Mobile Robotic Surveillance Systems: Detecting and Evaluating Changes in 3D Mapped Environments

I. Amorim, R. Rocha and J. Dias
Institute of Systems and Robotics
Department of Electrical and Computer Engineering
University of Coimbra, Polo II
3030 Coimbra (Portugal)
{ivonefamorim* | rprocha | jorge}@isr.uc.pt

***Abstract*—This paper presents a study of three possible techniques to detect changes in the environment based on the use of a combination of the *Earth Mover's Distance* (EMD) and each one of the following methods: Principal Components Analysis (PCA), Gaussian Mixture Models (GMM) and Plane Extraction (PE).**

It starts by describing how the EMD can be used with each one of the three methods to detect changes among 3D point clouds that one obtains as robot measurements made with a laser. The advantages of using a laser instead of cameras are related with the sensibility of cameras to weather conditions, while the capability of change detection is crucial for truly autonomous mobile robots, as well as for mobile robots integrated in surveillance systems, for example.

The techniques here presented where either implemented in MatLab or adapted from already existing implementations, and a set of simulations were made to evaluate their behaviour relatively to the following aspects: their sensibility to errors in the data, their ability to detect objects of different sizes and their computational complexity.

The results obtained show that the combination PCA-EMD is the one with the lowest computational complexity. However, the difference between the PCA-EMD and GMM-EMD techniques is not much significant. More importantly, the results clearly show that the GMM–EMD technique is by far the more stable, in the sense that presents a less sensibility to errors and is able to detect changes with greater reliability. In fact, the techniques PCA-EMD and PE-EMD yield too many fluctuations in the results, making it impossible to distinguish if the technique is detecting a change or the result obtained is because of errors. The GMM-EMD technique has proved to be the best one allowing the detection of changes as was intended in this work.

Index Terms - Change detection, Earth Mover's Distance, Principal Components Analysis, Gaussian Mixture Models, Plane Extraction.

I. Introduction

To have a truly autonomous mobile robot it is necessary to have a method that allows the robot to solve the three following problems:

- to use what it perceives from the environment in order to construct a representation of that environment;
- to localize itself in that representation;
- to decide its next movement (based on what it senses from the actual state of the environment).

There are a lot of methods/techniques that try to answer, simultaneously, the first two problems presented above. They are known as SLAM (Simultaneous Localization And Mapping). SLAM is a process by which the robot estimates a map of the environment and localizes itself relatively to that map, given its perceptions and movements.

In the last years there has been quite some progress towards solving the SLAM problem, as well as some computational improvements. [1], [2] The latest advances were brought forward by hierarquical and hibrid approaches, namely, by the integration of global topological maps with local metric maps [3]. This has the advantage of reducing the computational complexity, and makes it possible to deal with larger environments. But, there are still some unsolved problems related with SLAM: trade off between environment size and computational efficiency, dual problems of perceptual ambiguity and data association, and lack of robust loop closure for large environments.

To enable the robot to decide its own movements, i.e. to solve the third problem presented above, it is necessary to provide the robot with some kind of alarms that are actived when there are important changes in the environment, changes that may affect the path of the robot. Therefore, when a robot is moving through an environment for a second time, it needs to compare the data that is extracting with the data extracted the last time it was at that same place, in order to determine if there are any changes in the environment that will alter its predefined path. In order to do that, the robot needs to evaluate if there are any relevant changes in the environment, and if there are, it needs to detect where those changes are located.

Change detection can also be used, for example, in surveillance systems to detect if there are any changes, especially in critical places in the environment. In this setting, it is advantageous to use laser range finder instead of cameras, since the latter are negatively affected by weather conditions.

The present work evaluates and compares three different methods that we have used for change detection: principal components analysis, geometric primitives, namely planes, and gaussian mixture models. These methods are all compared using the so called "earth mover's distance" to evaluate their capabilities to detect change.

In section 2 we formulate the problem dealt with this paper, and we describe the *Earth Mover's Distance* in detail. In section 3 we describe the 3 methods used to attack that problem, namely: principal components analysis, Gaussian mixture models, and plane extraction. For each one we explain how the *Earth Mover's Distance* can be applied to measure differences among sets of points. In section 4 we describe simulations we made, and report the results obtained. Finally, in the last section, we draw some conclusions from what was done, and point out some possibilities for future work.

II. Analysis and Characterization of Changes in 3D maps

We start by formulating in an abstract way the problem of finding appropriate measures to detect and locate ???segment??? changes in sets of points. We then describe the *Earth Mover's Distance*, which will be the bedrock on which those measures will be built upon.

A. Formulation of the problem

Our problem can be formulated as follows:

Let $\mathcal{P}$ be the set of all finite subsets of $\mathbb{R}^3$, that is, the set of all finite sets of 3-dimensional points. For each $S \in \mathcal{P}$, we denote by $|S|$ the number of its elements.

Now, given $S, T \in \mathcal{P}$, that could be obtained by a laser mounted on a mobile robot, for example, we want to evaluate how different these sets are, and, if possible, segment the subset of points associated to those differences. In order to do that, we define, for each method that we are analyzing in the paper, a function:

$$d : \mathcal{P} \times \mathcal{P} \to \mathbb{R}_0^+$$

that assigns to each pair $S, T \in \mathcal{P}$ a number, $d(S,T)$, that quantifies the extent of the changes.

Moreover, given $S, T \in \mathcal{P}$, say S being the set of points observed by the robot at a certain location, and T being the set of points observed at that some location at a later time, if either $|S| \neq |T|$ or $d(S,T)$ is about a certain pre-determined threshold, δ, then we give procedures to determine the subsets of S where changes ocurred, which can either be single points or clusters of points, and what are the corresponding (new) subsets in T. Those procedures also should give points or subsets of T that have no correspondence in S (new features).

We will describe in the next section three different methods, based on Principal Components Analysis, Gaussian Mixture Models, Plane Extraction, respectively, to define such a distance d, using the Earth Mover's Distance, to be described shortly. One of the goals of this paper is to we compare all these three methods by applying them to the same 25(?) situations, and analyzing how each behaves in regard to each one of the following aspects:

- discriminating power relative to the percentage of change;
- scale used;
- computational cost.

B. Earth Mover's Distance

The earth mover's distance (EMD) is defined in [4] as a sort of distance[1] between two distributions of points in space for which a "ground distance" is given, that is, a distance between points. This "distance" between sets of points is based on the solution of a particular kind of transportation problem in linear optimization. The basic ideia is that, given two distributions, one can imagine one of them as a mass of earth spread in space, while the other can be thought of as a collection of holes in that same space. It can always be assumed that there is at least as much earth as needed to fill all the holes by switching the roles of earth and holes, if necessary. Then, the EMD measures the least amount of work needed to fill the holes with earth.

To rigorously define the EMD distance in $\mathbb{R}^n$, let us start by recalling the notion of an n-dimensional *weighted point*: it is a pair (x, w) with $x \in \mathbb{R}^n$ and $w \in \mathbb{R}_0^+$, the weight of the point x. Now, if

$$A = \{(x_1, w_1), (x_2, w_2), ..., (x_m, w_m)\}$$

and

$$B = \{(y_1, u_1), (y_2, u_2), ..., (y_n, u_n)\}$$

are two sets of n-dimensional weighted points, with $m \leq n$, and if

$$W = \sum\nolimits_{i=1}^{m} w_i \quad \text{and} \quad U = \sum\nolimits_{j=1}^{n} u_j$$

are the total weight of A and B, respectively, the EMD between A and B is defined as:

$$\mathrm{EMD}(A,B) = \min_{F \in \mathcal{F}(A,B)} \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{\min\{W, U\}}, \qquad (1)$$

where d_{ij} is the distance from x_i to y_j, and $F = \{f_{ij}\} \in \mathcal{F}(A,B)$, with $\mathcal{F}(A,B)$ being the set of all feasible flows between A and B defined by the constraints:

1) $f_{ij} \geq 0, \quad i = 1, ..., m, j = 1, ..., n;$
2) $\sum\limits_{j=1}^{n} f_{ij} \leq w_i, \quad i = 1, ..., m;$
3) $\sum\limits_{i=1}^{m} f_{ij} \leq u_j, \quad j = 1, ..., n;$
4) $\sum\limits_{i=1}^{m} \sum\limits_{j=1}^{n} f_{ij} = \min\{W, U\}.$

The motivation for this set of conditions is the following: think of, as mentioned above, the weight associated to each point in one of the sets of data as a portion of "earth" that needs to be moved to the other set of data, thought of as "holes", while the weight of these latter points are thought of as their capacity. The set of points taken as containing earth is the one that has greater total weight. In that context, the quantities f_{ij} represent the amount of earth that gets moved between the i-th point in one set and the j-th point in the other. Then, condition (1) simply states that there are no negative amounts of earth; condition (2) and (3) mean that from each point one cannot get more earth than the amount it contains,

[1] It is a distance only on some special cases: see the remark made at the end of this subsection.

and one cannot put more earth in a hole than what it capacity allows; finally, condition (4) means that, in the end, all holes must be filled.

Remark: The EMD is a true distance only when the total weight of the two sets of weighted points is the same (see appendix A of [4]).

III. Techniques for Detecting and Evaluating Changes

We now describe the three techniques that we use to detect and measure change in two different sets of data: principal components analysis, Gaussian mixture models and plane extraction. For each one of them, we explain how the EMD metric can be used to yield a way to quantify the differences between sets of points.

A. Principal Components Analysis

Principal Components Analysis (PCA) is a well known statistical technique to identify patterns in data [5]. It is especially useful when we are analysing data involving several variables or dimensions. Once one has found those patterns, it is possible to compress the data by reducing the number of variables without much loss of information, and this is the standard application of PCA. Here we will make a slightly different use of this technique.

The PCA method consists of, given a set of points[2]:

$$\mathbf{P}_k = (x_k, y_k, z_k) \in \mathbb{R}^3 \qquad (k = 1, \ldots, n),$$

to compute a referencial that strongly captures the way these points are distributed. It has the property that the new coordinates of the given points are ordered by decreasing variance. That is, in that new referencial, the first coordinate (called the first principal component) has greatest variance, the second greatest variance lies on the second coordinate, and so on.

The so called *components* in the PCA method are in fact the eigenvectors of a certain covariance matrix, obtained as follows: one starts by computing the mean $\mu(\mathbf{P})$ of the data $\mathbf{P} = (\mathbf{P}_1, \ldots, \mathbf{P}_n)$:

$$\mu(\mathbf{P}) = \frac{1}{n} \sum_{k=1}^{n} \mathbf{P}_k.$$

Then one subtracts that mean to each one of the values involved (this corresponds to moving the center of the referencial to the centroid of the data):

$$\tilde{\mathbf{P}} = \mathbf{P} - \mu(\mathbf{P}) = ((\tilde{x}_1, \tilde{y}_1, \tilde{z}_1), \ldots, (\tilde{x}_n, \tilde{y}_n, \tilde{z}_n)).$$

One now computes the corresponding square covariance matrix that correlates each pair of variables (three in this context) in $\tilde{\mathbf{P}}$, obtaining the 3×3 matrix:

$$\mathbf{M} = \begin{pmatrix} cov(\tilde{x}, \tilde{x}) & cov(\tilde{x}, \tilde{y}) & cov(\tilde{x}, \tilde{z}) \\ cov(\tilde{y}, \tilde{x}) & cov(\tilde{y}, \tilde{y}) & cov(\tilde{y}, \tilde{z}) \\ cov(\tilde{z}, \tilde{x}) & cov(\tilde{z}, \tilde{y}) & cov(\tilde{z}, \tilde{z}) \end{pmatrix},$$

where $\tilde{x} = (\tilde{x}_1, \ldots, \tilde{x}_n)$, $\tilde{y} = (\tilde{y}_1, \ldots, \tilde{y}_n)$, $\tilde{z} = (\tilde{z}_1, \ldots, \tilde{z}_n)$, and the covariance $cov(\tilde{x}, \tilde{y})$, for example, is obtained by the formula:

$$cov(\tilde{x}, \tilde{y}) = \frac{1}{n-1} \sum_{k=1}^{n} (\tilde{x}_k - \mu(\tilde{x}))(\tilde{y}_k - \mu(\tilde{y})).$$

Finally one computes the eigenvalues, $\lambda_1, \lambda_2, \lambda_3$, and corresponding unitary eigenvectors, $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$, of the matrix $\mathbf{M}$. These eigenvectors are the mentioned *components*.

In the general case, the number of components is equal to the number of variables being analysed. The component that corresponds to the highest eigenvalue is the one that describes the strongest relation in the data. The eigenvectors obtained depend on how the data is spread in space. In particular, if one has two sets of data, and one wants to analyse how different they are, one can compare their components to get an ideia of those differences: the differences between their eigenvalues increases with differences in the data sets.

As mentioned, PCA is mainly used to reduce the number of dimensions or variables, compressing the information by replacing the data points by their projections on the principal component axes or, more generally, on a d-dimensional plane defined by some subset of the d eigenvectors, obtained by removing the ones with smaller eigenvalues. One can also use PCA to represent the data on a referencial that brings forward the different levels of correlation among the data.

In mobile robotics, one possible use of PCA is to detect the existence of changes in an environment that a robot is revisiting, by comparing the components of the set of measurements taken by the robot on the first visit with the components of the data obtained the second time over. To be more precise, the EMD distance yields a way to compare the sets of components obtained from two different sets of points by using as weighted points the eigenvectors, their weights being the corresponding eigenvalues. Using the notations introduced in sections 2.1 and 2.2, if $S = (\mathbf{P}_i)_{i=1,\ldots,m}$ and $T = (\mathbf{Q}_j)_{j=1,\ldots,n}$ are two sets of points, and if $A = \{(\mathbf{v}_1, \lambda_1), (\mathbf{v}_2, \lambda_2), (\mathbf{v}_3, \lambda_3)\}$ and $B = \{(\mathbf{w}_1, \mu_1), (\mathbf{w}_2, \mu_2), (\mathbf{w}_3, \mu_3)\}$ are the corresponding sets of weighted points, consisting of the eigenvectors obtained by the PCA method, each one weighted by its associated eigenvalue, then:

$$d_{PCA}\left((\mathbf{P}_i)_i, (\mathbf{Q}_j)_j\right) = \text{EMD}\left(\{(\mathbf{v}_k, \lambda_k)\}_k, \{(\mathbf{w}_\ell, \mu_\ell)\}_\ell\right) \tag{2}$$

B. Gaussian Mixture Models

A *mixture of Gaussian functions* is a probability density function given by a convex linear combination of Gaussian density functions [6]. More precisely, a function is a mixture of Gaussian functions if it has the form:

$$f(\mathbf{x}, \Theta) = \sum_{k=1}^{K} p_k \, g(\mathbf{x}; \mu_k, \Sigma_k) \qquad \left(\mathbf{x} \in \mathbb{R}^N\right),$$

where the functions g are Gaussian densities given (as usual) by

[2] For simplicity, we describe PCA for 3 dimensions only, which is the case we are interested in, but this description carries over in an obvious way to any number of dimensions.

$$g(\mathbf{x};\mu_k,\Sigma_k) = \frac{1}{\left(\sqrt{2\pi}\,|\Sigma_k|\right)^N}\, e^{-\frac{1}{2}\left((\mathbf{x}-\mu_k)^T\,\Sigma_k^{-1}\,(\mathbf{x}-\mu_k)\right)},$$

($\mu_k \in \mathbb{R}^N$ are means and Σ_k are covariance matrices, positive-definite real $N \times N$ matrices which we assume to be non-singular, $|\Sigma_k|$ being the corresponding determinant); the coefficients p_k, known as the *mixing probabilities*, satisfy:

$$p_k \geq 0 \qquad \text{and} \qquad \sum_{k=1}^{K} p_k = 1. \tag{3}$$

Finally, Θ denotes the $K(1+N+N^2)$ dimensional vector:

$$\Theta = ((\theta_1, p_1), \ldots, (\theta_K, p_K)),$$

where

$$\theta_k = (\mu_k, \Sigma_k)$$

is a vector containing all the coordinates of the means μ_k and all the entries of the covariance matrix[3] Σ_k. That is, Θ is a vector containing all the parameters of the given Gaussian mixture. The conditons in (3) guarantee that f is indeed a density function.

Mixtures of Gaussian functions provide good models of clusters of points: each cluster corresponding to a Gaussian density with mean somewhere in the centroid of the cluster, and with a covariance matrix somehow measuring the spread of that cluster. Conversely, given a set of points in $\mathbb{R}^N$, one can try to find the mixture of Gaussian functions with a certain number of summands that best fits those points, using a method known as *expectation maximization* (see section 2.3 in [6]). This is known as the *density estimation problem*. This gives a technique to study the way a set of points is distributed in space.

Therefore, if S is any set of points in $\mathbb{R}^3$, one can use the expectation maximization method to get a mixture of Gaussian functions describing it as a set of clusters, in a way that reflects the distribution of those points in space. In this manner one obtains a set of vectors θ_k together with its corresponding mix probability coefficient p_k. These can be seen as weigthed points (θ_k, p_k), setting the stage to use EMD to compare two sets of points. Using again the notations introduced in sections 2.1 and 2.2, if $S = (\mathbf{P}_i)_{i=1,\ldots,m}$ and $T = (\mathbf{Q}_j)_{j=1,\ldots,n}$ are two sets of points, one gets by this process sets $A = \{(\theta_k, p_k)\}$ and $B = \{(\eta_\ell, q_\ell)\}$ of weighted points, and we define:

$$d_{GMM}\left((\mathbf{P}_i)_i, (\mathbf{Q}_j)_j\right) = \mathrm{EMD}\left(\{(\theta_k, p_k)\}, \{(\eta_\ell, q_\ell)\}\right). \tag{4}$$

This yields a way to measure differences between two sets of points using GMM.

C. Plane Extraction

To be able to detect geometric primitives, i.e. simple shapes from which all, or almost all, other shapes are made off, is a crucial part of pattern recognition. Since a set of short line segments approximate the shape of almost anything, the most popular geometric primitive is precisely the line segment [7]. However, in practice one has to have a lower limit to the length of line segments, and then objects with parts smaller than that value are missed. This can cause obvious problems in path planning, for example. In [8], a method to overcome this problem is proposed. It consists of an algorithm to extract two kinds of geometrical primitives out of the data: line segments used to model all objects with a width exceeding 30 cm, and circles representing clusters of points that are closer to a specified point by a distance that is closer than a maximum predetermined value (see (1) on p. 905).

In the present work we have found that it was more appropriate for our aims to deal with planes, instead of just line segments. We have therefore implemented, in MatLab, a procedure to extract planes out of a given set of points. Actually we extract what one may call *fuzzy planes*, that is a neighborhood of a plane: all points that distance from it less than a pre-assigned value. This is done in a more or less straightforward way, as follows:

```
0: INPUT:
     - a set $S$ of points (data);
     - some threshold values: $\epsilon$, $\delta$, $\eta$;
     - some limiting values: $N$, $m$, $p$.
   OUTPUT: a set of fuzzy planes contained in $S$ together
   with the points in the respective neighborhoods.
1: Let $\mathcal{L}$ be a list (initially empty) containing the parameters of each plane $\pi$ already detected, together with a weight $w_\pi$ that represents the number of times the plane $\pi$ has shown up in step [3] below;
2: Select, at random, 3 points in $S$ until one gets non-collinear points $P_1, P_2, P_3$;
3: Determine $\pi = (a,b,c,d) \in \mathbb{R}^4$ such that $ax+by+cz = d$ is the cartesian equation of the plane that goes through $P_1, P_2, P_3$ and $\mathbf{v}_\pi = (a,b,c)$ is a unit vector;
4: Look in $\mathcal{L}$ for planes $\rho$ close to $\pi$ in the following sense:
     a) $\|\mathbf{v}_\pi - \mathbf{v}_\rho\| < \epsilon$ (for the planes to be roughly parallel);
     b) the distances from $P_1, P_2, P_3$ to the plane $\rho$ are less than $\delta$ (so the part of the planes $\pi$ and $\rho$ that contain data are somewhat close to each other);
5: Put all such planes $\rho$, and their corresponding weights $w_\rho$, in a temporary list $\mathcal{L}_{tmp}$, which we initialize with the plane $\pi$, assigning to it the weight 1;
6: If $\mathcal{L}_{tmp}$ ends up containing only the plane $\pi$, we add it to $\mathcal{L}$ with weight 1;
   if not, we remove from $\mathcal{L}$ all the planes in $\mathcal{L}_{tmp}$, while adding to $\mathcal{L}$ a plane obtained by the weighted average of all those planes, assigning to it a weight equal to the sum of all their weigths.
7: Repeat steps [2]–[6] $N$ times;
```

[3] Using a natural identification of the space of $N \times N$ real matrices with $\mathbb{R}^{N^2}$.

8: Find the planes in $\mathcal{L}$ with maximum weight. If this maximum is less than m, STOP. Otherwise, sequencially remove from the data set the points that belong to the η-neighborhood on each of these planes. Put these planes on a list $\mathcal{F}$, together with the number of points in their η-neighborhood.

9: Repeat [1]–[8] for the new data set, the points leftover after [7]. If there are less that p points, STOP.

At the end, we obtain a list of fuzzy planes, $\mathcal{F}$, extracted from the given set of points, together with the number of points in an appropriate neighborhood of each one of those planes. Now, the planes obtained are nothing but a set of four parameters, which can be looked at simply as a 4D point. Therefore, we again get, from any given set of points, a set of weigthed points of the form (π, w_π) and apply to it the EMD metric to compare sets of points. Using one more time the notations of sections 2.1 and 2.2, if S and T are two sets of points, one obtains through the process just described two sets A and B of weighted points, and we define:

$$d_{PE}(S,T) = \text{EMD}(\{(\pi, w_\pi)\}, \{(\pi', w_{\pi'})\}). \qquad (5)$$

IV. Results and Discussion

In order to evaluate the methods for change detection described in section 3, we made a series of tests with simulated data. We started by building a set of 300 points in 3-dimensional space, simulating the readings of a perfect laser, with the shape of a corridor with two parallel walls and a ceiling. On this set of points we then introduced random errors, normally distributed, with zero mean and a variance going from 0.00001 to 0.1, by factors of 10. After that, we moved a certain number of points out of one of the walls of the corridor, simulating a box of various sizes in that corridor. The number of points of that box was made to vary from 3 to 33% of the total number of points. Finally we increased the number of points of our initial corridor to experimentally study the complexity of the various methods.

A. Methods' behaviour for different errors and different size of object

Figures 1–5 display the results for EMD distance variation as the percentage of change increases, comparing the behaviour of the three methods we studied: principal components analysis (PCA), Gaussian mixture models (GMM) and plane extraction (PE).

These graphs clearly show that the GMM method is the most stable as the percentage of change increases in the environment, while PCA is the most unstable change detector.

B. Methods' sensibility to different error sizes

To evaluate the sensibility to errors, we generated, for each one of the variances studied above, 30 different datasets, and calculated the standard deviation of the EMD values obtained when applying each method. The results are rather curious, as shown in figure 6.

This confirms what the results presented in the previous section already suggested: that the GMM method presents a

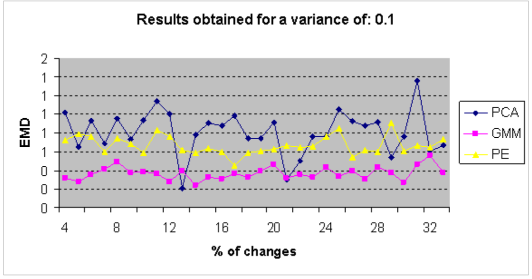

Fig. 1: The behaviour of PCA, GMM and PE for a variance of 0.1

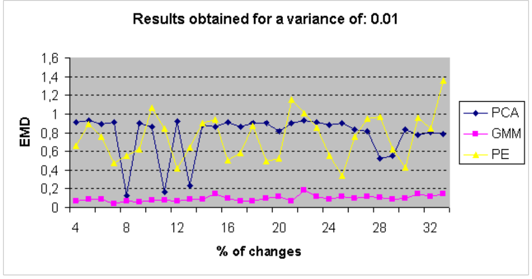

Fig. 2: The behaviour of PCA, GMM and PE for a variance of 0.01

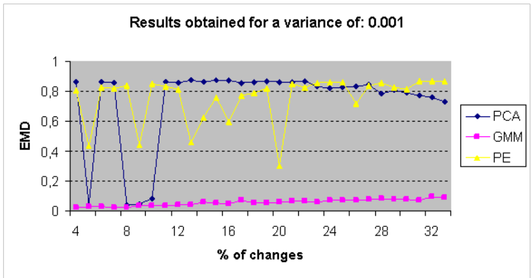

Fig. 3: The behaviour of PCA, GMM and PE for a variance of 0.001

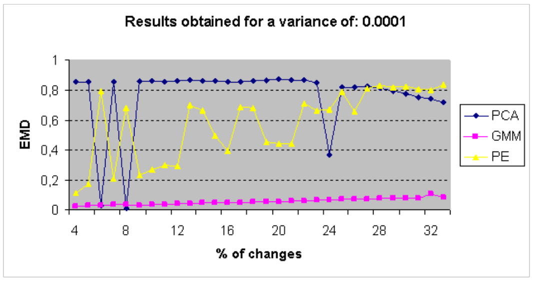

Fig. 4: The behaviour of PCA, GMM and PE for a variance of 0.0001

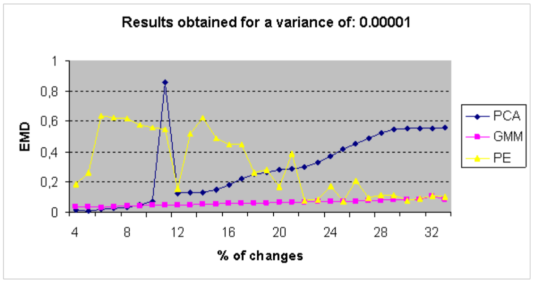

Fig. 5: The behaviour of PCA, GMM and PE for a variance of 0.00001

	standard deviation		
Variance	PCA	GMM	PE
0.1	0.3154	0.0840	0.0813
0.01	0.3821	0.0177	0.0796
0.001	0.4132	0.0062	0.3274
0.0001	0.4153	0.0026	0.3172
0.00001	0.4004	0.0037	0.3623

Fig. 6: Statistical behaviour of PCA, GMM, PE

much better behaviour than the other two, with a much lower standard deviation, and that improves as the errors decrease.

C. Experimental computational complexity

To evaluate the computational complexity of each method, we analysed the average time for 30 experiments, in seconds, needed to deal with datasets with a number of points ranging from 147 to 91875. Figure 4.3 shows how PCA and GMM performed.

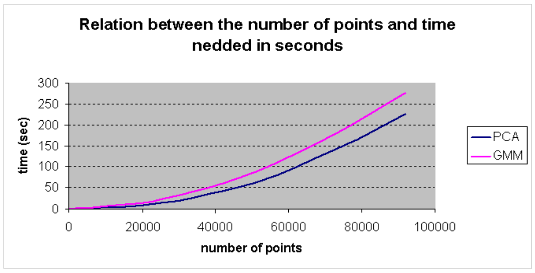

Fig. 7: The behaviour of PCA and GMM

The increase of time needed, as the number of points increases, it is quite similar for both PCA and GMM. We note that, while we used an implementation of GMM in the C language, the PCA implementation used was in MatLab. That of course has an impact on the time each algorithm takes for a given number of points, but not on the rate of change as the number of points increase.

The PE method turned out to be much more time demanding, but our implementation made in MatLab can certainly be improved, and used for other applications. However, given the excelent results obtained for GMM method we did not pursue that issue here.

V. Conclusions

The experiences and results described in the previous section, strongly support the conclusion that GMM performs much better than PCA and PE, detecting change in a robust way as errors are introduced in the data readings.

The plane extraction seems to be a fairly simple method to locate some changes in the environment. We did played a little with it, and it certainly seems be worthy of further study. (...)

References

[1] H. Durrant-Whyte and T. Bailey. Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms. *Robotics and Automation Magazine*, June, 2006.

[2] T. Bailey and H. Durrant-Whyte. Simultaneous Localisation and Mapping (SLAM): Part II State of the Art. *Robotics and Automation Magazine*, September, 2006.

[3] J. González J.L. Blanco, J.-A. Fernández-Madrigal. Toward a unified bayesian approach to hybrid metric-topological slam. In *IEEE Transactions on Robotics*, volume 24, 2008.

[4] C. Tomasi Y. Rubner and L. J. L. Guibas. A metric for distributions with applications to image databases. In *Proceedings of the 1998 IEEE International Conference on Compute Vision*, pages 59–66, 1998.

[5] Jolliffe I.T. *Principal Component Analysis*. 2nd ed., 2002.

[6] Pekka Paalanen, Joni-Kristian Kamarainen, Jarmo Ilonen, and Heikki Kälviäinen. Feature representation and discrimination based on gaussian mixture model probability densities-practices and algorithms. *Pattern Recogn.*, 39(7):1346–1358, 2006.

[7] Reina A. Gonzales J., Ollero A. Map building for a mobile robot with a 2d laser rangefinder. In *Int. Conf. Rob. & Autom.*, pages 1904–1909, 1994.

[8] H. Xu J. Vandorpe, H. Van Brussel. Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2d ranger finder. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, 1996.