ground plane, the relative height can be directly recovered for
Figure 20. 2D and 3D plots comparing visual odometry based on homology compared with GPS trajectory measurements.

Each corresponding pixel. This suffices to order these pixels by their height, but it is not an absolute measurement. Again, additional information is needed to recover scale from imagery. The absolute height of these points may be recovered if the absolute height on both views is known—case (a), or, equivalently, if the height of one view and relative height corresponding to the ground plane is known—case (b).

In case (a), defining \( \mu_i = \frac{\partial Z_i}{\partial x_i} = \frac{h - h_{i}}{h_{i} - h_{p_i}} \), as the relative height for the corresponding pixel pair \((x_i, x'_i)\), the height of the 3D point \(X_i\) imaged by \(x_i\) as \(h_{p_i}\), and \(h\) and \(h'\) as the known camera heights, as shown in Fig. 21, and then solving for \(h_{p_i}\), we have:

\[
h_{p_i} = \frac{\mu_i h - h'}{\mu_i - 1}
\]  

(84)

Figure 21. Calculating the height of a 3D point from an image pair under pure translation.

In case (b), we substitute \(h' = \mu h\), where \(\mu\) is the relative camera height over the ground plane, into (84) to obtain:

\[
h_{p_i} = \frac{(\mu_i - \mu)h}{\mu_i - 1}
\]  

(85)

Supposing that the ground plane is visible in the majority of the image, the term \((\mu_i - \mu)\) will be used to compensate for errors in the IMU orientation measurements. An image wrongly projected on the ground plane due to measurement error results on a deviation on the image ratios calculated for the pixel correspondences, related to the difference between the real ground plane orientation and the one given by the orientation measurement. Therefore, given all image ratios for all pixel correspondences, a 3D plane is fitted over the \((x_i, y_i, \mu_i)\) triplets, where \(x_i = (x_i, y_i, 1)\) in inhomogeneous image coordinates, using RANSAC to look for the dominant plane on the scene, that should be the ground plane.

This fitted 3D plane is used to compensate for a linear deviation induced by orientation errors, by calculating, for each \(x_i\), the value \((\mu_i - \mu_0(x_i, y_i))\), where \(\mu_0(x_i, y_i)\) is the \(\mu\) value corresponding to the \((x_i, y_i)\) coordinates on the fitted plane, and taking this value as \((\mu_i - \mu)\) on (85).

Figure 22 shows the image ratios calculated before and after correction with a fitting 3D plane. These points are very fast to obtain, as no homography or homology is recovered, and just image ratios are needed for individual points. The plane fitting is a simple model for RANSAC, and potentially it can be skipped if better measurements are available.
point $X_i = (X_i, Y_i, h_{p_i})$ may be calculated by similarity of triangles (see Fig. 21) as:

$$r_i = \frac{\text{dist}(x_i, N)}{f} (h - h_{p_i})$$  \hspace{1cm} (86)$$

where $x_i$ is the image projection of $X_i$ and $N$ the principal point of the virtual downward camera. The angle $\theta_i$ between the line $x_iN$ and the east axis is directly calculated from their coordinates. Then, transforming $(r_i, \theta_i)$ from polar to rectangular coordinates yields $(X_i, Y_i)$, the remaining two components of $X_i = (X_i, Y_i, h_{p_i})$, in the local NED frame. If the camera pose in the world frame is known, these coordinates may be registered onto the world frame, incorporating the height measurements in a global DEM.

Each point $X_i$ represents a height measurement for an individual DEM cell. The height of each cell and the heights measurements are represented by random variables with gaussian distribution, and the cell update follows the Kalman Filter update rule. There is not an initialization and the cell takes its first value when incorporating its first measurement.

The position of each point $X_i$ has also an uncertainty on the $xy$ axes, i.e., it may be uncertain which cell the measurement belongs to. Currently the influence of measurements on neighboring cells is approximated by considering all measurements exact on the North-East axes, and then smoothing each local DEM with a gaussian kernel with variance similar to the measurements.

Figure 23 shows a DEM constructed over a 10 frame sequence (a 5s portion of the flight), with the GPS-measured vehicle localization shown as red stars. The cell size is $3m$, and the gaussian convolution kernel has standard deviation of $2m$. The highest cells correspond to the building, and the smaller blue peaks correspond to the airplanes and vehicles. Points more than $1m$ below the ground plane are discarded as obvious outliers. The mosaicing on the left - included just to allow the reader to compare visually the covered area and the DEM grid - is built from homographies calculated between each successive image pair (see [42, 48] for details).

### 4.2.3. Applications and Future Improvements

In section 4.1.2, the complete UAV trajectory was reconstructed from rotation compensated imagery. In this section, a 3D map of the environment, in the form of a sparse DEM, is built from punctual height measurements generated from a sequence of images and the vehicle pose. These two techniques assume mutually exclusive conditions: the trajectory recovery requires imaging a planar area, and the mapping procedure is meaningful only if there are obstacles above the ground plane. Therefore, they could not make a complete SLAM (Simultaneous Localization and Mapping) scheme on their own, but they may be complementary methods to other SLAM approaches. A SLAM scheme also needs to obtain, from the maps built at each step, another displacement measurement to constrain the vehicle pose estimation, by matching local maps with the previous local map or with the global map built so far. A comparison of different approaches for matching 2D occupancy grids built from sonar data indicates that both options are feasible [59], and the DEM grids could be correlated this way.

Figure 22 shows that all pixel correspondences are concentrated into one small portion of the image. Outside this region, there are other correspondences found by the interesting
airships for environmental monitoring and aerial inspection missions. The research is at the moment divided in two pathways - guidance and control, and vision perception - for which significant results are presented.

we have compiled a list of requirements for a system to be able to collect the data sets used on this chapter and also to perform automatic flight tests, even if we have not satisfied all these requirements ourselves, neither performed automatic flight. This list has evolved through various years and two different projects and should provide an overall road map to similar endeavors. A short overview of the system developed so far is also provided.

A novel approach for the airship automatic control problem is described. The main outcome in this area is the resulting asymptotically stable backstepping control law which takes into account input saturations and bounded wind estimation errors. Important implementation issues are considered in the design process, namely control allocation and reference shaping to deal with airship underactuation. The simulation results obtained for a representative mission covering all the usual tasks like take-off and landing, stabilization and path-tracking, illustrate the controller performance even in the presence of realistic wind disturbances. The presented single global control law proves to be a strong and robust solution for the automatic guidance and control of an airship, valid over a wide flight envelope ranging from hover to aerodynamic flight.

The complete UAV trajectory was reconstructed from rotation compensated imagery. The reconstructed trajectory has visible errors and drift in the long term, but it may be useful in the case of temporary GPS dropout. The process is relatively fast, and can be made more accurate by having more accurate orientation measurements or incorporating other sensors to measure the speed of the vehicle, i.e., the direction of the FOE. A 3D map of the environment, in the form of a sparse DEM, is built from punctual height measurements generated from a sequence of images and the vehicle pose. After interesting point matching is performed, the mapping process is very fast, measuring the height for each corresponding pixel pair with just a simple calculation. These two techniques may be useful into a SLAM context, although they can not build a complete SLAM scheme on their own.

Autonomous flight and visual perception are two key issues to be addressed in order to define an aerial surveillance system. The approaches currently under development in project DIVA have been presented in this chapter and are gradually being validated and integrated in the airship prototype.

5. Conclusion

This chapter introduces the Portuguese Project DIVA, which focuses on the establishment of the technologies required to substantiate autonomous operation of unmanned robotic

Figure 23. 2D and 3D plots of the DEM generated from the 10 frames mosaiced on the left. The red stars indicate the vehicle trajectory.

point matching algorithm, but they are considered outliers during the FOE estimation and rejected, probably due to errors on the orientation measurement. For the estimation of the homology, and specially of its cross ratio used on the height component of the trajectory estimation, usually there are enough correspondences for a reliable estimation. But for the mapping problem, the area covered by height measurements is quite smaller than the total area imaged (see Fig. 23), and thus the map becomes too sparse for a reliable correlation between the DEM grid maps. This problem could be addressed by improving the accuracy of the orientation measurements, by increasing the image frame rate, or by developing models that take into account small uncompensated rotations. A Technical Report is available with more details on the theory and usage of the homology model, and on the mosaicing procedure [48].

References


AN ARCHITECTURE FOR ADAPTIVE SWARMS

Suranga Hettiarachchi, Paul Maxim and William M. Spears
University of Wyoming, Laramie, WY 82071, USA

Abstract

The focus of our research is to design and build rapidly deployable, adaptive, cost-effective, and autonomous distributed robot swarms. Our objective is to provide a scientific, yet practical, approach to the design and analysis of swarm behaviors. This chapter provides an overview of our work in this area. First, we summarize the basis for our robot control algorithms, which we call artificial physics or physicomimetics. Unlike biomimetic approaches, we focus on robotic behaviors that are similar to those shown by solids, liquids, and gases. Solid formations are useful for distributed sensing tasks, while liquids are for obstacle avoidance tasks. Gases are practical for coverage tasks, such as surveillance and sweeping. Physicomimetics is scalable, robust, and fault-tolerant.

Despite the fact that physicomimetics is amenable to theoretical analyses that guide its use, the fact remains that a real-world environment will often have unanticipated qualities that hurt the performance of the robot swarm. Hence, we also describe our novel technique for adaptive swarms. Unlike prior off-line approaches that attempt to re-train the behavior of the swarms in a simulation environment, our on-line approach adapts the behavior of the swarm in real time, while the swarm is performing the task.

In order to function properly the robots in the swarm must be able to accurately localize their local neighbors and to share information. Hence we also outline our enabling hardware technology for swarms of robots. Our plug-in hardware module provides the capability to accurately localize neighboring robots, without using global information and/or the use of vision systems. It also couples localization with data exchange, allowing physicomimetics and adaptation to be fully integrated onto physical robots.

1. Introduction

The focus of our research is to design and build rapidly deployable, adaptive, cost-effective, and autonomous distributed robot swarms. Our objective is to provide a scientific, yet practical, approach to the design and analysis of swarm behaviors.
Robotics Research Trends

Xing P. Guô
Editor

Contributor
José Azinheira
Gabriele Bruzzone
Martin Buss
Massimo Caccia
Shen Cheng
Hernán P. Dacharry
B. S. Dhillon
Jorge Dias
Anthony Engwirda
Norbert Giambiasi
Suranga Hettiarachchi
Kenichi Kanatani
Gregor Klancar
Yonghuai Liu
Takahumi Matsumaru
Paul Maxim
Luiz Mirisola
Yoshikazu Mori
Alexandra Moutinho
Chang-Woo Park
Angelika Peer
Xuejun Ren
Thomas Schauf
Igor Skrjanc
William M. Spears
Ulrich Unterhinninghofen
Hong Zhou
CONTENTS

Preface vii

Expert Commentary 1
Robotics Research Trends 3
Anthony Engwirda

Short Communication 7
Intelligent Modeling and Adaptive Control of Flexible Robot Manipulators 9
Chang-Woo Park

Research and Review Studies 31

Chapter 1 Latest Progress of 3-D Reconstruction from Multiple Camera Images 33
Kenichi Kanatani

Chapter 2 Project Diva: Guidance and Vision Surveillance Techniques for an Autonomous Airship 77
Alexandra Moutinho, Luiz Mirisola, José Azinheira and Jorge Dias

Chapter 3 An Architecture for Adaptive Swarms 121
Suranga Hettiarachchi, Paul Maxim and William M. Spears

Chapter 4 Preliminary-Announcement Function of Mobile Robots’ Upcoming Operation 155
Takafumi Matsumaru

Chapter 5 DEVS and Timed Automata for the Design of Control Systems 193
Norbert Giambi and Hernán P. Dacharry

Chapter 6 Noisy Surface Smoothing Using Tsallis Entropy 223
Hong Zhou, Yonghui Liu and Xuejun Ren