TÉCNICO LISBOA

Localização indoor de um robô móvel usando imagem de profundidade do teto

João Paulo Lança Rodrigues
(Licenciado)

Dissertação para obter o grau de Mestre em

Engenharia Mecânica

Júri

Presidente:	Prof. João Rogério Caldas Pinto
Orientador:	Prof. Carlos Baptista Cardeira
Co-Orientador:	Prof. Paulo Jorge Oliveira
Vogal:	Prof.

Outubro 2013

The only place success comes before work is in the dictionary.

Vince Lombardi

Acknowledgments

The completion of this dissertation was possible due to the help and commitment of a group of people, to which I would like to thank here.

First, I would like to thank Prof. Carlos Baptista Cardeira, Prof. Paulo Jorge Coelho Ramalho Oliveira for their advice, support, guidance and motivation that allowed me to complete this work successfully.

Secondly, I would like to express my sincere thanks to Fernando Paulo Neves da Fonseca Carreira, without his help, friendship and dedication the dissertation would have been much less enjoyable.

Furthermore, I would like to thank Sr. Luís Raposeiro and Eng. Camilo Christo for their availability, helping fixing all the hardware problems. A special thank to every friend I had during my years in the IST for their friendship, for all the great moments and for putting the bar high and forcing me to reach it.

Finally, without the support of my family this work would not be possible and therefore, they deserve all my love.

Abstract

The main objective of this dissertation is to design, implement and test an indoor localization method that relies on data from a Kinect depth camera as main sensor. The proposed localization method relies on Markov localization using Visual odometry as building block.

Firstly, changes and adjustments to the Markov localization were made in order to achieve the desired performance. Secondly, in order to tackle the localization method weakness, a new building block, visual odometry, was tested to replace the classical wheel odometry. Finally, the new building block was merged to the localization method, resulting in a computationally efficient and high performance indoor localization method.

All tasks carried out throughout this thesis are presented and discussed, allowing a more accurate assessment on the proposed methods.

Keywords

- Ceiling;
- Indoor;
- Mobile robot localization;
- Depth image;
- Markov localization;
- Visual odometry;

Resumo

O objetivo principal desta dissertação é desenhar, implementar e testar um método de localização de robôs móveis para ambientes interiores, que utiliza dados capturados por uma câmara de profundidade da Kinect como sensor principal. O método de localização proposto é o Markov localization usando odometria visual como método secundário.

Primeiro, mudanças e ajustamentos foram feitos na Markov localization com o objetivo para atingir o desempenho desejado. Em segundo lugar, para combater as desvantagens do método de localização, um método secundário novo, odometria visual, foi testado para substituir a odometria de rodas clássica. Finalmente, o novo método secundário é incorporado no método de localização, resultando num método de localização computacionalmente eficiente e com desempenho elevado.

Todas as tarefas realizadas durante esta tese são apresentadas e discutidas, permitindo uma avaliação mais precisa sobre o método proposto.

Palavras Chave

- Tecto;
- Indoor;
- Localização de robô móvel;
- Imagem de profundidade;
- Markov localization;
- Odometria visual;

Contents

List of Figures

List of Tables

Abbreviations

GPS Global Positioning System
SLAM Simultaneous Localization And Mapping
PCA Principal Component Analysis
MCL Monte Carlo Localization
RGB Red-green-blue
WO Wheel odometry
VO Visual odometry
SLAM Simultaneous Localization And Mapping
SIFT Scale Invariant Feature Transform
SURF Speeded Up Robust Features
NNR Nearest Neighbour ratio
PDF Probabilistic density function
BBF Best-Bin-First
CS Consensus set
MSS Minimal Sample Sets
Mixture-MCL Mixture Monte Carlo Localization
KF Kalman filter
RMCL Reverse Monte Carlo Localization
EKF-SLAM Extended Kalman Filter - Simultaneous Localization And Mapping
SEIF Sparse Extended Information Filter
FastSLAM Fast Simultaneous Localization And Mapping
KL Karhunen-Loève
EKF Extended Kalman Filter
IST Instituto Superior Ténico
VOM Visual odometry with mapping
VGA Video Graphic Array
MLVOM Markov Localization with Visual odometry + mapping
MLWO Markov Localization with Wheel odometry

1

Introduction

Contents

1.1 Problem Presentation

In Robotics, the localization problem is of utmost importance. Without a good localization method, it is very difficult for that robot to be controlled, for the simple fact that it does not know where it is or where it is going to be in the next moment. This field in Robotics has been a great challenge to the scientific community in the area of mobile robotics [2, 3], therefore many solutions were proposed. Nowadays, the Global Positioning System (GPS) is the standard solution for outdoor environments, as in a global coordinate frame can be obtained the position of a mobile robot with great accuracy. However, for indoor environments, or any other environment where the GPS signal is not available, are required the use alternative approaches to obtain the mobile robot position and attitude.

There are many solutions for this problem, but none is as dominant as GPS, in outdoor application. There are many localization methods for indoor environment. All these methods have their strengths and weakness, however it is not the localization method that is the only important aspect in indoor localization, the sensor that acquire the surrounding environment information is also very important. Nowadays, the Computer Vision techniques are common practice in this situation, due to the large amount of information that can be extracted from an image [4–6]. Since the main sensor are cameras, the camera position and its direction are the two points that define each type of approaches.

The first approach was taken directly from nature, turning the vision sensor into the robots eyes, in other words, the cameras are on the front of the robot and are pointed to the floor ahead of the robot. The next approach was to use a 360 degree camera to acquire information, this made possible for the robot to have a much wider view of the environment. Finally, the last approach if to use a vision sensor to capture ceiling images. Apart from some exceptions, the ceiling always have the same elements in the same place and have enough information to be used for the mobile robot localization. In addition to that, a ceiling based localization is much less sensitive to changes in the room layout. It is this approach which is explored in this dissertation, along with other improvements.

1.2 Motivation

This dissertation aims to develop an indoor localization method using ceiling information acquired by a depth camera and it can separated in three steps, which one corresponding to an adversity solved. All experimental tests were done in the control, automation and robotic laboratory in Instituto Superior Técnico and there were many problems to overcome. The environment light condition, the ceiling elements repeatability and the hardware flaws, like wheel slippage and digital compass errors, were the main problems faced when developing this localization method. Overall, whenever a problem was solved, there was an improvement on the development of the indoor localization method.

1.2.1 RGB vs Depth camera

In an industrial or laboratory environment there is usually windows or ceiling panels where the natural sunlight comes inside. The influence on the amount of light inside the laboratory is not negligible, even when most of the artificial lights are turn on. When the localization is based on ceiling

images with a RGB camera, the artificial lights must be turned off, in order to not blind the camera (see Fig.1.1 and Fig. 1.2). However, with the artificial lights off, the change of the environment lights were enough to make every test unique. This fact turns the process of comparing a new method to an older method very difficult and therefore slows the development of the indoor localization method.

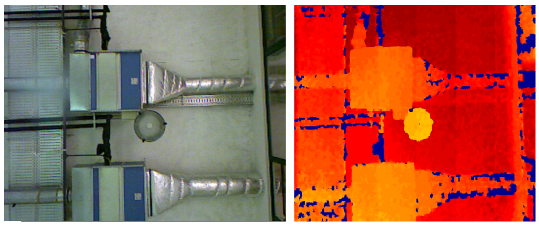

Figure 1.1: Ceiling image taken with RGB camera (letf) and with depth camera (right) with artificial lights off

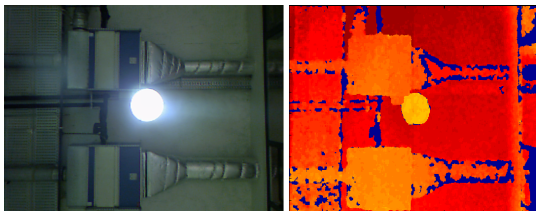

Figure 1.2: Ceiling image taken with RGB camera (letf) and with depth camera (right) with artificial lights on

In order to overcome this problem, it was decided to use a depth camera, which is immune to the light (see Fig.1.1 and Fig. 1.2). This fact alone should be enough to change the type of vision sensor, however the depth camera does not acquire as much information as the RGB camera. Consequently, it is necessary to confirm that it is not only possible to use depth camera instead of a RGB camera, but also if it has better performance and robustness.

1.2.2 Markov localization

When the whole laboratory ceiling was used in the indoor localization method, the problem with ceiling element repeatability appears. In other words, there are zones in the ceiling which are very similar, with the same elements in similar positions. In addition to that, the ceiling also has zones with little information (see Fig. 1.3). Both this factors influence negatively the indoor localization method, which was based on Principal Component Analysis method(PCA)[7]. Another problem faced when using this method with all laboratory the database size starts to be also a problem. Therefore, it is necessary to use a localization method that not only gives more importance to the robot motion than the PCA method but also uses a smaller database.

The solution founded was the Markov localization method. The necessary work to build the Markov localization database is much lower than the PCA database. The PCA needs a ceiling depth image for each possible position, one image each $0.3\ m^2$ in our experimental environment. On the other hand, the Markov localization just needs a ceiling depth image that covers all the trajectory. In our experimental environment, one image used in Markov localization is equivalent to 257 images for PCA. However, the PCA compresses the image database with a ratio of $99.7\ \%$ and, whereas one

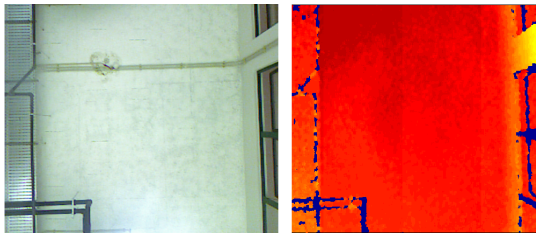

Figure 1.3: Ceiling image taken with RGB camera (letf) and with depth camera (right) showing a zone with low information

image used as Markov localization database is $54\ kB$, the PCA database correspondent is $42kB$. In addition to that, the Markov localization also gives great importance to the robot motion problem. Therefore, it is expected for this method to escape from the ceiling repeatability and have a better overall performance.

1.2.3 Visual odometry

In the section. 1.2.2, it was explained that one of the Markov localization strength is that it gives importance to the robot motion and the robot motion estimation must be very close to the reality. First, the robot motion was calculated by the wheel odometry(WO) plus a Kalman filter.

The WO is a mathematical method that uses the mobile robot kinematic equation to determine its motion, just using the wheel encoders as an input. This is a very common building block method in mobile robot localization, however the robot has a very significant wheel slippage in this particular case. In other words, the WO estimations are far from the reality and it is only possible to have a decent robot motion estimation if it is compensated by a Kalman filters, which uses a digital compass as an additional input. The digital compass finds the mobile robot attitude by analysing the magnetic fields surrounding the mobile robot, and since all the experimental test are done inside the laboratory, it is very difficult to avoid other machine that influence the magnetic field. Overall, the Kalman filter fuses two ”faulty” sensors to estimate the localization.

Since ceiling images are captured online , it was decided to use visual odometry (VO) instead of WO. This new building block find the robot motion by comparing the current ceiling image with the last ceiling image. If the results are good enough it will replace the WO and the Kalman filter. Consequently, the depth camera will replace the wheel encoders and the digital compass, making this mobile robot localization method dependent on only one sensor.

1.2.4 Markov localization with VO

If the VO results are a great improvement comparing to the WO, then the next step to is change the Markov localization method to use the data from the VO. Both methods are very similar, since both are a building blocks and both estimate the mobile robot attitude. Therefore, it is not necessary to make any major changes to the localization method in order to exchange the building block. With this test it will be possible to confirm that the VO is the solution to the problems presented before.

1.3 Thesis innovations and scientific contributions

The thesis innovations and scientific contributions depend on the starting point. In this case, the starting point was an indoor localization method developed by Carreira et al. in [8]. This indoor localization was a PCA based localization method that used ceiling RGB images captured by a webcam. The PCA localization method used the WO, correct by a Kalman filter, as a building block. Therefore, there are four major points that characterizes the starting point, which are the following:

- Used RGB images;
- Used three sensors in total, RGB camera, wheels encoders and a digital compass;
- Used a PCA based localization method, which is a uni-modal method;
- Only tested in a small area under a ceiling very rich in information;

Most of these characteristics were detailed in the previous section (section. 1.2), along with their negative problems.

In the end of this thesis, many innovations were made. Just by replacing the RGB images with depth images many problems were solved. Not only was reduced drastically the number of test and mapping necessary to obtain a decent result, but also proved that the depth image has enough information to be used in a localization method, even with a 5.5 meters high ceiling.

This was a big improvement when comparing to the starting point, however the most important characteristic of the localization method developed in this thesis is that it only uses one sensor. Whereas the starting point need three sensors, a vision sensor, wheel encoders and a digital compass, the new localization method only uses the vision sensor, which was possible with the change of building block. Instead of using WO corrected by a Kalman filter, that used two sensors, the new localization method only uses one sensor and, in addition to that, that sensor was already used. This innovation is incredibly important, not only the mobile robot is simpler and have much less hardware problems with the sensors, but also the wheel encoders and digital compass problems were solved. Moreover in the case where only one sensor is used, the tasks of synchronization and multi-rate date acquisition are simplified.

In addition to that, the use of Markov localization has a huge bonus, the work necessary to build its database is much less than the PCA, as it was explained previously in section 1.2. Furthermore, with the tests done using the whole laboratory, it was proved that the PCA cannot solve the global localization problems when there are ceiling areas with low information and with ceiling element repeatability, whereas the Markov localization can overcome all these problems successfully.

The following list summarize the new localization method characteristics:

- Use of depth images;
- Use only one sensor in total, depth camera;

- Use a Markov localization method, which is multi-modal and is able to solve most of the hardest mobile robot problems, like global localization and mobile robot kidnapping problem;
- Tested successfully in the whole laboratory, under a ceiling with areas of very rich to very poor information;
- The necessary work to build the Markov localization database is lower than the PCA, since 1 image Markov localization database correspond to 257 images for PCA database;

Regarding scientific contributions, the Markov localization is not a new localization method, but the use of depth image in this situation is a very recent. Even more uncommon it the use of ceiling depth images in a localization method. In this thesis, this localization method was tested in almost all possible situations, assessing its performance against the PCA localization method and others. In addition to that, some areas of my work were published in [8] (in acknowledgements) and other part of this work lead to a publication in an international scientific conference [9].

1.4 Thesis Outline

This dissertation is divided in 5 chapters:

- In **Chapter 1** the problem is presented, along with the motivation to tackle the adversities faced. Afterwards the innovations and scientific contributions of this thesis are summarized.
- **Chapter 2** shows the state of the art on the current mobile robot localization methods, detailing every possible solution to the problems faced.
- In **Chapter 3** the Markov localization and VO implementations are explained. New approaches used to solve the problem presented on chapter 1, along with the simulation results.
- **Chapter 4** displays the experimental results of each method explained in chapter 3.
- Finally, in **Chapter 5** the conclusion are outlined and the future research is revealed.

2

State of the art

Contents

2.1 Introduction

In this chapter, the state of the art of mobile robot indoor localization method is presented, while using a vision sensor to acquire environment information. A mobile robot localization method is composed by two independent blocks, robot motion and measurements, which are merged by a filter (Fig. 2.1).

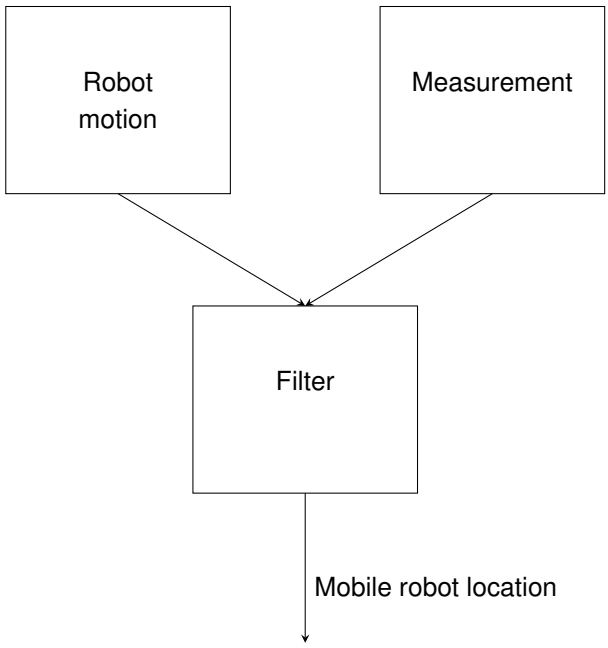

Figure 2.1: General architecture of any localization method

The information corresponding to the mobile robot motion is gathered by the wheel encoders, which are used in the mobile robot kinematics equations to determine the mobile robot motion. This method is called WO and it is the most famous odometry method. Nevertheless, due to the problems faced by this method, the VO is gaining more importance. The VO determines the mobile robot motion by comparing the current sample image with the image of the previous sample. In addition to VO, it is also possible to use feature detection, extraction and matching techniques to determine the mobile robot motion. These techniques have more computational cost, which is compensated by a good performance and robustness. Regarding the other block in figure 2.1, measurements, it is the information gather the main sensor, which, in this case, in a vision sensor that acquire surrounding information. Depending on the localization method, this information will be used differently .

The factor that define most of the mobile robot localization method characteristic is the filter that merges both blocks. There are three main filters with different strengths and weaknesses, which are Bayes filter, Kalman filter and Particle filter. The Kalman filter and the Bayes filters are the most common filters, whereas the Particle filter is know to be used in Monte Carlo localization (2.4). Therefore, a mobile robot localization is nothing but a combination of available techniques for each block, robot motion, measurements and filter. Before explaining the most used mobile robot localization methods, the feature detection and extraction techniques are explained in a succinct way, followed by feature matching and motion estimation technique. Afterwards, a brief overview of the filters is done. A

localization system may involve several techniques, namely:

- Feature detection, extraction, matching and motion estimation techniques;
- Bayes, Kalman and Particle filters;
- Monte Carlo localization (MCL);
- Simultaneous Localization And Mapping (SLAM);
- Principal Component Analysis (PCA);
- Markov localization;

2.2 Feature detection, extraction, matching and motion extraction techniques

Although these techniques are very common when a vision sensor is used in a localization method, in this thesis the features detection, extraction, matching and motion extraction techniques are out of the scope. In any case, in this section the most common techniques are presented briefly followed by a table summarizing their strengths and weaknesses. These techniques have, normally, a much higher computational cost, which is balanced with a great performance and robustness. First, the most used feature detection and extraction techniques are presented, followed by the feature matching techniques and ending with motion extraction techniques.

2.2.1 Feature detection and extraction techniques

Scale Invariant Feature Transform:

The Scale Invariant Feature Transform (SIFT) is an algorithm developed by David Lowe in 2004 [10] for the detection and extraction of interest points, also named features, from an image, which all together provide a local image description. This technique is normally used to find visual correspondences between images for different applications, like image alignment or object recognition. This method has proved to be very useful for many researchers. Se et. al. [11] developed a visual SLAM algorithm for a ground-based autonomous vehicle which was equipped with a trinocular camera system. The SIFT was used for both motion estimation and landmark description in this article. Later, Scleicher et. al. [12] developed a real-time visual SLAM algorithm based on 128 elements SIFT keys.

Speeded Up Robust Features:

Speeded up robust features (SURF) was developed by H.Bay and T.Tuytelaars [13] in 2006. This technique shares many conceptual similarities with the SIFT. Therefore, in order to convince researchers, the authors of SURF demonstrated experimentally that their new feature scheme outperforms SIFT and other popular feature detection techniques, both in terms of speed and accuracy [13, 14]. In addition to that, Murillo [15] obtained an efficient global localization combining SURF

features with a hierarchical method. In a similar way, Cummins [16] developed a topological SLAM localization method that used SURF features to extract information from captured images.

Precise Harris Corner Detector:

This feature detection and extraction technique was introduced by Harris and Stephens [17] as a low-level processing step to aid researchers trying to build interpretations of a robot's environment based on image sequences. The precise Harris corner detector was developed to overcome the limitations of the Moravec operator [18]. The result is a far more desirable detector in terms of detection and repeatability rate at the cost of a higher computation cost. Due to the great performance of this technique, there are many corner detectors based on the precise Harris corner detector, which only proves that it is a great feature detection and extraction method.

Pros and cons

In this section the strengths and weakness of each technique presented are summarised in the next table (table.2.1):

Table 2.1: Advantages and disadvantages of the feature detection techniques

Technique	Advantages	Disadvantages
SIFT	Invariant to translation, scaling and rotation; Low sensitivity to illumination changes and affine projection	Very high computational cost for feature extraction
SURF	More robust than SIFT in scale changes; Fastest feature extraction technique	Finds few keypoints; With larger scales, there is a necessity to interpolate the keypoint location;
Precise Harris corner detector	Very robust to scale and translation changes	Sensitive to noise; Anisotropic response; high computational cost to achieve a desired performance

2.2.2 Feature matching techniques

Nearest Neighbour ratio:

The keypoints are matched by identifying its nearest neighbour, which is defined as the keypoint with minimum Euclidean distance for the invariant descriptor vector. However, there are extra features that do not have a match and need to be discard. A global threshold on distance to the closest feature does not perform well, as some descriptors are much more discriminative than others. A more effective measure is obtained by comparing the distance of the closest neighbour to that of the second-closest neighbour, which is the nearest neighbour ratio (NNR) [10].

Best-Bin-First

The Best-Bin-First (BBF) [19] is very similar to the NNR. This new feature matching method trades the performance for speed, in other words, the BBF is faster than the NNR but the final result is

an approximation. This is most useful when dealing with huge amount of keypoints, in the order of $100,000$ keypoints. In this situation, any time saved by avoiding some calculation, will result in a enormous overall time reduction.

Pros and cons:

The following table (table. 2.2) presents the strengths and weaknesses of the feature matching techniques presented above.

Table 2.2: Advantages and disadvantages of the feature matching techniques

Technique	Advantages	Disadvantages
NNR	Can eliminate 90% of the false matches while discarding less than 5% of the correct matches; The final result is exact;	Become slower when the number of keypoint is very high;
BBF	Significantly faster than NNR when the number of keypoint is very high	The final result it is an approximation;

2.2.3 Motion extraction techniques

Random sample consensus:

Random sample consensus (RANSAC) was introduced by Fischler and Bolles [20], as a method to estimate the parameters of a certain model starting from a set of data contaminated by larger amounts of outliers. The outliers is a data that not fit the ”true” model instantiated by the ”true” set of parameters within some error threshold that defines the maximum deviation attributable to the effect of noise. The RANSAC is able to handle data sets where more than 50% are outliers. This method is widely accepted due to its simple implementation and robustness.

Least median square:

The least median square (LmedS) was introduced by Rousseeuw [21] and the basic concept is very similar to the RANSAC. Both methods are based on randomly selecting matched points and both are iterative methods. LmedS is devised to have a high breakdown point, usually defined as the smallest percentage of outliers needed to shift the estimate by an arbitrary amount.

Pros and cons:

In this section the strengths and weakness of each motion extraction technique presented are summarised in table. 2.3.

2.3 Bayes, Kalman and Particle filters

As it was said in section 2.1, the filter combines the robot motion block with the measurement block in order to find the mobile robot correct position. The main filters that already showed great

Table 2.3: Advantages and disadvantages of the motion estimation techniques

Technique	Advantages	Disadvantages
RANSAC	It is very accurate even with a significant number of outliers, over 50%; Is simple to implement; Can become more accurate, faster and more robust with some modifications	It is a slow method due to the number of iterations; May not reach the optimal value if the threshold is not correct;
LmedS	Robust estimation even with 50% of outliers;	Overall less accurate than RANSAC; It is a slow method due to the method iterative nature;

performance when used in a localization method can be divided in two groups, parametric and non-parametric filters. The parametric filters use a fixed function form of the posterior, such as Gaussians, being the Kalman filter the most common parametric filter used in mobile robot localization. As for the nonparametric filter, they do not rely on a Gaussian or similar function, they instead approximate posteriors by a finite number of values, each roughly corresponding to a region in state space. Both particle filter and Bayes filter, which will be explained in the following sections, are nonparametric filters.

2.3.1 Bayes filter

Bayes filters probabilistically estimate a dynamic system's state from noisy observations. This filter represent the state at time t by random variable x_t. At each point in time, a probability distribution over x_t called belief represents the uncertainty. Bayes filters aim to sequentially estimate such beliefs over the state space conditioned on all information contained in the sensor data. Therefore, Bayes filter is a probabilistic technique for data fusion. The main advantages of this method is that it is a nonparametric filter, or it is multi-modal. In other words, in the probabilistic map there is more than one bump, one corresponding to the correct position and others corresponding to others possible positions. Using this filter makes the localization method much more robust and able to overcome the some of the hardest robotic problems, such as the mobile robot kidnapping problem. The weakness of this filter is the computation time increases exponential with the system dimensions.

Since the Bayes filter can be applied in continuous or discrete state space, there are two types of Bayes filters, Continuous Bayes filter and Discrete Bayes filter. Apart from some algorithm differences, the difference between both Bayes filters is that, unlike the Continuous Bayes filter, the Discrete Bayes filter can be applied to problems in a discrete or continuous state space. When this filter is applied in discrete state space it is called Discrete Bayes filter, however when it is applied in continuous state space it is not Continuous Bayes filter, it is called Histogram filter. When using a discrete filter in a continuous state space, in addition to act as a filter, it is also an approximate inference tool for continuous state spaces, decomposing it into finitely many regions, and represent the cumulative posterior for each region by a single probability value, just like a histogram graphic. The Bayes filter algorithm is detailed in the Markov localization algorithm in chapter 2.4 in [4] and in 2.7. Since the

Bayes filter is the base of the other filters, the number of application of the Bayes filter is enormous.

2.3.2 Kalman filter

The Kalman filter (KF) is an optimal way to fuse observations that follow a Gaussian distribution. For a large class of linear system with Gaussian uncertainty, the KF is the perfect estimator. Under certain assumptions, the KF has many interesting characteristics, which are explained in the next paragraph.

The KF is optimal for linear models and Gaussian noise. One of the reasons the filter performs well is because it uses all available information that it gets, good or bad information, as long as the Gaussian and Linear approaches remain valid. This allows KF to make an overall best estimate of a state. Furthermore, the KF is recursive, which brings the useful property that not all data needs to be kept in storage and re-processed every time when a new measurement arrives. In addition to that, the KF is a data processing algorithm, which is useful for the reason that only knowledge about system inputs and outputs is available for estimation purposes. Variables of interest can not be measured directly. Due to the use of Gaussian distribution, this filter is uni-modal or a parametric filter, in other words, the probabilistic map only has one bump. However, the number of units does not increase as much as the Bayes and it is continuous. The KF algorithm is explained in detail in section 2.5.1. The PCA-based position system proposed in [8] uses a KF to estimate the position.

2.3.3 Particle filter

Particle filter, also known as Sequential Monte Carlo localization, exploits the idea of representing the posterior distribution $p(X_{1:k}|Z_{1:k})$ though a finite set of particles that can be used to estimate any property of $p(X_{1:k}|Z_{1:k})$ in an ordinary Monte Carlo estimation framework. When a new observation $Z_{1:k}$ arrives, the particles are updated in order to represent the new posterior $p(X_{1:k+1}|Z_{1:k+1})$. During this update, the particles with more importance survive to the next iteration, whereas the particles with lowest importance are eliminated. The closer a particle is to the estimated position, the more importance that particle have.

A main computational problem with the general approach is that the dimension of the distribution increases with time and the computation time increases exponential with the system dimensions, just like the Bayes filter. However, since the Particle filter has a resource-adaptive algorithm, this problem is not as bad as in the Bayes filter, despite being far from the KF. The resource-adaptive algorithm means that it can adapt the number of parameters to represent the posterior online. The Particle filter algorithm is detailed in [22] and in the Monte Carlo Localization section (section. 2.4). Since this filter is also multi-modal or a nonparametric filter, the Particle filter is known as a direct competitor of the Bayes filters. Recently, the Particle filter has gained a huge popularity in certain robotics problems [4], being mostly used in the Monte Carlo Localization.

2.3.4 Pros and cons

The pros and Cons of all the presented filters are summarized in the following table (tab.2.4):

Table 2.4: Advantages and disadvantages of the filters techniques

Technique	Advantages	Disadvantages
Bayes filter	Multi-modal; Able to solve successfully many hard robotics problems; Can be used in continuous or discrete state space	Computation time increases exponential with system dimension; Lower overall performance, comparing with the Particle filter, in certain robotics problems
Kalman filter	Continuous; Computation time only increases quadratically with system dimension;	Uni-modal; hardest to program
Particle filter	Multi-modal;resource-adaptive algorithm; easy to program; Continuous	Computation time increases exponential with system dimension;

Despite all the advantages and disadvantages, all these filters are very important for mobile robot localization method, and no one is completely inferior to another filter.

2.4 Monte Carlo Localization

The main purpose of mobile robot localization method is to estimate the robot's position and orientation and Monte Carlo Localization (MCL) is no exception. Normally, the robot's position and orientation is estimated at time k and using a set of measurements $z_{1:k} = \{z_1, z_2, ..., z_k\}$ from the environment and the movements $u_{1:k} = \{u_1, u_2, ..., u_k\}$ of the robot. On the other hand, in MCL [23], the probability density function $p(x_k|z_{1:k}, u_{1:k})$ is represented by a set of M random samples $\chi_k = \left\{x_k^i, i = 1...M\right\}$ extracted from it. These random samples are called a particle, where each one represents a hypothesis of the true state of the robot $x_k^i = (x^i, y^i, \theta^i)$. In this method, each particle have a weight that determines the importance of that particle. The set of samples defines a discrete probability function that approximates the continuous belief.

The idea of estimating the mobile robot position recursively using particles is not new, although most work on this area is recent. Despite being known as particles filters [24, 25] by the statistical literature, recently computer vision researchers proposed the same algorithm under the name of condensation algorithm [26]. In robot localization field, the particle representation has a range of characteristics that make it unique. Firstly, the particle filters can accommodate arbitrary sensor characteristics, motion dynamics and noise distributions. Secondly, they are universal density approximators and they focus computational resources in the most relevant areas, reducing the computational cost. In addition to that, particle filters also can adapt to the available computational resources. Finally, the particle filter is not hard to implement which make it more appealing to researchers than other localization method.

However, the MCL has flaws too, and most comes from the stochastic nature of the approximation. It is possible for the MCL to lose track of a well-localized mobile robot if the sample set size is not big enough, because it fails to generate a sample in the right position. One of the main downfall of this localization method is the huge difficulty in solving the kidnapped robot problem, since there might not

be any old samples near the new mobile robot pose. Furthermore, the MCL may fail with perfect and noiseless sensors, which is counter-intuitive. All these problems can be overcome and some of the solutions, like generating sample consistent with the most recent sensor reading [27] or augmenting the sample set through uniformly distributed samples [28], are mathematically questionable. On the other hand there are other solutions, like Mixture-MCL [29] and Reverse Monte Carlo Localization [30], that have better performance and are not mathematically questionable. Both solution will be presented after the MCL algorithm.

2.4.1 MCL algorithm

The initial set of particles represents the initial knowledge $p(x_0)$ about the mobile robot position on the map. When the particle filter algorithm is used in global localization, the initial belief is a set of poses drawn according to a uniform distribution over the robot's world map. If the initial position and orientation are partially known up to some small margin of error, the initial belief is represented by a set of samples drawn from a narrow Gaussian centred at the known starting pose of the mobile robot. The Monte Carlo Localization algorithm can be separated in two steps, which are described in the next lines:

Prediction Phase: At time t a set of particles $\overline{\chi}_k$ is generated based on the set of particles χ_{k-1} and a control signal u_k. This step uses the motion model $p(x_k|x_{k-1};u_k)$. In order to represent this probability function, the movement u_k is applied to each particle while adding a pre-defined quantity of noise. As a result, the new set of particles $\overline{\chi}_k$ represents the density $p(x_k|z_{1:k-1};u_{1:k})$.

Update Phase: In this second phase, for each particle in the set $\overline{\chi}_k$, the observation z_k obtained by the robot is used to compute a weight ω_k^i. This weight represents the observation model $p(z_k|x_k)$ and is computed as $\omega_k^i = p(z_k|x_k^i)$. In the following subsection we propose different methods for the computation of this weight. The weights are normalized so that $\sum \omega_k^i = 1$. As a result, a set of particles accompanied by a weight $\overline{\chi}_k = \left\{x_k^i, \omega_k^i\right\}$ are obtained.

The resulting set χ_k is calculated by resampling with replacement from the set $\overline{\chi}_k$, where the probability of reuse each particle is proportional to its importance weight ω_k^i, in accordance with the literature on the Sampling Importance Resampling algorithm [31], also known as SIR. Finally, the distribution $p(x_k|z_{1:k};u_{1:k})$ is represented by the set χ_k.

2.4.2 Mixture-MCL

The key idea of Mixture-MCL is to modify the way samples are generated in MCL. Mixture-MCL combines original MCL sampling with a dual of MCL. The dual MCL is basically an invert MCL sample process, which is done by exchanging the roles of the proposal distribution and the importance factors in MCL. More specifically, dual MCL generates samples of the state $x_k^{(i)}$ by virtue of the following proposal distribution:

$$\overline{q}_k = fracp(o_k|x_k)\pi(o_k) \tag{2.1}$$

where:

$$\pi(o_k) = \int p(o_k|x_k)dx_k \tag{2.2}$$

Here the normalizer, $\pi(o_k)$, is assumed to be finite, which is the case for mobile robot localization in environments with finite size. Dual MCL can be viewed as the logical inverse of the sampling in original MCL. In other words, rather than guessing the state x_k^i using odometry then uses the sensor information to adjust the importance of the state guess, dual MCL guesses position using the most recent sensor measurement then uses odometry to adjusts, with the odometry estimation, the importance factor in accordance with the prior belief $Bel(x_{k-1})$. Consequently, the dual proposal distribution possesses complimentary strengths and weaknesses: while it is ideal for highly accurate sensors, its performance is negatively affected by measurement noise. The major dual MCL advantage is that when the distribution of $p(o|x)$ is narrow, which is the case for low-noise sensors, the dual sampling can be more effective than original MCL. In addition to that, it also recovers faster in the mobile robot kidnapped problem than the original MCL.

In dual MCL there are a couple of approaches to calculate the importance factors for the proposal distribution, $\overline{q}_k$. These approaches are very similar, being one easier to implement and mathematically most straightforward but with low performance and vice-versa. As it was said before, Mixture-MCL is the combination between the original MCL and the dual MCl. The extra computational cost of doing two MCL method is rewarded with a huge increase in performance, in certain situations. The Mixture-MCL accuracy is monotonic in perceptual noise, in other words, with more accurate sensors give better position estimation. However, the use of the dual MCL carries a disadvantage, a wrong sensor measurement have a terrible effect on the final result, since , afterwards, almost all samples will be generated at the wrong place.

2.4.3 Reverse Monte Carlo Localization

This method is a combination between two methods, Monte Carlo Localization (MCL) and grid-based method, in [32] it is used the Markov localization as the grid-based method. The Reverse Monte Carlo Localization (RMCL), benefits from the advantages of these two methods, while avoiding their disadvantages. The basic idea behind this method is to converge to several cells by Markov localization, then produce a limited number of samples inside those grids to find the mobile robot position. In the original MCL, the number of samples is increased to decrease the bias in the result. In RMCL, since we converge by selecting the cells with maximum probability, the bias is already lower.

This method is called Reverse Monte Carlo Localization because it reverses the order of the MCL events. In the original MCL, the samples are used along the world map, then the position is estimated. As for the RMCL, first the place where the samples should converge is estimated, then the all the samples are used in that particular area. This is the main advantage of RMCL, by focusing all the samples on a smaller area, it is necessary to use less samples, which leads to a much lower computational cost. Even thought the RMCL has this great advantage, it has not been implemented in many situation, other than in [32] and related work in robot soccer, there are not much more successful implementation of this MCL variant.

2.4.4 Pros and cons

Overall, these three method mention above are capable of estimate the mobile robot position and orientation, in certain situations. The original MCL has its flaws, which some where solve by the Mixture-MCl and RMCL, which have their own flaws. The advantages and disadvantages are displayed in the next table (table.2.5).

Table 2.5: Advantages and disadvantages of the MCL techniques

Technique	Advantages	Disadvantages
MCL	In normal situation MCL can estimate the mobile robot position very well; It may easily adapt to different noise distribution and sensor models;	It can lose track of a well-localized mobile robot if the sample set size is not big enough; Has a huge difficulty solving the kidnapped robot problem; Ambiguity coming from symmetries;
Mixture-MCL	Is monotonic in perceptual noise; Overall better performance, especially in the kidnapped robot problem and other less common situations;	Higher computational cost than MCL; A wrong sensor measurement will influence negatively the final result
RMCL	Reduce computational cost due to use samples near the estimated position	It has a hard time solving the kidnapped robot problem

2.5 SLAM

Simultaneous Localization and Mapping (SLAM) was original developed by Hugh Durrant-Whyte and John J. Leonard [2] who took off the work of Smith, Self and Cheeseman [33]. Durrant-Whyte originally named this method SMAL, but later changed it to a more appealing name, SLAM. This method aims to solve the problem of a robot being autonomously able to build a map of an unknown environment and simultaneously localizing itself in that environment. This ability would make a mobile robot truly autonomous.

The main characteristic of SLAM is the fact that the mobile robot localization is done using the map created during the experimental test to complement the other sensors information. However, the original SLAM cannot achieve the desire performance and many solutions were developed. From all the solution, the most known and common method is the Extended Kalman Filtering (EKF-SLAM), which will be explained with more detail than the other solutions. As for the others solutions, only the Sparse Extended Information Filtering (SEIF) and Particle filtering (FastSLAM) will be presented in this section.

2.5.1 EKF-SLAM

This SLAM technique results from the combination between the EKF and the original SLAM method. The following table (table.2.6) resumes the iteration between both parts that created the

EKF-SLAM:

Table 2.6: EKF and SLAM response to all possible events

Event	SLAM	EKF
Robot moves	Robot motion	EKF prediction
Sensor detects new landmark	Landmark initialization	State augmentation
Sensor observers known landmark	Map correction	EKF correction
Mapped landmark is corrupted	Landmark deletion	State reduction

As the robot performs SLAM, at a time instant k, let x_k be the vector representing the current robot state(position and orientation), u_k be the control input applied at time $k-1$ to move the robot to state x_k, which can also be the WO estimation during the interval $[k-1;k]$). Let m be the set representing the locations of all landmarks and z_k be the set of landmark observations at time instant k.

In EKF-SLAM, the motion model is described by the following equation:

$$x_k = f(x_{k-1}, u_k) + w_k \tag{2.3}$$

where function f models the robot kinematics and w_k accounts for the un-modelled kinematics that is approximated to a zero mean uncorrelated Gaussian noise with covariance Q_k. As for the observation model, it is described as:

$$z_k = h(x_k, m) + v_k \tag{2.4}$$

where function h describes the observation geometry and v_k accounts for the observation errors, which is approximated to a zero mean uncorrelated Gaussian noise with covariance R_k.

After defining this variables, the EKF-SLAM can be applied. A simple way to understand this method it to divide it in three different steps, the robot motion, the prediction and then the update step. For the first part, robot motion, the mean and covariance of the previous distribution $P(x_k, m_k | Z_{0:k}, U_{0:k}, x_0)$ are calculated by equation 2.5 and equation 2.6 respectably.

$$\begin{bmatrix} \widehat{x}_{k|k} \\ \widehat{m}_k \end{bmatrix} = E \begin{bmatrix} x_k \\ m_k \end{bmatrix} \quad |Z_{0:k}\Big] \tag{2.5}$$

$$P_{k|k} = E\left[\begin{pmatrix} x_k - \widehat{x}_k \\ m_k - \widehat{m}_k \end{pmatrix} \quad \begin{pmatrix} x_k - \widehat{x}_k \\ m_k - \widehat{m}_k \end{pmatrix}^T \quad |Z_{0:k}\right] \tag{2.6}$$

This variables are calculated using an iterative prediction and one correction algorithm. As for the second step, prediction, at time instant k, the mean (eq. 2.5.1) and covariance (eq. 2.5.1) are determine by the following equation:

$$\widehat{x}_{k|k-1} = f(\widehat{x}_{k-1|k-1}, u_k) \tag{2.7}$$

$$P_{k|k-1} = \nabla f P_{k-1|k-1} \nabla f^T + Q_k \tag{2.8}$$

In equation 2.5.1, the ∇f is the Jacobian of f calculated with $\widehat{x}_{k-1|k-1}$.

Finally, the update step mean and covariance are calculated as follows:

$$\begin{bmatrix} \widehat{x}_{k|k} \\ \widehat{m}_k \end{bmatrix} = \begin{bmatrix} \widehat{x}_{k|k-1} & \widehat{m}_{k-1} \end{bmatrix} + W_k \left[Z_k - h(\widehat{x}_{k|k-1}, \widehat{m}_{k-1}) \right] \tag{2.9}$$

$$P_{k|k} = P_{k|k-1} - W_k S_k W_k^T \tag{2.10}$$

where:

$$S_k = \nabla h P_{k|k-1} \nabla h^T + R_k \tag{2.11}$$

$$W_k = P_{k|k-1} \nabla h^T S_k^{-1} \tag{2.12}$$

The ∇h is the Jacobian of h calculated with $\widehat{x}_{k|k-1}$ and $\widehat{m}_{k-1}$.

By far EKF-SLAM is the most used SLAM technique. However, it carry the flaws of its component. One of the main problems is that the EKF-SLAM computational cost grows quadratically with the number of landmarks. Secondly, it use linearised models of non-linear motion and observation models. In addition to that, this technique is extremely sensitive to a wrong association between the landmarks and the observation. Even with this flaws, most of the EKF-SLAM implementations are successful. For example, Davison [34] tested an EKF-SLAM structure with active vision, using a high-performance stereo head cameras to acquire surrounding information. This approach was developed to operate in small environment, but rich in useful landmarks. Other implementations of EKF for visual SLAM can be found in [35–37].

2.5.2 SEIF

This solution is based on extended information filters, which are computationally equivalent to an EKF. The difference is that the information is presented on a different way, instead of the covariance matrix, SEIF uses the inverse of the covariance matrix, also know as the information matrix [38]. The implementation of this method was done by Yufeng Lui and Sebastian Thrun [39], in an outdoor environment and using a car as the mobile robot. The results obtained were similar to the results using an EKF, however, for bigger maps SEIF results were superior to EKF results.

2.5.3 FastSLAM

The FastSLAM algorithm was introduced by Montemerlo and Thurn [40] and it uses a particle filter to estimate robot position and EKF for estimating landmark locations. This method is based on one SLAM particularity, if the robot position is known, the individual landmark measurements are independent. In other works, if the robot position is known then the estimation of landmark locations can be decoupled into an independent estimation problem for each of the landmark. Due to this

process separation, the FastSLAM is able to overcome the huge growth of the computational cost when the number of landmark increases. The FastSLAM original algorithm, FastSLAM 1.0, needs to execute great part of the SLAM method for each landmark, which waste processing time and increase the computational cost. The FastSLAM 2.0 is an significant improvement in many areas when comparing to FastSLAM 1.0. It solves the computational cost problem by using an efficient data structure.

Results obtained using this solution can be seen in [41], where it is used a stereo camera and landmarks extracted using the SIFT. In this example, position estimation where done with a frequency of $3Hz$ with a 4% error. Looking for less recent work, Montemerlo and Thrun also worked with this technique [40, 42] and proved that it is possible to achieve great results by using the FastSLAM technique.

2.5.4 Pros and cons

In conclusion, all techniques presented above have their strengths and weakness and characteristics that make them different from the others. These advantages and disadvantages are summarised in the next table (table.2.7).

Table 2.7: Advantages and disadvantages of the SLAM techniques

Technique	Advantages	Disadvantages
EKF-SLAM	Works reasonably well for small number of feature and distinct landmarks; Can estimate the position in real time due to the submapping;	Needs to simulate errors as a Gaussian noise, which is not true in every situation; Non-linear model linearisation may cause divergence
SEIF	Low computational cost due to the disparity of the information matrix;	Very hard to recover the covariance matrix if it is necessary to use it afterwards;
FastSLAM	Multi-hypothesis data association (robustness); With a large number of landmark is able to simulate non-linear or non-Gaussian models;	The particle number grows exponential with the state dimension; Over optimistic position estimation

2.6 Principal Component Analysis

The Principal Component Analysis [43] (PCA) is a dimensionality reduction technique used in signal and image processing, communications and other scientific fields. This method is based on a linear transformation, the Karhunen-Loève (KL) transformation, which allows for the optimal approximation to a stochastic signal, in the least squares sense. In addition to that, it can also be used as a signal expansion technique with uncorrelated coefficients for dimensionality reduction. These and other features make the KL transform, and consequently the PCA, very useful in signal application and other scientific areas. The KL transform is used in data data compression, image and voice processing, data mining, exploratory data analysis and pattern recognition.

As mentioned before, the PCA main advantage is the ability to compress a set of high dimensional vectors into a set of lower dimensional vectors, which is possible since it is an optimal linear scheme, in terms of mean squared error. In addition to that, there are another two advantages, which are the following: the PCA model parameters can be computed directly from the data, by diagonalising the ensemble covariance; and given the model parameters, projection into and from the bases are computationally inexpensive operations, $O(nN)$.

With this characteristics it is possible to implement this dimensionality reduction technique in a localization method. The PCA-based position sensor will be detailed after the PCA background. In addition to use this technique in a localization method it is also possible to use it to remove the missing data existing in an image. As it was proved by Oliveira in [44], the PCA outperforms some other methods for missing data recovery. In a brief way, this is done by compress the image with the PCA then doing the inverse process to obtain a new image without the missing data.

2.6.1 PCA background

The data compression is obtained through the use of a database eigenspace approximation by the best fit eigenvectors. This method makes the PCA an algorithm that has a high compression ratio and requires reduced computational resources. The PCA eigenspace is created based on a set of M stochastic signals $x_i \in R^N,\ i = 1, \ldots, M$.

This eigenspace is characterized by the corresponding mean $\mathbf{m}_x = \frac{1}{M}\sum_{i=1}^{M} \mathbf{x}_i$. The purpose of the KL transform is to find an orthogonal basis to decompose a stochastic signal x, from the same original space, to be computed as $x = \mathbf{U}\mathbf{v} + \mathbf{m}_x$, where vector $v_i \in R^N$ is the projection of x in the basis,i.e. $\mathbf{v} = \mathbf{U}^T(\mathbf{x} - \mathbf{m}_x)$. Matrix $\mathbf{U} = [\mathbf{u}_1\mathbf{u}_2 \ldots \mathbf{u}_n]$ should be composed by the N orthogonal column vectors of the basis, verifying the eigenvalue problem.

$$\mathbf{R}_{xx}\mathbf{u}_j = \lambda_j \mathbf{u}_j, j = 1, \ldots, N, \tag{2.13}$$

Where $\mathbf{R}_{xx}$ is the covariance matrix, computed from the set of M experiments using

$$\mathbf{R}_{xx} = \frac{1}{M-1}\sum_{i=1}^{M}(\mathbf{x}_i - \mathbf{m}_x)(\mathbf{x}_i - \mathbf{m}_x)^T \tag{2.14}$$

Since the eigenvalues are ordered, i.e. $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N$, the choice of the first $n \ll N$ principal components will lead to an approximation of the stochastic signals given by the ratio on the covariances associated with the components, i.e. $\sum_n \lambda_n / \sum_N \lambda_N$. Due to this approximation, the PCA is able to drastically reduce the dimension and, consequently, reduce the computational complexity.

2.6.2 PCA-based positioning system

This dimensionality reduction technique can be also used in a mobile robot localization method, usually just called PCA. The PCA-based position sensor [7, 45] merges the information gathered

by vision sensors with the information acquired from odometry, in order to estimate a mobile robot position and orientation.

This localization method has four main goals, which are to extract the most important information from the data acquired by the sensors, to compress the size of the data set by keeping only the most important information, to simplify the description of that data set and to analyse the structure of the observations and the variables. When comparing to other localization method, it is the ability to compress the huge data set size that gives this method an edge over the competition. In localization methods that are feature based techniques, the computational cost can get intolerable with the data set growth. This factor encouraged researchers to find other methods that turn this process more efficient. With the PCA advantages, the use of PCA in mobile robot for self-localization has been explored [46]. However, all these approaches use front or omnidirectional cameras, causing the algorithm to address problems of occlusion or comparison with images in different planes.

In order to implement the PCA in this mobile robot localization method, it is necessary to do some adjustments. The set of stochastic signals, which create the PCA eigenspace, are acquired by the Kinect depth sensor, or the RGB camera, considering an area with N mosaics in two dimensional space, $N = N_x N_y$, where N_x and N_y are the number of mosaics in x and y axis, respectively. After computing the KL transform using 2.13 and 2.14, the eigenvalues must be ordered and the number n of the principal components to be used should be selected. Afterwards, the data ensemble mean m_x and the matrix transformation with n eigenvectors, $\mathbf{U}_n = [\mathbf{u}_1 \mathbf{u}_2 \dots \mathbf{u}_n]$ are computed and saved for further use. By using this mean, any problem with the data captured from the sensor will damage the PCA mean value computation and, consequently, the position estimation.

Following the computation of the projection of the signal x into the basis, which is done by using $\mathbf{v}_i = \mathbf{U}_n^T(\mathbf{x}_i - \mathbf{m}_x), i = 1, ..., M$, the next step is to search on a given neighbourhood δ the mosaic that verifies the following equation.

$$\forall_i \| \left[\widehat{x}\widehat{y}\right]^T - [x_i y_i[^T \|_2 < \delta, r_{PCA} = \min_i \|\mathbf{v} - \mathbf{v}_i\|_2; \tag{2.15}$$

After determine which mosaic *i* is the closest to the present input, its center coordinates (x_i, y_i) will be selected as the x_m and y_m measurements.

2.6.3 Pros and cons

The PCA localization method composed by the PCA sensor and KFs to fuse the data with odometry and digital compass has many advantages when comparing to other localization methods, which are the following:

- It is fast and the computational cost is low;
- The database size is very low when comparing to the number of images in that database;
- Under Gaussian assumption for the disturbances, the localization system estimates in real time the position and slippage with global stable error dynamics.

Nevertheless, there are disadvantages when using the PCA method as localization method:

- The work to build the database is greater than other localization methods and it increases greatly with larger trajectories, which can discourage the use of this method;
- The surface used in the database must be static and rich in information;
- The PCA has a difficulty in overcoming the global localization problem, due to the ceiling element repeatability;
- Due to the difficulty of estimating the mobile robot correct position in a ceiling with low information and repeatability, the PCA solves that problem by using local localization, which makes almost impossible for that implementation to solve the mobile robot kidnapped problem;

In addition to these disadvantages, using just WO to estimate the robot motion will result in a poor performance, as it was explained in section. 1.2.3. In order to help the fusion of the PCA position estimation with the WO position estimation, it is used an KF, which needs information acquired from a digital compass to correct the WO estimation. This brings the number of sensors onboard to a total of 3, wheel encoders, digital compass and a vision sensor. Overall, the PCA is a very good localization method for indoor environments, and especially when the researchers want to keep computation power, energy and data storage onboard to a minimum.

2.7 Markov localization

Markov localization addresses the problem of state estimation from sensor data. Markov localization is multi-modal, or in other words, is a probabilistic algorithm. Instead of maintaining a single hypothesis as to where in the world a robot might be, Markov localization maintains a probability distribution over the space of all such hypotheses. The probabilistic representation allows it to weigh these different hypotheses in a mathematically sound way.

Before explaining the Markov localization algorithm, the basic concept will be explained with a simple example. Consider the environment depicted in Figure 2.2. For the sake of simplicity, let us assume that the space of robot positions is one-dimensional. Now suppose the robot is placed somewhere in this environment, without knowing the starting position. Markov localization represents this state of uncertainty by a uniform distribution over all positions, as shown by the graph in the first diagram in Figure 2.2. Now let us assume the robot queries its sensors and finds out that it is next to a door. Markov localization modifies the belief by raising the probability for places next to doors, and lowering it anywhere else. This is illustrated in the second diagram in Figure 2.2. Notice that the resulting belief is multi-modal, reflecting the fact that the available information is insufficient for global localization. Notice also that places not next to a door still possess non-zero probability. This is because sensor readings are noisy, and a single sight of a door is typically insufficient to exclude the possibility of not being next to a door.

Now let us assume the robot moves a meter forward. Markov localization incorporates this information by shifting the belief distribution accordingly, as visualized in the third diagram in Figure 2.2. To

account for the inherent noise in robot motion, which inevitably leads to a loss of information, the new belief is smoother (and less certain) than the previous one. Finally, let us assume the robot senses a second time, and again it finds itself next to a door. Now this observation is multiplied into the current (non-uniform) belief, which leads to the final belief shown at the last diagram in Figure 2.2. At this point in time, most of the probability is centred around a single location. The robot is now quite certain about its position.

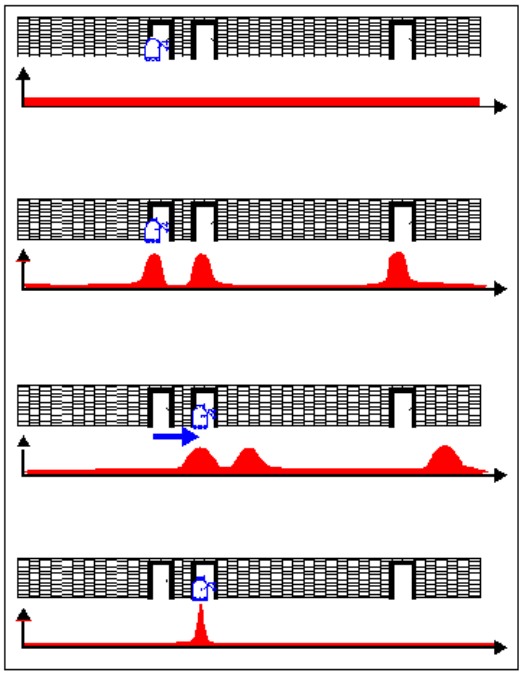

Figure 2.2: The basic idea of Markov localization [1]

2.7.1 Pros and cons

In this section it is presented a list of advantages and disadvantages of the Markov localization when comparing with other localization methods.

Advantages:

- Multi-modal;
- Database is small even when the world is very large;
- Is able to overcome most of the hardest problem in robot localization;
- It can estimate correctly the mobile robot position even when the sensors acquire low information;

Disadvantages:

- Can be badly influenced by a poor odometry estimation;
- Needs time to converge to the mobile robot correct position

3

Proposed methods

Contents

3.1 Introduction

When faced with the challenge of creating an indoor localization method that can find the mobile robot position and its motion, it was necessary to be certain that the chosen method has not only a good performance but is also versatile and adaptive to many situations. Therefore, the chosen method to replace the PCA method is the Markov localization, which is probabilist method unlike other localization methods. This method is detailed in section 3.2.

Afterwards, the attention was turned into the localization method building block. The first building block used was the WO and it showed room for improvement, in overall performance and in the reduction of the number of sensors required. The VO was chosen to be used as building block, which is explained in section 3.3. Afterwards, it was also implemented the VOM as a straight improvement to the VO, explained in detail in section 3.4.

In conclusion, this chapter contains the detailed information about the proposed methods, Markov localization, VO and VOM, applied to overcome the adversities faced.

3.2 Markov localization

As said in section 3.1, this localization method has a major difference from other localization method, the Markov localization is a probabilistic method. In order to explain it better, its architecture is represented in the next figure (see Fig. 3.1).

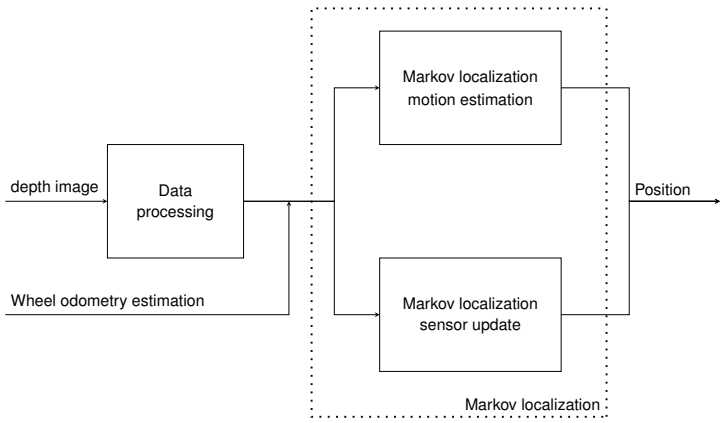

Figure 3.1: General architecture of the Markov localization method

As it is possible to see, this method can be separated in three steps, Data processing, Markov localization motion estimation and Markov localization sensor update. The first step, Data processing (section 3.2.1), consist in all processes applied to the captured depth image before being used in the localization method. The last two steps, which combined are the Markov localization method, are explained in detail section 3.2.2.

However, before applying this mobile robot localization methods there are some requirements that need to be met. As it will be explained in section 3.2.2, the Markov localization need to have a world map. This world map is a ceiling depth image large enough to contain all the trajectory. For most of the trajectories, a single well-placed image is enough, but for larger trajectories it is necessary to merge

multiple ceiling images in order to create one depth image large enough. Like all the images captured by the Kinect Depth camera, they are corrupted with missing data, which must be removed, see the following section (section. 3.2.1) for more detailed information. In addition to these two operations, it is also require to resize the world map. Due to the way that Markov localization works, the size of the world map has a relation of $52[pixels/meter]$, due to the value of certain variables, like velocity and sample time.

Another aspect of the Markov localization is the need of a building block. The Markov localization uses the robot motion estimated only by WO, unlike the implementation done by Carreira et al. in [7]. Although the Markov localization is applied offline and the mobile robot real trajectory is already known, the Markov localization still uses the mobile robot motion estimation in order to simulate an online test. After meeting all the requirements, having a world map and a building block, it is possible to use the Markov localization method. The following sections explain in detail all the steps since the captured depth image until the mobile robot position estimation.

3.2.1 Data processing

Since the data acquisition until the begin of the localization method, there is a need to apply a combination of processes, which can be called Data processing. After the Kinect Depth camera captures a ceiling image, it is necessary to remove the existing missing data. It was shown by Carreira et al. in [8] that the mean substitution method provides good results for missing data.

In the image matrix it is possible to find the missing data, since it corresponds to entries the have value $0mm$ when it should not be $0mm$. This can be understood if the way that Kinect Depth camera work is analysed carefully. Just like a sonar sensor, the Kinect Depth cameras has an emitter, which emits an infra-red grid, and a receptor. The Kinect Depth camera determines the distance between an object and the camera by measuring the disturbances on the emitted pattern when it reaches the receptor.

After some tests, it was easy to conclude that there is always missing data in a ceiling depth image, with a percentage of missing data around 10%. Since it is a small percentage of missing data, replacing the missing data with a mean value is acceptable. However, if the percentage of missing data was much higher, then the depth image should be discarded, since there is not enough uncorrupted information in that depth image. This approach is one of the most used in this type of situation, only usable when the missing data percentage is low. For each depth image, it is calculated a different mean value to replace the missing data, which is determined by the following equations:

$$m_x(j) = \frac{1}{c(j)} \sum_{i=1}^{M} l_i(j) x_i(j), j = 1, \ldots, N \tag{3.1}$$

where $c(j)$ is the number of j^{th} components for a set of M depth images $x_i \in \Re^N, 1, \ldots, M$ without missing data. The counter c is a vector with length N defined by:

$$c = \sum_{i=1}^{M} l_i \tag{3.2}$$

All missing data in the acquired database is replaced by the mean value of the corresponding component. In other words, if there is a missing data in the j^{th} component of the i^{th} depth image, the missing value $x_i(j)$ is replaced with the value of $m_x(j)$. This process is very noticeable, the next image (Fig. 3.2) shows a ceiling depth image before and after removing the missing data. Notice that colours changed due to the colormap, the lowest value is no longer zero, which was the missing data value in the ceiling depth image.

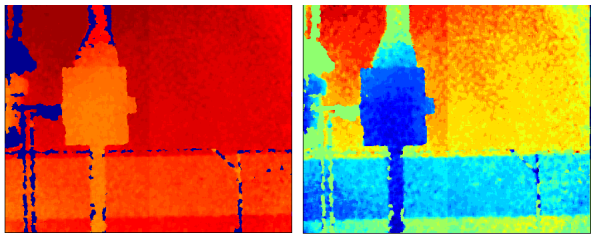

Figure 3.2: Before and after removing the missing data from a ceiling depth image

Although the missing data problem is solved by replacing it with the mean value, the missing data is not removed. Removing the missing data from one image is the process that replace the missing data with the correct value, which is very unlikely to happen. As it was explain in section. 2.6, the PCA can be used to completely remove the missing data of an image, as Oliveira proved in [44], however this process is not used in this dissertation, since the solution presented previously is enough for the Markov localization to work.

3.2.2 Markov localization algorithm

The Markov localization purpose is to compute a probabilistic distribution over all possible location in the environment. In this method, the robot estimate that it is in position l at time t, $Bel(L_t = l)$. The position l is a vector with three different values, x, y and φ. The last one correspond to the robot's orientation and first and second value are Cartesian coordinates. Since the robot does not know the start position, $Bel(L_0)$ is uniformly distributed through out the world map to simulate the robot location uncertainty, which is called probabilistic map. This method can be separated in two steps, when the robot move and when the robot senses.

In the first step, the method simulates the robot motion and reshape the probabilistic map. In this particular case, the Markov localization is applied offline and the real robot motion is known, which makes this step easier for this method. The probabilistic map is reshaped by moving all the probabilities in this map with the robot movement, the new belief is calculated with the following equation (eq. 3.3)

$$\widehat{Bel}(L_t = l) \leftarrow Bel(L_{(t-1)} = l') \tag{3.3}$$

The second step is when the method uses the information acquired by the Kinect Depth camera to updates the original probabilistic map. However, only part of the image taken by the Depth camera

is used in this step, a square places in the center of the image with size $(X_{center} - w : X_{center} + w, Y_{center} - w : Y_{center} + w)$, where X_{center} and Y_{center} is the center of the image and w is a pre define value. By changing the last value, it is possible to change the speed of the Markov localization and its performance. The information taken by the captured image is compared with world map, creating a second belief map.

$$Bel'(L_t = l) \leftarrow (Bel(L_{(t-1)} = l') - dimr)^2 \tag{3.4}$$

Where $Bel'(L_t = 1)$ is the second belief map and $dimr$ is the square taken from the image captured by the Kinect Depth camera. After finishing this step, the Markov localization method has enough information to create the final belief map, $Bel(L_{t+1} = l')$. This last map is created by combining the other two belief maps (eq.3.5) and in the next iteration it will be used as the original belief map. Both belief maps ($\widehat{Bel}(L_t = l)$ and $Bel'(L_t = l)$) have areas where the robot is more likely to be and, when combined, the probability of the robot being in that area is increased even more. By repeating this process, the belief map is refined, ending with a small area with high probabilities, which is the robot location.

$$Bel(L_{t+1} = l') \leftarrow (\widehat{Bel}(L_t = l) \times Bel'(L_t = l)) \tag{3.5}$$

Overall, Markov localization method essentials are that the robot maintains a belief distribution $Bel(L)$ which is updated upon robot motion, and upon the arrival of images taken by Depth camera. Such probabilistic representations are well-suited for mobile robot localization due to their ability to handle ambiguities and to represent degree-of-belief.

The world map must be resized to a pre-defined size, so that the mobile robot moves 1 pixel in each sample. This fact is very important for the probabilistic map displacement. To simulate the mobile robot movement, the probabilistic map is displaced in the same direction as the mobile robot. This would not be a problem if the movement was 1-D, however, this case is 2-D. In order to displace the probabilistic map in this case, it is necessary to know the mobile robot attitude and its velocity in each sample. Depending on the mobile robot movement, the displacement is done by the following equations:

No movement:

$$Pw_n(x, y) = (1 - |sin(\psi)|)(1 - |cos(\psi)|)P_{n-1} \tag{3.6}$$

Movement in yy axis:

$$Pw_n(x, y + 1) = (1 - |sin(\psi)|)(|cos(\psi)|)P_{n-1} \tag{3.7}$$

Movement in xx axis:

$$Pw_n(x + 1, y) = (|sin(\psi)|)(1 - |cos(\psi)|)P_{n-1} \tag{3.8}$$

Movement in both axis:

$$Pw_n(x + 1, y + 1) = (|sin(\psi)|)(|cos(\psi)|)P_{n-1} \tag{3.9}$$

Where Pw_n is the probabilistic map in sample n, Pw_{n-1} is the probabilistic map in the previous map and ψ the mobile robot attitude estimated.

3.2.3 Markov localization simulation results

Before applying the new indoor localization method with real data, it is necessary to test with a virtual world and with a virtual trajectory. This is a normal step in development, simulation tests allows the programmer to analyse how the new localization method performs when faced to certain adversities and to make the necessary adjustments. The most important advantage on doing simulation tests is to test each possible problem one at a time, fixing every problem individually, which the best way to troubleshooting.

The new localization method was tested with a virtual world and with a virtual trajectory. Although it is possible to create the most complex and the longest trajectory imaginable, there are not many advantages on testing those type of trajectories, since most of the problems faced wont be faced when using the localization method with real data. Therefore, the virtual trajectory created started as simple as a straight line and ended with long and complex trajectories, as it can be seen in the following figures (Fig.3.3, Fig. 3.4 and Fig. 3.5).

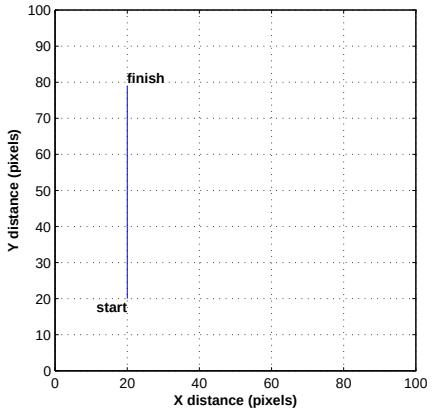

Figure 3.3: The straight line trajectory

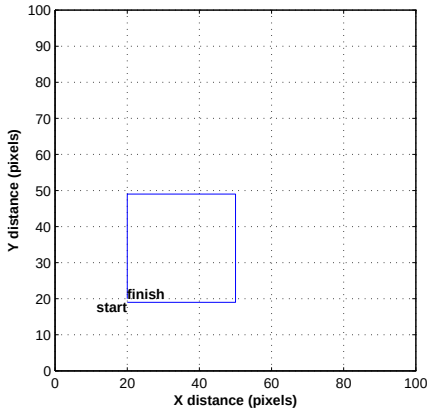

Figure 3.4: The square trajectory

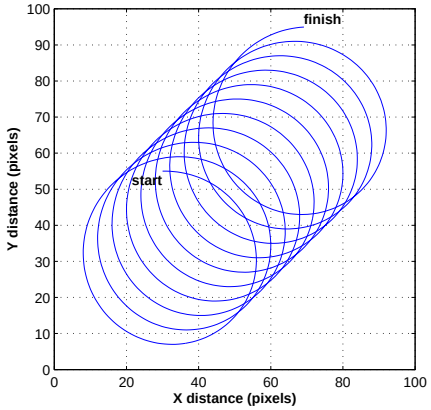

Figure 3.5: The circle trajectory

The virtual worlds were chosen to test the localization method to certain problems. One virtual world is a mathematical created world similar to a dome viewed from above, named ”Dome” (see Fig. 3.6), which is a world that is symmetrical in two axes.

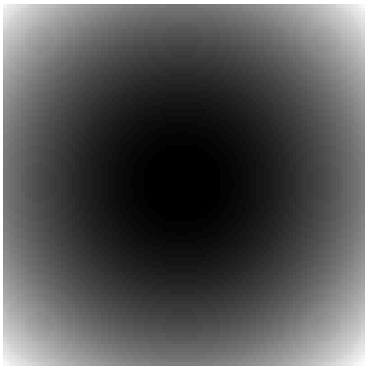

Figure 3.6: Dome virtual world used in the simulation

The other virtual world is an areal view of Instituto Superior Técnico, which can be see in the next figure (Fig. 3.7). This is the selected world due to the good amount of information and the existence of some similar zones, which is a good overall test to the new localization method.

In addition to use a different trajectory and a different world map, there are a couple of changes that need to be done in order to obtain these simulation results. The mobile robot sees the world as a 5×5 square without any rotation, due to the fact that the mobile robot sees the world rotated by the mobile robot attitude and before using the captured image in the localization method it is necessary to rotate it back. In this case, the mobile robot pose is already known in each iteration, which means that the probabilistic map displacement is perfect. However, in the experimental test case, the mobile robot pose is estimated, in other words, the probabilistic map displacement is far from perfect, turning into a major problem for the Markov localization method.

Simulation results

Figure 3.7: IST areal view virtual world used in the simulation

As it was said before, the virtual trajectories range from a very simple straight line to a longer and complex trajectory. These allowed to progressively challenge the new localization method and to fix any existing problem. In this section it will be only presented a table summarizing all the simulation results for each world (table. 3.1 and table. 3.2). First it will be presented the result with Dome world, followed by IST world. The full detailed results are in appendix A.2. The position error presented is calculated by following equation 3.10:

$$error = \frac{(real\ result - estimated\ result)}{real\ result} \times 100 \tag{3.10}$$

Table 3.1: Markov localization simulation results in Dome world

Trajectory	Position x error	Position y error
Straight line	0.339%	0.432%
Square	0.122%	0.213%
Circle	3.674%	3.747%

In the Dome world, the Markov localization is always able to find the mobile robot correct position almost immediately and keeping the correct estimation during all trajectory. In conclusion, the Markov localization is able to overcome the problems caused by this type of world, showing excellent results.

Table 3.2: Markov localization simulation results in IST world

Trajectory	Position x error			Position y error		
	Before	After	Sample	Before	After	Sample
Straight line	20.177%	0.01%	16	12.882%	1%	16
Square	26.353%	0.167%	16	20.118%	0.657%	16
Circle	11.268%	1.475%	29	18.969%	0.394%	23

Unlike the Dome world, with the IST world the Markov localization has more difficulty to find the mobile robot correct position. Therefore, the Markov localization has a very high error in the first part of the trajectory. However, the error after the Markov localization finding the correct position is almost zero. If the error mean was calculated using the whole trajectory, the value would indicate that the

Markov localization is not as good as it is, and that is why the mean error is divided in two, *before* and *after* the correct position is founded by the Markov localization. In the same table it is showed the sample where the *before* become *after*.

Comparing both worlds used, the Markov localization has a better results when the IST world is used. The difference between *before* and *after* is very visible, changing from a huge error into a very small error. The success of the Markov localization method in these virtual worlds and with these trajectories supported the application of this localization method with real data.

3.2.4 Markov localization kidnapping problem simulation

In addition to test the Markov localization in a global localization problem, which showed great results, the localization method is also tested in a kidnapping problem. Just like local and global localization problem, this problem is a standard test for any localization problem. Whereas the others problems let the mobile robot to follow its planed route, the kidnapping problem takes it to another difficulty by kidnapping the robot and putting it in another location and another direction. With this test it is possible to test the ability of the localization method to adapt and react to a strange situation. The mobile robot kidnap usually happens after the mobile robot localization method converged to the mobile robot correct position successfully, in other words, when the localization method is almost sure that the mobile robot is in the estimated position. After a significant change to the mobile robot position or direction, the localization method will have to change its estimation to the mobile robot correct position. This problem is one of the hardest problem a localization method might have to face. As it was said before there are two cases of mobile robot kidnap, a change of position and a change in both position and direction. Therefore, it was tested these two cases and, unlike the other localization problems, this one needs time for the localization method to correct the estimation after the kidnapped, so the Straight line and Box trajectory were not be used in this simulation due to their small size.

In order to be able to conclude that the Markov localization can solve this problem it is necessary to compare its performance with another localization methods performance. First, the Markov localization was tested against the PCA, which was implemented in the same as Carreira et al. approach in [**?**]. In order to compare multi-modal localization systems, the Markov localization was also tested against the Particle filter. This method is multi-modal, just like Markov localization, and should be able to overcome the mobile robot kidnapping problem.

PCA:

As it was said before, the PCA implementation is the same as in [7]. However, before applying the localization method it is necessary to build the PCA database. Therefore, the world map was divided into 30×30 images every 5 pixels. Just like in the experimental tests, the PCA in both cases has overlapping images(Fig. 3.8). Afterwards, the mobile robot sees a 30×30 image, which is used in the PCA localization method to find the position, resulting in the vision estimation. When this estimation is merged with the motion estimation with a KF, a new estimation is created, which is the

PCA localization method estimation, as it is possible to see in the figure 3.9. Both vision and motion estimation are approximated to a Gaussian distribution with a standard deviation of 15.

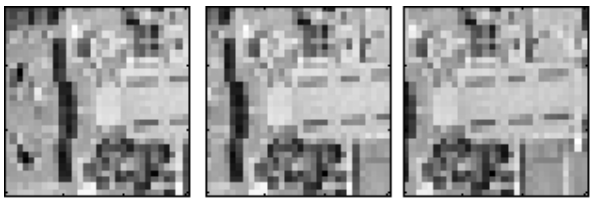

Figure 3.8: Sequence of images used to build the PCA database of world IST

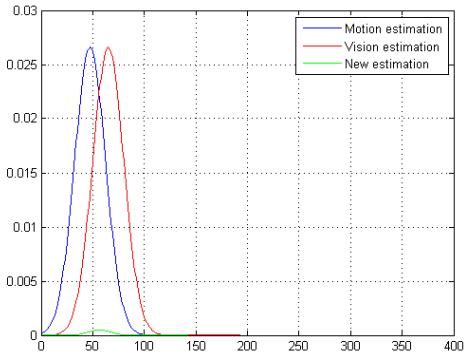

Figure 3.9: Example of merging both estimations to obtain the PCA localization estimation

Particle Filter:

The implimantation used in these simulation tests was the same approach explained in section . In this situation it was used 100 particles, randomly placed in the world, as it is possible to see in the Fig. . Each particle sees the world as a 5×5 square rotated by the particle attitude. These images are compared to what the mobile robot sees, another 5×5 image. The particle has an attiude noise of $0.12 \times \pi$ and a velocity noise of 0.01. As it was explained in section, in the end the particles are resampled, which in this case is done by a Matlab function named *randsample*.

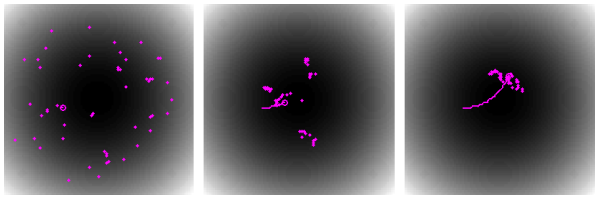

Figure 3.10: Example of the particles (magenta points) converging to the mobile robot correct position (magenta circle)

In conclusion, the localization methods will be tested with only one virtual trajectory in two different virtual worlds for two different kidnapping variants. Every kidnapping is done in sample 800, both the change of direction and change of position. The results will be summarized in the following tables, where *Before* is the error just before the kidnapped, *End* is the error in the end of the trajectory and *Sample* is the number of samples necessary to recover from the kidnapping.

Table 3.3: Mobile robot kidnapping problem simulation results, changing position

Localization method	Position x error			Position y error		
	Before	End	Sample	Before	End	Sample
Dome world						
PCA	0%	21.861%	*Never*	1.612%	36.095%	*Never*
Markov localization	5.555%	6.261%	89	8.064%	7.874%	136
Particle filter	2.778%	0.382%	326	3.29%	5.32%	282
IST world						
PCA	2.779%	15.165%	*Never*	1.613%	44.469%	*Never*
Markov localization	0%	0.905%	167	1.613%	0.5%	189
Particle filter	0.125%	0.474%	412	1.032%	1.049%	420

Table 3.4: Mobile robot kidnapping problem simulation results, changing position and direction

Localization method	Position x error			Position y error		
	Before	End	Sample	Before	End	Sample
Dome world						
PCA	0%	46.931%	*Never*	1.619%	9.638%	*Never*
Markov localization	5.555%	1.331%	313	6.451%	10.91%	114
Particle filter	0.543%	20.382%	549	1.742%	8.899%	*Never*
IST world						
PCA	2.778%	48.619%	*Never*	1.612%	13.456%	*Never*
Markov localization	0%	2.046%	207	1.612%	0.728%	57
Particle filter	0.014%	1.078%	318	1%	0.061%	328

Regarding the PCA method, it is not able to recover from the mobile robot kidnapping, ending always far from the mobile robot correct position. These results show that the PCA eigensapce would need to be enlarged to cope with repeatability scenarios in the ceiling. Even starting with a correct localization in the Dome world, the PCA loses track from the mobile robot correct path after the kidnapping.

As for the Particle filter, the method successfully overcame the mobile robot kidnapping problem, however the performance was not as good as the Markov localization. In every case, the Particle filter takes more time to recover than the Markov localization. In addition to that, during the simulation tests it was visible that few times the Particle filter was not even able to recover from the mobile robot kidnapping. This slightly lower performance in addition with being a slower method, when the particle number increase, support the conclusion that the Markov localization is a great method and it is not only able to solve the local and global localization, but also the mobile robot kidnapping problem. The full results are displayed in the appendix A.3

3.2.5 Markov localization improvements

In this section, it is presented a list of the points that were improved when changing the localization method, from the PCA localization method to the Markov localization. In the section 2.6, the main problems with the PCA localization method were detailed and, as expected, the Markov localization not only solved a great of the existing problems, but also brought others advantages. All the major advantages are summarized in the following list:

- Drastic reduction on the number of images needed to create the database, a Markov localization world map that is one image correspond to almost 200 images in PCA localization method;
- The Markov localization can overcome the problem of the ceiling elements repeatability and zones with very low information, unlike PCA localization method;
- The Markov localization can solve the mobile robot kidnapping problem better than the PCA method, in this situation;
- The Markov localization can pinpoint the mobile robot position with more precision, due to the PCA grid mapping;

As it was explained before, in these simulation results, the localization method tested above needed the mobile robot attitude and velocity in each iteration. In the PCA-based position sensor developed by Carreira et. al. in [7], these data was obtained by fusion between WO and the digital compass with a KF. In the next sections (section. 3.3 and section. 3.4), it will be tested a new approach to obtain this data, the visual odometry.

3.3 Visual odometry

In robotics, VO [47] is the process that predicts the motion based on consecutive images captured by cameras installed onboard of the robot and has the advantage to be more immune to wheel slippage than WO. In this section, a VO approach is detailed, to be used in robot localization in unstructured environments. The proposed method just requires a depth camera pointing to the ceiling and uses the captured depth information to compute the localization of the mobile robot, without the need of a previous mapping and any feature extraction. Thus, analysing the general architecture of the proposed system (Fig. 3.11), there are three blocks, the first being where the the raw data is transformed to usable data. The last two block, which are the VO method, receives the current image and the last image and estimate both the mobile robot attitude and the mobile robot velocity. These blocks will be detailed in this section.

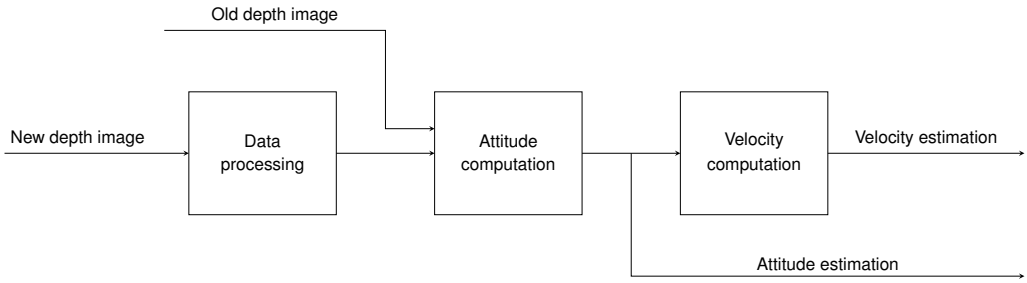

Figure 3.11: Architecture of the VO localization

3.3.1 Data processing

Before using the data acquired by the kinect depth camera, there are a couple of processes that need to be applied. First, the missing data problem must be solved in order to improve the localization

method overall performance. For a method that only uses depth images, the quality of those images is very important. As explained in section 3.2.1, the missing data problem is solved by replacing the missing data by the image mean. In addition to that, it is also necessary to do a circular crop, which consists on transforming the rectangular image in a circular image by removing the corners, using the most information possible. With a circular image, it is possible to easily rotate the image, without facing the problems that appear when rotating a non circular image in Matlab. In the next figure (Fig. 3.12), it is possible to see the evolution from a raw depth image until a depth image usable in this localization method.

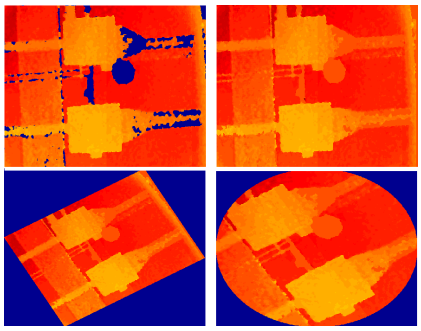

Figure 3.12: From raw image (top left), to missing data replaced (top right), to rotated 30° image (low left), to final image (low right)

3.3.2 Attitude computation

Following the sequence of the algorithm, the value of the mobile robot attitude is obtained comparing the captured depth image with the last depth image, testing possible turning angles of the robot. In order to improve the VO performance, this process is done twice. The first test focus on testing the biggest range possible, whereas the second refine the result obtained in the first test. Thus, considering a depth image $\mathbf{dimage}(k)$ captured in instant k, a set of possible rotated images is created based on the robot attitude in the previous instant of time for $\psi_j \in [\psi(k-1) - \Delta\psi, \psi(k-1) + \Delta\psi]$, resulting in j rotated images

$$\mathbf{dimr}_j = imrotate(\mathbf{dimage}(k), \psi_j), j = 1, ..., M \tag{3.11}$$

where M is the number of images to be analysed, to be selected in the implementation phase.

The new robot attitude is computed, finding the angle ψ_j that minimizes the mean square error between the current image, $\mathbf{dimage}(k)$ and the last image, $\mathbf{dimage}(k-1)$ (3.12)–(3.13),

$$\mathbf{m}_j = (\mathbf{dimage}(k-1) - \mathbf{dimr}_j)^2 \tag{3.12}$$

Finally, the robot attitude is obtained, finding the ψ_j that minimizes (3.12):

$$\psi(k) = \min_j(\mathbf{m}_j). \tag{3.13}$$

3.3.3 Velocity computation

The process of estimating the velocity is very similar to the process detailed in attitude computation (section 3.3.2). After the VO estimate the mobile robot attitude, the velocity is computed by testing different velocity values and finding the one that results in the best fit. Thus, considering a depth image $dimage$ captured in instant k and rotated by the obtained attitude in Section 3.3.2, resulting in $\mathbf{dimt}$. A set of possible displaced images along the direction of $\psi(k)$ is created based on the velocity of the robot in the previous instant inside the range $u_j \in [u(k-1) - \Delta u, u(k-1) + \Delta u]$. As in the attitude computation, the mobile robot velocity is obtained by the mean square error of the possible tested images (3.14)–(3.15).

$$\mathbf{mu}_j = (\mathbf{dimt}(k-1) - \mathbf{dimt}_j)^2, \tag{3.14}$$

where $\mathbf{dimt}_j$ is the image translated with the possible velocity u_j. Finally, the robot velocity is obtained, finding the u_j that minimizes (3.21):

$$u(k) = \min_j(\mathbf{mu}_j). \tag{3.15}$$

3.3.4 Visual odometry simulation results

As in section 3.2.3, it is necessary to apply this method in a virtual world with virtual trajectories in order to test the created localization method. The same IST areal image and the Dome world will be used as a world map in this simulation, along with the same three trajectories also used in Markov localization simulation tests. Unlike the Markov localization method, it is not necessary to analyse the position estimation error, since this localization method purpose is to estimate the mobile robot attitude. Therefore, the attitude estimation error means will be summarized in a table (table. 3.5), showing the attitude estimation error of each trajectory in each world.

Table 3.5: Visual odometry simulation results

Trajectory	Dome world	IST world
Straight line	$0\ \%$	$0\ \%$
Square	$0.002\ \%$	$0.001\ \%$
Circle	$0.241\ \%$	$0.127\ \%$

In the appendix A.1, it will be presented all the experimental test results in detail. When analysing the experimental results, it is possible to see that, as long as the trajectory is simple and short, the VO is capable to estimate the attitude almost without any error. However, when the trajectory is as long as the circle trajectory, the VO attitude error is slightly higher. When analysing the results showed in the appendix A.1.3 , it is possible to see in every situation the VO estimate the mobile robot correct position almost without error. In conclusion, the VO simulation tests allow to not only improve the localization method, but also to confirm that it is possible to estimate the mobile robot attitude using only two ceiling depth images. Although the attitude estimation got slightly worse when the trajectory

complexity and length increase, the localization method is able to overcome the many problems faced with the different worlds used. Overall, the results obtained in this simulation are exceptional, therefore giving more motivation to apply this method with real data.

3.3.5 Visual odometry improvements

In the section 1.2.3, the main problems of the WO were presented and with the implementation of VO, the problems were solved. All the advantages that came from this switch are summarized in the following list:

- Not influenced by wheel slippage;
- Although it increment the error overtime, it is much less than the WO;
- At least one sensor can be removed, wheel encoders, leading to a less hardware dependent localization method;

3.4 Visual odometry with mapping

This localization method is very similar to the last method, the VO. Whereas the VO compares the current image with the last image, the visual odometry with mapping (VOM) compares the current image with a map, which is created simultaneous with the localization method. This variant does not require any other sensors to create the map, in other words, it just need a depth camera pointing to the ceiling. Analysing the general architecture of the proposed system (Fig. 3.13), the method consist in the mobile robot localization and the construction of the environment map based on depth images captured from the ceiling. The first step of the method is the definition of the new position in the map, which is performed with the depth image and the knowledge that the robot has about the environment and, in a second step, the new depth image is added to the map, increasing the database.

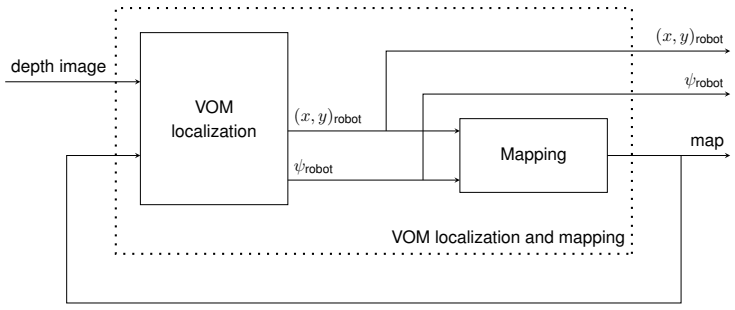

Figure 3.13: General architecture of localization and mapping system

Looking to Fig. 3.14, that details the proposed VOM localization method, the localization of the robot is performed considering the computation of attitude and position of the mobile robot. When comparing to the method presented in the section 3.3, there is no data processing block, since it is not necessary to remove the missing data or turn the rectangular image into a circular image.

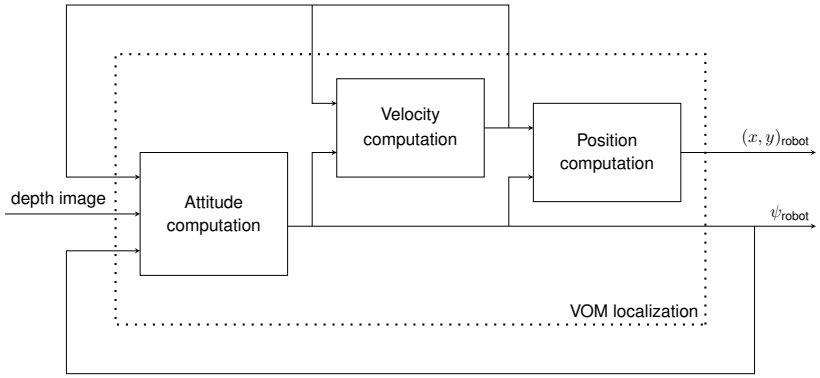

Figure 3.14: Architecture of the VOM localization method

3.4.1 Attitude computation

When comparing to the same block in the VO (section 3.3.2), there is only one difference due to the missing data existent in every depth image. Since the data used in the VOM is corrupted with missing data, there are only a couple of equations that must be modified to take in account this factor.

In the VOM method, the value of the mobile robot attitude is obtained by comparing the captured depth image with map, testing possible turning angles of the robot. Just like the CO previously presented, this process is done twice. The first test tests the widest range possible, followed by a refine test. Therefore, considering a depth image $\mathbf{dimage}(k)$ captured in instant k, a set of possible rotated images is created based on the robot attitude in the previous instant of time for $\psi_j \in [\psi(k-1) - \Delta\psi, \psi(k-1) + \Delta\psi]$, resulting in j rotated images

$$\mathbf{dimr}_j = imrotate(\mathbf{dimage}(k), \psi_j), j = 1, ..., M \tag{3.16}$$

where M is the number of images to be analysed, to be selected in the implementation phase.

To eliminate the possible disturbances caused by the missing data in signals the comparison between images is only calculated in pixels with non corrupted data, i.e. for values in the map and in the captured image with valid depth information.

The new robot attitude is computed, finding the angle ψ_j that minimizes the mean square error between the image stored in the map, in the previous position of the robot ($\mathbf{map}_{x,y}(k-1)$), and the rotation of the captured depth image (3.17)–(3.19),

$$\mathbf{m}_j = (\mathbf{map}_{x,y}(k-1) - \mathbf{dimr}_j)^2 \tag{3.17}$$

$$\bar{\mathbf{m}}_j = \frac{\sum_{i=1}^{N} \mathbf{m}_j}{nd} \tag{3.18}$$

where N is the number of pixels of map and nd, the number of pixels of $\mathbf{m}_j$ with depth information ($\mathbf{m}_j > 0\ mm$).

Finally, the robot attitude is obtained, finding the ψ_j that minimizes (3.18):

$$\psi(k) = \min_j(\bar{\mathbf{m}}_j). \tag{3.19}$$

3.4.2 Velocity computation

In a similar way, the velocity is computed by testing different values and finding the one that results in the best fit. Thus, considering a depth image $dimage$ captured in instant k and rotated by the obtained attitude in Section 3.4.1, resulting in $\mathbf{dimt}$. A set of possible displaced images along the direction of $\psi(k)$ is created based on the velocity of the robot in the previous instant inside the range $u_j \in [u(k-1) - \Delta u, u(k-1) + \Delta u]$. Following the same process that lead to the attitude computation, the robot velocity value is obtained by the mean square error of the possible tested images (3.20)–(3.22).

$$\mathbf{mu}_j = (\mathbf{map}_{x,y}(k-1) - \mathbf{dimt}_j)^2, \tag{3.20}$$

where $\mathbf{dimt}_j$ is the image translated with the possible velocity u_j.

$$\bar{\mathbf{mu}}_j = \frac{\sum_{i=1}^{N} \mathbf{mu}_j}{nd} \tag{3.21}$$

Finally, the robot velocity is obtained, finding the u_j that minimizes (3.21):

$$u(k) = \min_j(\bar{\mathbf{mu}}_j). \tag{3.22}$$

3.4.3 Position computation

After the computation of the attitude and the velocity of the mobile robot based on the depth information, the robot kinematics is used to allow the computation of the new position, based on the well know Euler discretization of the differential drive robot:

$$x(k) = x(k-1) + u(k)T\cos(\psi(k)) \tag{3.23}$$

$$y(k) = y(k-1) + u(k)T\sin(\psi(k)) \tag{3.24}$$

where T is the sampling time.

3.4.4 Mapping

Mapping is crucial in mobile robot navigation because improve the knowledge about the environment in future localization. Therefore, in the fourth part of the proposed method, the new captured depth image is added into the global map of the environment in the localization computed in Section 3.4.1 and Section 3.4.3. For experimental assessment purposes, a naive approach to map building was exploited in this phase of the work. Thus the addition of the new captured depth image in the map is performed replacing all null pixels existing in the global image by the pixels captured by the Kinect sensor. In this process only the non corrupted data of the captured depth image is considered. With this method, the created global map is immune to the corrupted data and only the rich information about the environment is stored.

3.4.5 Visual odometry with mapping simulation results

Just like the Markov localization and the VO, the application of these localization method in a virtual world with virtual trajectory is a necessity. Since the VOM is a variant of the VO, the simulation test conditions are the same. In both simulation tests, the virtual trajectories and the virtual worlds used are the same. In addition to that, the results are presented in the same way, in the following table (table. 3.6) the attitude estimation error are summarized. In the appendix A.1, it is possible to see the results in detail.

Table 3.6: VOM simulation results

Trajectory	Dome world	IST world
Straight line	$0.008\ \%$	$0\ \%$
Square	$0.254\ \%$	$0\ \%$
Circle	$0.164\ \%$	$0.172\ \%$

When comparing to the results on the table 3.6 , it is possible to conclude that the VOM is as better as the VO, the attitude estimation error are too similar to be able to conclude which method is better. In the appendix A.1.3, it is possible to see that both method are almost equal, however, when the trajectory complexity and length increase, the VOM performance is slightly better than the VO. Since in the experimental tests the trajectory will not be as simple as the Straight line and Square virtual trajectories, the performance for the Circle trajectory is more valuable than the other virtual trajectories. In conclusion, the VOM shows better results for more complex and long trajectory, independently of the virtual world used. The fact that this localization method does not forget the pass and compare the current depth image with a map and not another depth images, proved to be an advantage for the VOM.

3.4.6 Visual odometry with mapping improvements

In the section 3.3, the VO was presented and with the implementation of VOM, some problems were solved. All the new advantages that came from this switch are summarised in the following list:

- There is not need to remove the missing data;
- It is much less incremental than the original VO and it does not forget the pass;
- It is not necessary to have data processing for each and every ceiling depth image;
- Adds mapping to the localization method, which makes the mobile robot even more independent;

4

Experimental Results

Contents

4.1 Introduction

In this chapter it will be presented not only the experimental results, but also it is explained the hardware and the trajectories used in the experimental tests. In section 4.2 it will be explained which hardware was used and details about all the trajectories chosen. From section 4.3 until section 4.4, the experimental results will be shown and discussed.

Before starting an experimental test, it is necessary to go over the following steps:

- Define the experimental test goal;
- Chose a trajectory to achieve that goal;
- Optimize the variables;
- Test with another trajectory ;

The first step is essential for defining which trajectory to use. Depending on the goal, the trajectories used may ranged from a simple short straight line to a longer trajectory with many curves. As for the third step, it is the most time expensive step. Optimizing demand a methodical process to not only make the same test while changing the variables values, but also to understand what is happening and how to change the value in order to achieve the optimal result. Since it is not possible to validate a method if it only works for one specific situation, the final step allow you to support your method and your conclusion. This process it is used with all experimental test done.

4.2 Experimental set up

In section 1.2 it was explained that all experimental tests were done inside a laboratory and all the problems faced were detailed. In this laboratory, the test zone is limited to an area similar to a 19 meters by 9.6 meters rectangle, which is partially occupied by an assembly line and its machinery in the middle. When looking to the available area and the ceiling above it, there is one place that has enough room for complex trajectories and has a ceiling with plenty of information, and there is where most of the trajectories where done.

For this indoor localization method, the ceiling is the most important part of the laboratory. The ceiling is around 5.2 meters high, which is near to the kinect depth camera limit. The reason why the Kinect depth camera is able to acquire the necessary information for the localization method is because the ceiling is not completely flat, it has elements. These elements are air conditioner ducts and filter, lighting bulbs and other objects that usually found in a ceiling.

The mobile robot used for these experimental tests is a low cost mobile robotic platform, with a differential drive configuration. These mobile robot is the most common mobile robot in this laboratory, and for these experimental tests it was necessary to make some modifications to bear the necessary sensors.

In this section it will be presented the hardware(section 4.2.1) and each trajectory(section 4.2.2) used in the experimental tests.

4.2.1 Hardware

The mobile robot is used as a low cost mobile robotic platform [48], with a differential drive configuration. The mobile robot used in all experimental tests is a modification of that mobile robotic platform, which can be seen in the next image (see Fig. 4.1).

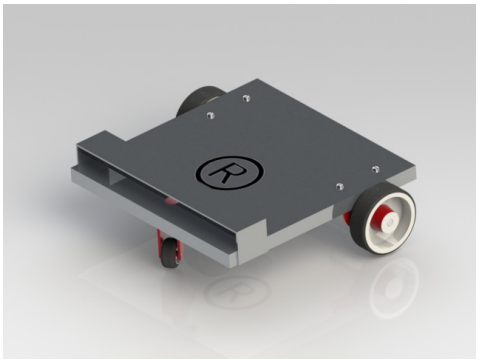

Figure 4.1: The basic mobile platform

This mobile robot has two dc motors that are independent from each other, which greatly increase the mobile robot mobility. The dc motor are the most common electric motor used in mobile robots, due to its price and the rpm it can reach, over 100 rpm. Both these advantages make the dc motor an excellent choice to be used as the motors of a small mobile robot. These motors are controlled by a laptop, placed on the mobile robot, connect to the acquisition card by USB.

Since it was necessary to add sensors, some modifications were necessary, resulting in the following mobile robot(see Fig. 4.2).

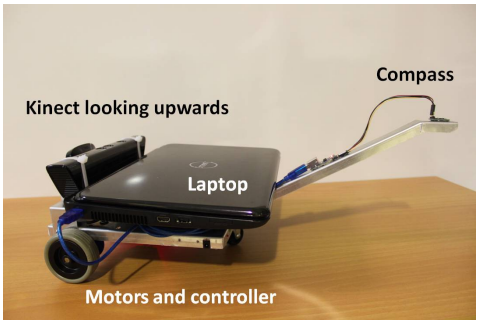

Figure 4.2: Mobile platform equipped with Kinect sensor

On top of the mobile robot there is a PC laptop that receive information from the sensors and also controls the motors. In addition to that, on top of the mobile robot there is also a Microsoft Kinect pointing upwards to the ceiling, which captures ceiling depth images. On the right of the picture, an extension with a digital compass can be seen, to provide alternative attitude measurements. The digital compass, also know as magnetometer, is an instrument for measuring the strength and sometimes the direction of a magnetic field. In this case, the digital compass a magnetic device sensitive to an external magnetic field, instead of a permanent magnet or an electromagnet. By defining the four cardinal points before starting any experimental test, the digital compass is calibrated to the existing magnetic, determining the mobile robot rotation during the experimental test.

Overall these mobile robot has 3 different sensors, being the wheel encoders and the digital compass, as the secondary sensors, and the Kinect depth camera as the main sensor, which is detailed in the next paragraph.

Kinect depth camera:

Recently, Microsoft Kinect was released and researcher were exposed to new cheap sensor, the Kinect Depth camera. The Microsoft Kinect was design mostly for entertaining, however researcher see it as a cheap bundle of sensors. This bundle of sensors includes a RGB camera with a VGA resolution (640×480 pixels) using 8 bits and a 2D depth sensor (640×480 pixels) with 11 bits of resolution. In addition to that, it also has a multi-array microphone and a motorized tilt in the base, which are not used in any experimental tests. Since the Kinect has both RGB and Depth camera, it is possible to merge the information captured by both cameras and create a RGB-D image. However, this Kinect feature is not used in any experimental tests.

All experiences are performed under a ceiling height of $5.2\ m$ resulting in a depth images with resolution $7.8 \times 10^{-}3\ m/pixel$. Notice that mobile robot motion is aligned with vertical axis of Kinect sensor, which has 480 captured pixels and a vision angle of 43°

4.2.2 Trajectories

Along with the three trajectories used in the simulation (section 3.2.3), which are virtual trajectories, there are two real trajectories that were used to test the localization method. These trajectories are completely different, the Longer trajectory which is a trajectory that focus on long straight lines, whereas the Lawn-mower trajectory is small but focus more on the curves. In this section, each real trajectory is detailed, showing their strength and weakness.

Lawn-mower trajectory:

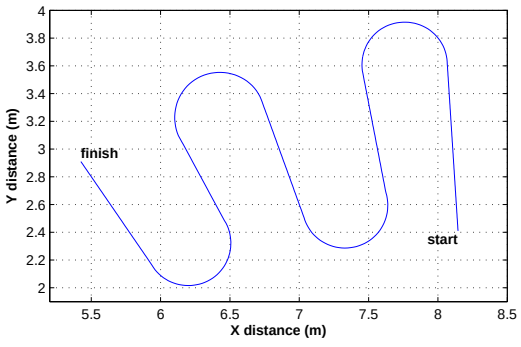

Figure 4.3: Lawn-Mower trajectory map

The Lawn-mower is a classical trajectory, which combines lines with curves, alternating the turning direction of the mobile robot. This trajectory is 8.1 meters long and takes 81 seconds(1 minutes and

21 seconds), or 406 samples, to complete. The great combination of straight lines and curves make this trajectory perfect for an overall test. As it is possible to see in the figure 4.3, there are five straight line and four $90°$ turns, which alternate from left to right. This trajectory shape is perfect for a small test zone, since it uses most of the available space, without crossing its own path more than once. Therefore, this is probably the most common trajectory to test a mobile robot localization method after a straight line(1-D problem).

Longer trajectory:

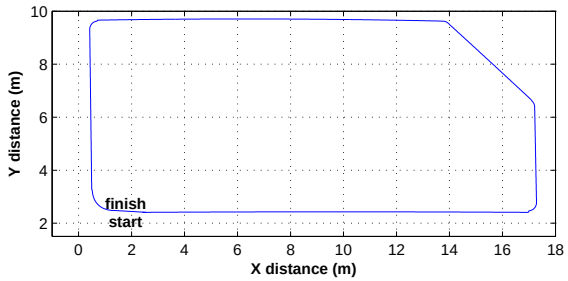

Figure 4.4: Longer trajectory map

In order to test the indoor localization method in a series of long straight lines, it was necessary to create a trajectory that would go around the assembly line, that is in the middle of the testing zone, doing a complete lap near the edge of the testing zone. This is considered the final test for the localization method, due to the high difficulty. This trajectory forces the mobile robot to pass through zones without many information and through zones with similar ceiling elements, which is a great challenge for the localization method. In the figure 4.4 it is possible to see the complete lap around the laboratory, however, for the experimental tests, it was only possible to use the information until the end of the second big straight line. That trajectory is 32 meters long and takes 320 seconds(5 minutes and 20 seconds), or 1600 samples to complete.

4.3 RGB vs Depth camera experimental results

In order to be able to tell which type of camera is better, it was necessary to put both in similar situations, changing some variables and analysing the results. Since the Microsoft Kinect have both type of sensor, the conclusions were more accurate.

During the tests, the Kinect uses both RGB and depth cameras at the same time, making it possible to compare both cameras on the same trajectory. However, just one test is not enough, so some variables are changed between experiences. One of those variable is the environment light condition, in other words, if the laboratory lights are turned on or off. In section. 1.2.1 it was expose how each type of camera perform under these situations. From here, ”lights on” correspond to the runs

done with the laboratory artificial lights on, as for the ”lights off” correspond to the runs done with the laboratory artificial lights off.

In section 2.6, the PCA method was detailed and for this test the implementation is the same as Carreira et al. approach in [7]. Since it is possible to change the environment light condition between the runs, it is also possible to build two different PCA database. When the ceiling depth images taken with lights on are used to create the PCA database it has the name ”PCA1”. On the contrary, when the ceiling depth images taken with lights off are used to create the PCA database it has the name ”PCA2”.

As was explain in previously, there are a total of 4 different situations that can be tested for each type of camera. Since the point of this paper is not to test the localization method, the trajectory does not need to be complex or long. As a result, the chosen trajectory is a simple 3 meters straight line, which was repeated 10 times. The first run was used to compute a certain PCA and the others 9 were used to test both PCA. Testing all the possibles situation allow us to test the robustness of both camera and their performance.

Instead of showing all the results, only the best and more significant result are shown and analysed. The different combinations of conditions were separated into 2 different groups, matching condition(section 4.3.1) and non matching condition(section 4.3.2). The first one contain the results where the run light conditions are the same as the images used to create the PCA, both with light on or both with lights off. On the other hand, the non-matching condition have the result where the run light conditions are not the same as the images used to create the PCA, when one uses images with light on, the other uses images with light off.

4.3.1 Matching conditions results

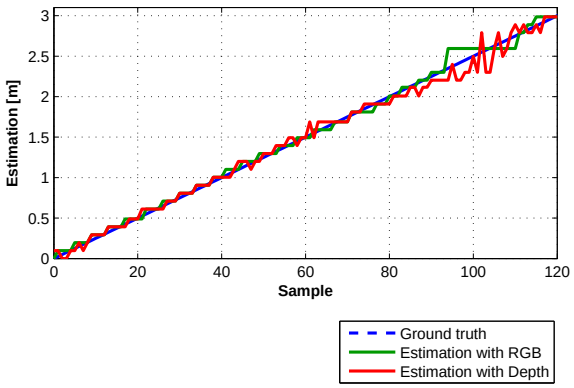

Figure 4.5: Position estimation using PCA 1 and lights on

With matching conditions, both RGB and Depth camera have similar results, with a small advantage for the RGB camera. Both have a very good start, but it is in the end of the run that the RGB camera tops the Depth camera. The reason why the depth camera have a slightly worse result in the

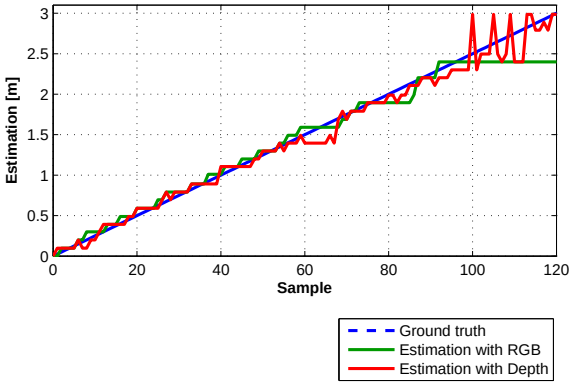

Figure 4.6: Position estimation using PCA 2 and lights off

end is because there is almost no ceiling information in that zone. Since the RGB can use colours as information, it is not as affected as the Depth camera.

There is a small difference between the estimated position in both figures(Fig.4.5 and Fig.4.6), however it is nothing significant. When the lights were turn off, the results between runs, using RGB camera, were not as regular as when the lights were turn on. The reason is because when the lights are turn off, the environment natural light factor gains weight and can change completely the performance of the RGB camera. This conclusion allow us to predict that in non matching conditions the RGB camera results wouldn't be as good as the Depth camera results.

4.3.2 Non matching conditions results

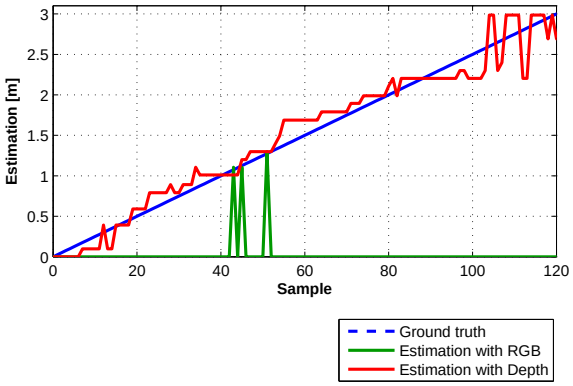

Figure 4.7: Position estimation using PCA 1 and lights off

With both figures (Fig. 4.7 and Fig. 4.8) it is possible to conclude that using different light condition is indeed prejudicial for the RGB camera performance. With this conditions, the localization method is completely lost when using the information captured by the RGB camera. This can be seen in both figures where the position estimated it is the same value, or around the same value, during all the

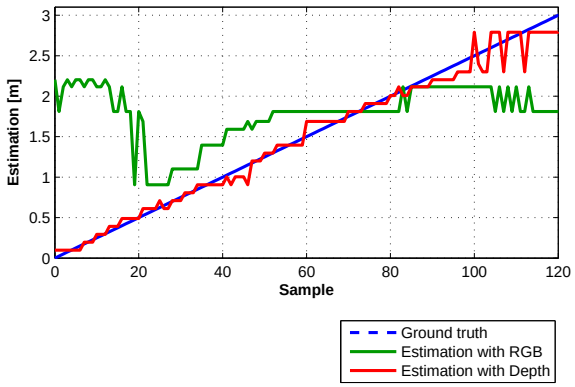

Figure 4.8: Position estimation using PCA 2 and lights on

run. In addition to that, the position estimation using the depth camera is slight worse comparing to the results with matching conditions. Just like in the others cases, the position estimation achieved its worse result near the end of the run. Overall, it is possible to conclude that the depth camera performance is independent of the light conditions.

In the first figure (Fig. 4.7), the localization method always returns the start position, with exception of only 3 moments. As for the second figure (Fig. 4.8), the position estimation is always around the same value, 2 meters. Both results show that the RGB camera performance is completely dependent of the environment light conditions, as expected.

4.4 Visual odometry experimental results

To test all the visual odometry approaches, VO, visual odometry using features (FVO) and VOM, several tests were performed with different trajectories. In every experimental tests, the mobile robot starts with a velocity of $0.1\ \mathrm{m} \cdot \mathrm{s}^{-1}$, maintaining it during all trajectory. During the motion at constant speed along the predefined trajectory, the mobile robot captures depth images from the ceiling with $5\ \mathrm{Hz}$ of sampling rate and several points are marked on the ground to be measured after the finish of the experience.

Except in the FVO, the attitude computation (green filled circle in Fig. 4.9) have been performed with two iterations. The first is a broad search for the correct attitude, therefore it has a big range, $\Delta\psi = 180°$, and a big step, $10°$. As for the second, the attitude found in the first iteration is refined with a smaller range, $\Delta\psi = 5°$, and a smaller step, $1°$. The velocity range is $\Delta u = 0.12\ m \cdot s^{-1}$, with a step of $0.04\ m \cdot s^{-1}$. In the next figure (Fig. 4.9), the red circles correspond to a attitude, or velocity, value tested, where the green filled is the best value.

The FVO implementation used in this situation is one example of the Matlab Computer Vision Toolbox as it is, with the necessary modifications. The FVO method starts by finding SURF features, which are matched using the NNR method. Afterwards, RANSAC is used to estimate geometric transform. Then, after all these steps, it is possible to estimate the mobile robot attitude and velocity.

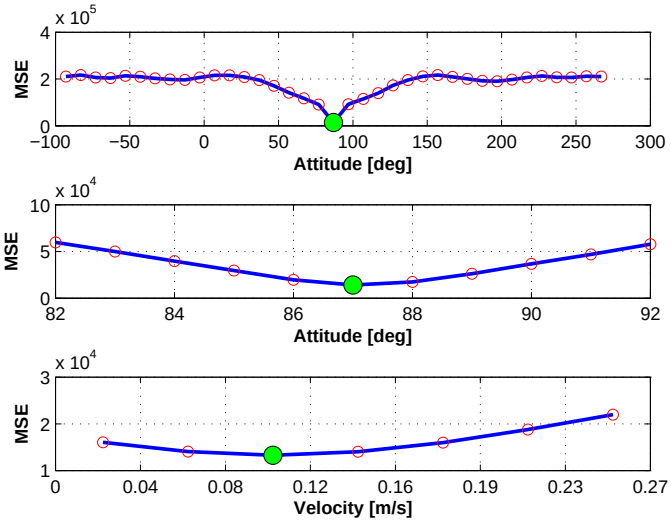

Figure 4.9: Attitude and velocity computation in the instant $t = 2\ s$

4.4.1 Results for Lawn-mower trajectory

The first experimental test uses the classical Lawn-mower trajectory (more details in section 4.2.2). This is the perfect trajectory to start the experimental tests, since it is small, but complex enough to test the proposed methods in many ways.

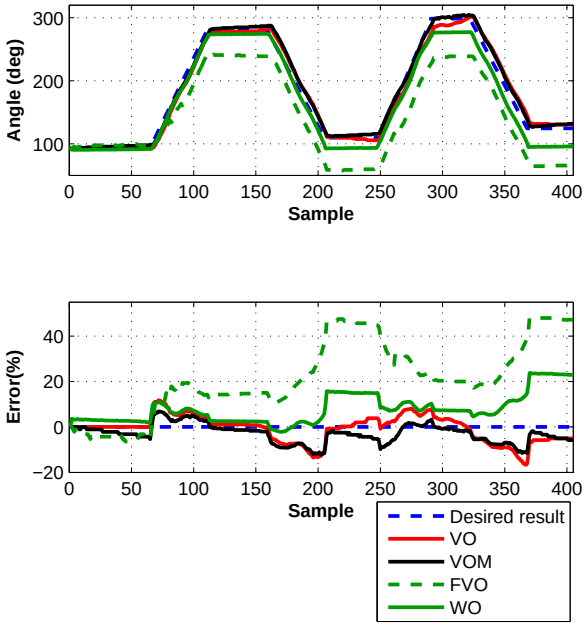

Figure 4.10: Estimated robot's attitude(top) and its error(bottom) along time of all odometry methods

As it is possible to see in figure 4.10, every visual odometry approach proposed is superior to WO, except the FVO. Starting with the method with the worse result, the FVO. The results obtained show that this method is unable to correctly estimate the mobile robot attitude even for the most simple

and short trajectory tested. However, the reason why the attitude estimation is so far from the mobile robot correct attitude, is the poor attitude estimation during the first turn. After that point, the FVO estimation has a similar behaviour to the others visual odometry approaches (Fig. 4.11), despite the increasing error (Fig. 4.10).

As for the VO method, it is much better than the FVO, however it does not outperforms other methods. Until sample 160, the VO attitude estimation error is very similar to the WO attitude estimation error. On the contrary, after that sample, the VO attitude estimation is much better than the WO estimation, ending with only 6% error against 28% error(Fig. 4.10). In the estimated attitude figure (Fig. 4.10) it is possible to confirm that the WO attitude estimation gets worse each turn, being significantly different than the ground truth attitude after the second turn. The figure 4.11 allows to easily see difference between the two methods that seemed to be similar, confirming that the VO is indeed better than the WO.

With a similar result to the VO, the VOM proved to be one of the best approaches proposed. In the attitude estimation figure (Fig. 4.10) it is not possible to clearly see that the VOM has a better performance than the VO. On the other hand, in the attitude error figure, it is possible to see that the VOM attitude estimation error is more closer to the desired error than the VO in many samples. This can be explained by the fact that, whereas the VOM is able to correctly estimate the mobile robot attitude during the turns, while the VO is forced to use the straight line to correct the wrong attitude estimation. In figure 4.11, the small difference between these two method is more evident. The VOM is much closer to the ground truth path during the whole trajectory, unlike the VO.

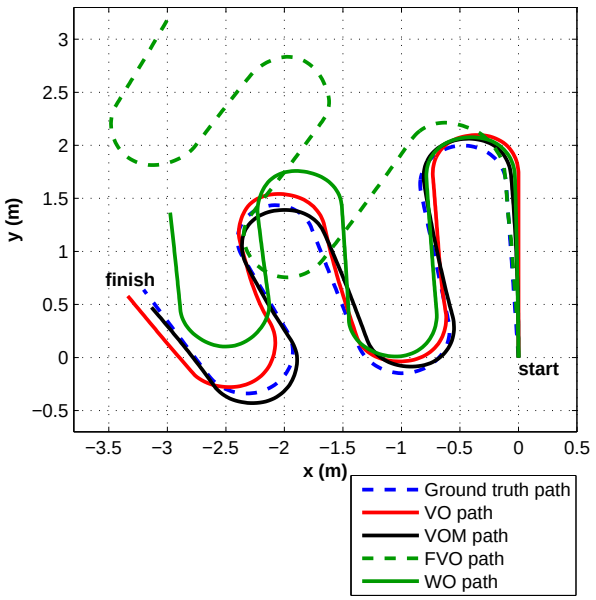

Figure 4.11: Map of all odometry methods estimated position, considering a ground truth path

In addition to estimate the attitude, the VOM is also able to build a ceiling map during the experimental test. Unlike VO, the VOM does not need uncorrupted ceiling depth images, in other words

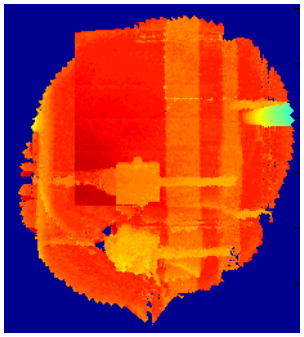

Figure 4.12: Ceiling map built along the trajectory

without missing data, since during the mapping the missing data is automatically reduce to a minimum. In the following figure (Fig. 4.12), the final ceiling map built by the VOM is presented to show that the missing data is almost completely removed. Notice that, although the missing data is represented by the blue colour, the blue area outside the ceiling map it is not missing data.

With the classical Lawn-mower trajectory it is possible to conclude that the FVO underperforms the other visual odometry approaches. However, for the VO and VOM, it is not possible to say which one is superior.

4.4.2 Results for Longer trajectory

As it was said before (section 4.2.2), this trajectory has the longest straight line of all trajectories used, as well as it is the biggest trajectory, taking more time to complete than any trajectory.

Whereas the Lawn-mower trajectory is not long enough to see the differences between the approaches proposed, the Longer trajectory takes it a step forward, since it is $32\ m$ long. In this trajectory, it is expected for those differences to be more significant, exposing this way the strengths and weakness of each method. Due to the duration of this trajectory and to the size of the straight lines, the overall performance of each method it is expected to be lower than in the Lawn-mower trajectory.

Analysing the results presented in the estimated attitude figure (Fig. 4.13), it is possible to see that the attitude estimated by the FVO method it is definitely the method with the poorest performance. Since the start, the FVO it is not able to correctly estimate the mobile robot attitude, presenting the highest error (Fig. 4.13). However, even with these results, the final position estimated is very close to the mobile robot correct position, as it can been seen in figure 4.14. In the same figure it is also possible to see that unlike the others trajectories, the FVO path is the least similar comparing to the ground truth path.

With a longer trajectory, the difference between VO and WO are more significant, which can be seen in figure 4.13. In this case, the VO shows results slightly closer to the ground truth attitude than the computed by WO especially during the last straight line. When analysing the evolution of the attitude estimated by the VO in figure 4.13, it is possible to see that the major problem is during the curves, where the VO has a worse estimation than the WO and it is not able to reach the correct attitude. However, since this trajectory has two extremely long straight lines, the VO is

able to compensate the attitude estimation during those straight lines, ending much closer to the real trajectory than the WO. In figure 4.14, it is easily visible that the VO is not perfect, the estimation during the last straight line is the reason why it ends so far from the ground truth path final position. Nevertheless, it still ends closer than the WO.

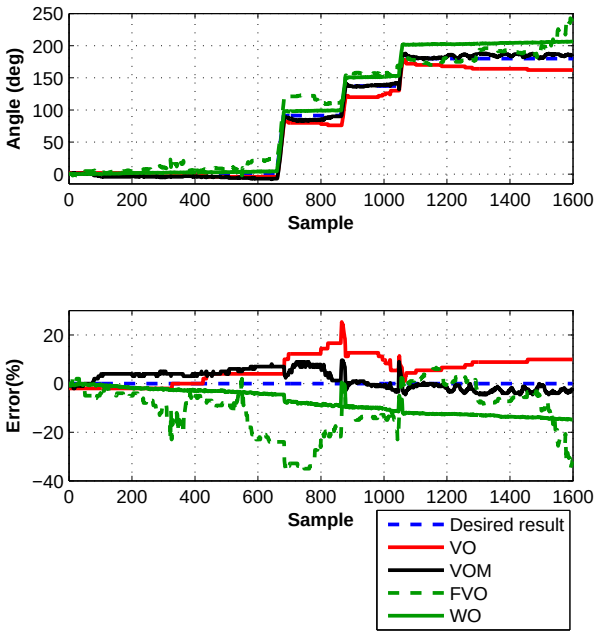

Figure 4.13: Estimated robot's attitude(top) and its error(bottom) along time of all odometry methods

This trajectory not also confirmed conclusion made above (section 4.4.1), but also made it much easier to decide if VOM is superior to VO or not. Except in the first straight line, the VOM is completely superior to the VO. In figure 4.13, the attitude estimated by the VOM is very close to the mobile robot real attitude, unlike the VO that gets worse after each turn. In the attitude estimation error figure (Fig. 4.13), it is possible to see how the VO attitude error gets incrementally bigger, ending with almost 20%. Despite starting with an hight error, the VOM is able to recover and end with a very small error, ranging from 0% to 8% during the last straight line. Despite the excellent results presented by the VOM, the figure 4.14 shows that the VOM is not as close to the mobile robot correct path as desired. Notice that, in this experience the robot is moving along $32\ m$, navigating $320\ s$ (5 minutes and 20 seconds) with only one sensor in an unknown environment. This allow to conclude that, to develop localization systems able to navigate in an unknown environment, the VOM method must be changed or fused with other sensors. However, these results shows that the use of VOM can provide better motion prediction than the original method, the WO.

In addition, this result confirms the hypothesis that the VOM is less influenced by the ceiling zones with low information above that last straight line path than the VO. Whereas the VO only uses the last ceiling depth image to compare with the current ceiling depth image to find the mobile robot attitude, the VOM compares the current ceiling depth image with a ceiling map (Fig. 4.15). When comparing the current ceiling depth image with this ceiling map, the VOM has more information to correctly

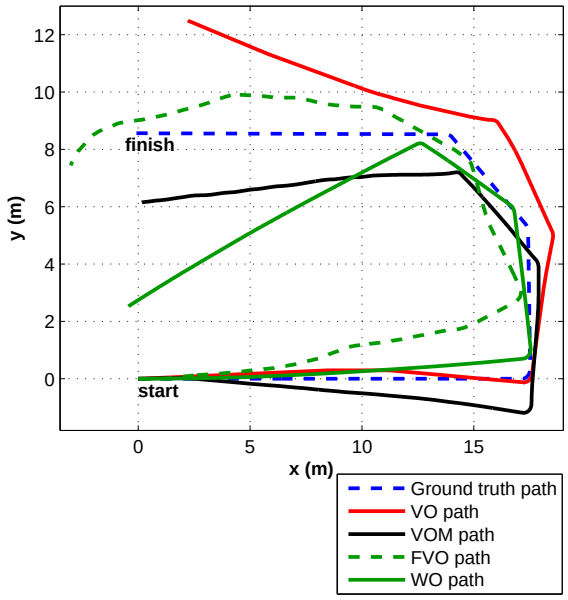

Figure 4.14: Map of all odometry methods estimated position, considering a ground truth path

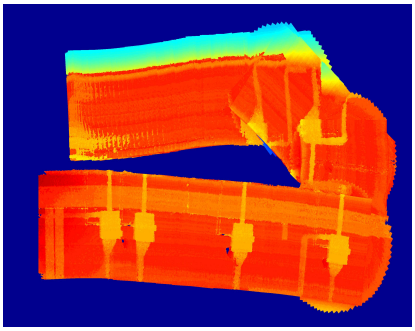

Figure 4.15: Built map along a trajectory with few image overlapping

estimate the attitude than the VO. Just like in the case of the Lawn-mower trajectory (Fig. 4.12), the VOM is also able to remove the missing data with success, ending with a ceiling map very similar to the real ceiling [9].

4.5 Markov localization experimental results

The goal of these experimental tests is to first compare the Markov Localization using WO (MLWO) with the PCA localization method, then to compare the Markov localization using VOM (MLVOM) with the MLWO. In order to test these localization methods, two different trajectory were chosen. The first being a classic Lawn-mower and the other being the Long trajectory, for detailed information see section 4.2.2.

In every experimental tests, the robot starts at $x_0 = 0$ m, $y_0 = 0$ m, $u_0 = \dot{y}_0 = 0.1$ m $\cdot$ s^{-1}, $\dot{x}_0 = 0$ m $\cdot$ s^{-1}. During the run, the robot speed is constant and it captures depth images from the ceiling with 5 Hz of sampling rate. In order to find the robot real path, several points were marked on

the floor during the run and measured afterwards.

Instead of just using two different trajectories to test every localization methods, it is possible to change a component in every methods, the world map. As it was explain in section 3.2, the world map is an image which the current image is compared to. Being this also true to the PCA method, the change of world map will influence the performance of both localization methods. Therefore, two different world maps with different size are used, one that is big enough to contain all the trajectory and one which is an image of the whole laboratory ceiling, the complete world map. With this world map, there is the problem of certain ceiling zones having similar values, due to repeatability of ceiling objects, which may have a negative influence on the final result. Both worlds maps are showed in the next figures Fig. 4.16 and Fig. 4.17)

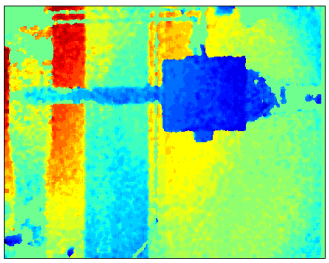

Figure 4.16: Small world map for Lawn-mower trajectory

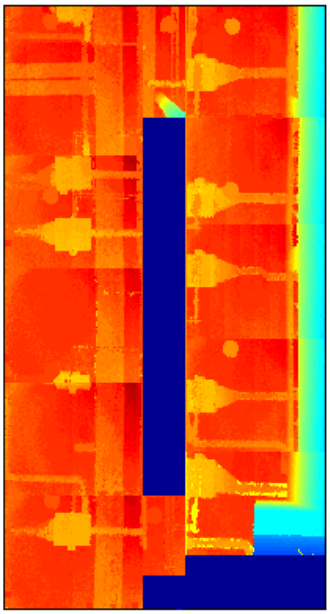

Figure 4.17: Complete world map for any trajectory

Whereas the first world map showed is one single ceiling depth image, for the second world map it was necessary to merge multiple ceiling depth images in order to create the whole laboratory world map. Since this thesis does not focus on this area, this process was done by stitching all the images

by hand. This method resulted in a not so perfect world map, and by using this map, the Markov localization performance was negatively influenced by all imperfections of the world map.

In conclusion, every localization methods are tested by one trajectories with two different world maps and another trajectory with one world map, which means every localization methods are tested in three different situations. For every experimental test it will be presented not only the both position estimation, position x and position y, of both method comparing to the Ground truth, but also the evolution of both Markov localization probabilistic map. Due to the trajectories duration, it is not possible to show every sample of the both Markov localization probabilistic map. Therefore, it will be presented only few samples in order to illustrate the evolution of both Markov localization probabilistic map. In these images, it is also presented the mobile robot correct position as a magenta "$*$", which leaves behind a line with the same colour, that correspond to the part of trajectory already made. With the evolution of the probabilistic map, it is possible to see the how the Markov localization works and how it finds the mobile robot correct position taking in account all the possibilities.

The following subsections are divided first by type of trajectory and then by type of world map. In this subsections first is presented the position estimation graphic then the evolution of the probabilistic world.

4.5.1 Results for Lawn-Mower trajectory

Since this trajectory spends more time moving in the yy axis than in the xx axis, it is expected for the MLWO and MLVOM position estimation to be more precise in the y position than in the x position. On the contrary, the PCA is not as influenced by the robot motion as the Markov localization, therefore the position estimation using the PCA method should be more consistent. As for the difference between both Markov localization methods, since this is the smallest trajectory, only 406 samples, it is not expected to see a huge improvement.

Small world map:

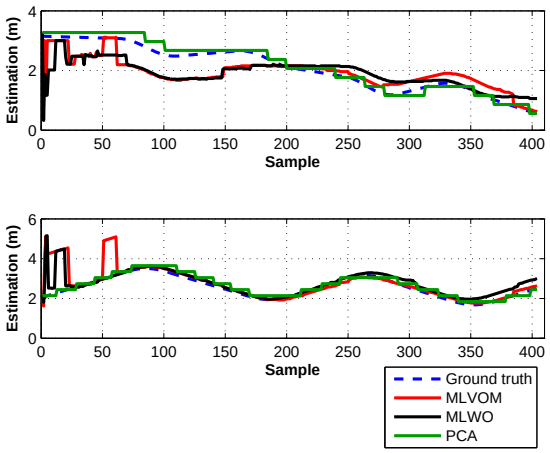

Figure 4.18: x position (top) and y position (bottom) estimation on Lawn-Mower trajectory

In the position x estimation figure (Fig. 4.18) it is possible to see that the PCA method has a much better result than both Markov localization methods. On the other hand, every methods position y estimation are very close to the ground truth. As for the both Markov localization methods, the results are very similar, except in the end of the trajectory. With just the information of this experimental test, it is not possible to conclude that is beneficent to change the building block.

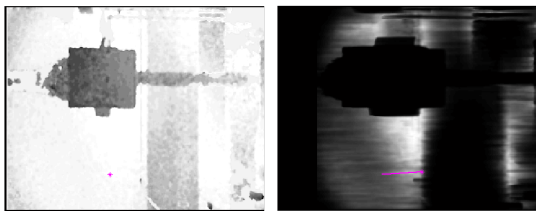

Figure 4.19: MLWO probabilistic map in sample = 1 (left) and sample = 42 (right)

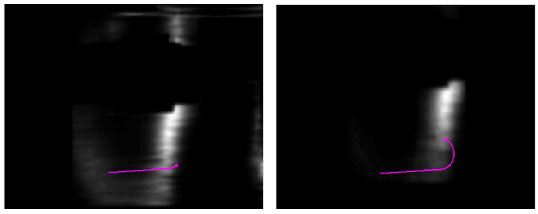

Figure 4.20: MLWO probabilistic map in sample = 72 (left) and sample = 102 (right)

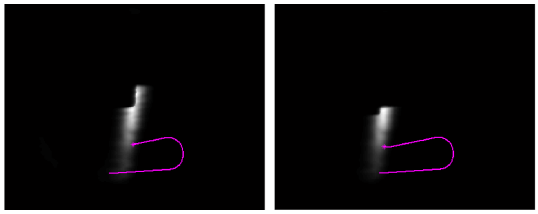

Figure 4.21: MLWO probabilistic map in sample = 147 (left) and sample = 167 (right)

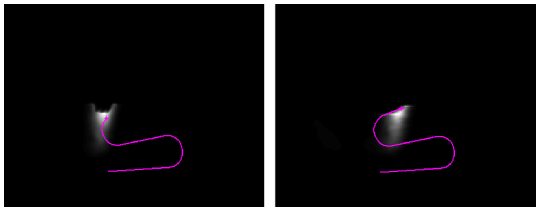

Figure 4.22: MLWO probabilistic map in sample = 197 (left) and sample = 222 (right)

Following the evolution of the probabilistic map, it is possible to see that in the first samples the method quickly converge to the correct position(Fig. 4.19). However, the method is not able to reduce the possible zone where the mobile robot might be until the mobile robot passes under a zone with high information(Fig. 4.21). With one more passage under that zone, the MLWO is able to pinpoint the mobile robot real position (Fig. 4.22). Due to the negative influence of the WO estimation, the MLWO will end with a wrong estimation, as it is possible to see in figure 4.18.

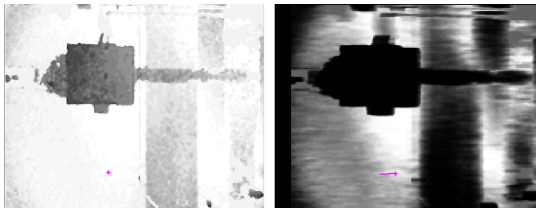

Figure 4.23: MLVOM probabilistic map in sample = 1 (left) and sample = 17 (right)

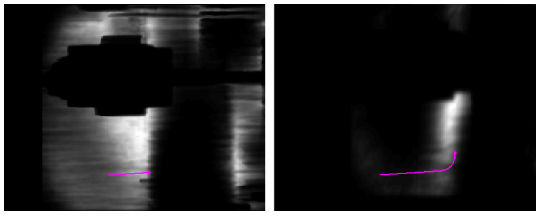

Figure 4.24: MLVOM probabilistic map in sample = 42 (left) and sample = 87 (right)

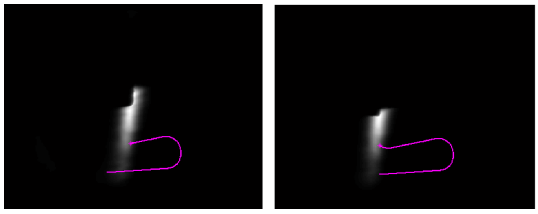

Figure 4.25: MLVOM probabilistic map in sample = 147 (left) and sample = 172 (right)

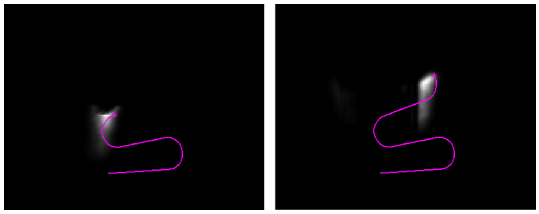

Figure 4.26: MLVOM probabilistic map in sample = 207 (left) and sample = 272 (right)

The MLVOM probabilistic map evolution is almost the same as the MLWO probabilistic map evolution. It quickly converge to the correct position(Fig. 4.24), but it is not capable of pinpointing the mobile correct position until it reaches a zone with high information, just like the MLWO. After passing the zone with high information, the MLVOM pinpoint the correct position and ends with a better estimation due to a better building block(Fig. 4.26).

Analysing the both Markov localization methods position estimation results, it is possible to conclude that the y position estimation is better than the x position, as it was expected. This happen because this particular trajectory starts in a zone with low information, and until it passes under a zone with high information, both Markov localization methods can not pinpoint the mobile robot position. Afterwards, both Markov localization methods is able to keep up with the correct results. In the trajectory end, the MLWO is mislead by the WO position estimation, ending slightly far from the mobile robot correct position, whereas the MLVOM is able to end very close to the mobile robot correct position.

Complete world map:

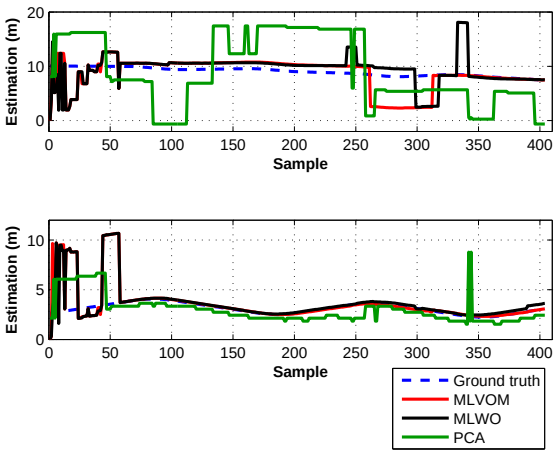

Figure 4.27: x position (top) and y position (bottom) estimation on Lawn-Mower trajectory

When the complete world map is used, the Markov localization performance is visible higher than the PCA performance. Even though the PCA position estimation is very good for y (Fig. 4.27), the x position estimation is always estimating the wrong position (Fig. 4.27). When looking to the laboratory it is possible to see that the PCA has this results due to the repeatability of ceiling elements, has it was expected.

On the other hand, the Markov localization result (Fig. 4.27 and Fig. 4.27) are exceptional, proving that it can overcome the repeatability of ceiling elements.

In the first figure (Fig. 4.27), the MLWO estimate the correct position during more samples than the MLVOM. However, both methods ends very near the mobile robot correct position. In addition to that, both method show a very similar result in figure 4.27. In conclusion, both MLVOM and MLWO are very similar with short trajectories.

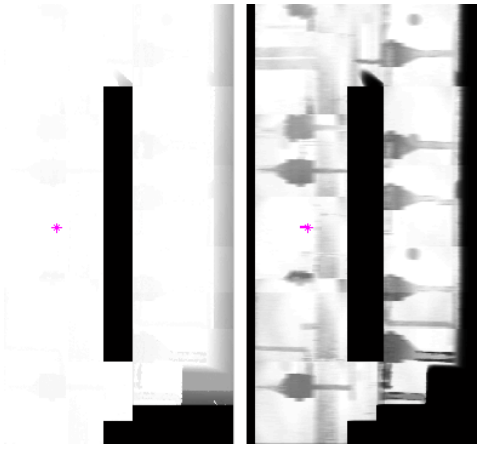

Figure 4.28: MLWO probabilistic map in sample = 1 (left) and sample = 22 (right)

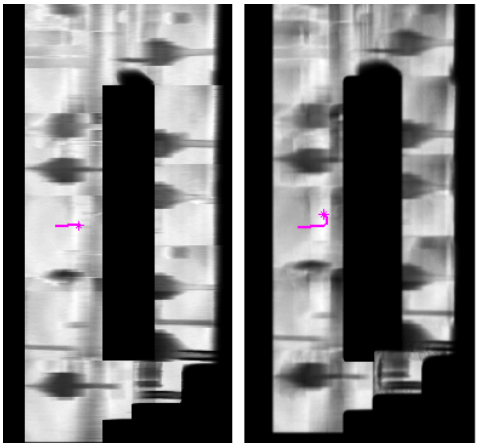

Figure 4.29: MLWO probabilistic map in sample = 62 (left) and sample = 102 (right)

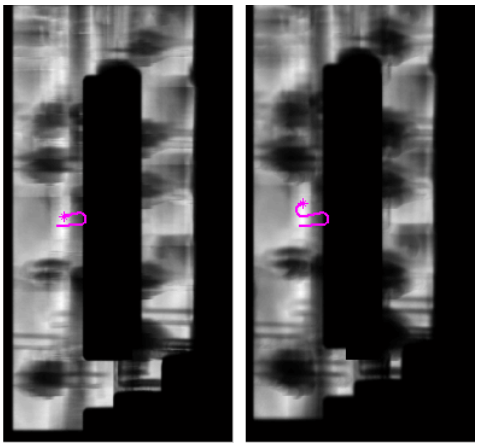

Figure 4.30: MLWO probabilistic map in sample = 152 (left) and sample = 212 (right)

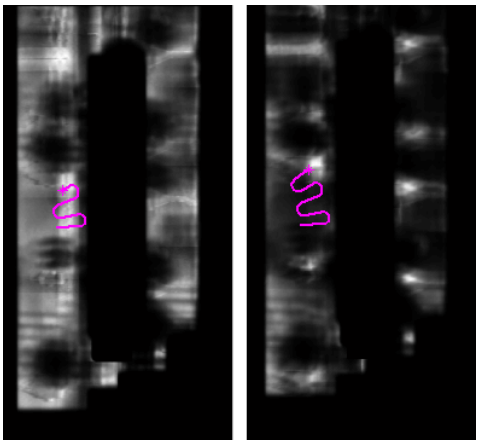

Figure 4.31: MLWO probabilistic map in sample = 307 (left) and sample = 402 (right)

Due to the world map used, the complete map, the MLWO needs more time to converge to the correct position and to eliminate other wrong possible positions. At figure 4.29 it is possible to see that there is a focus near the mobile robot real position, however it is not possible to tell which is the mobile robot correct position without looking for the ground truth marker. During some moments, the MLWO does not significantly converge to the correct position (Fig. 4.30). As it is possible to see in

figure , there is a moment where the MLWO estimate the wrong position, which is showed in figure 4.31. In the same figure is possible to see that it recovers from the wrong estimation. In addition to that, if the trajectory was longer, the MLWO would single out the correct position, eliminating the white zone in the bottom.

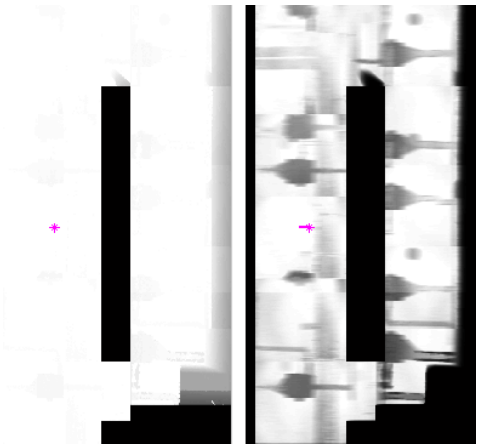

Figure 4.32: MLVOM probabilistic map in sample = 1 (left) and sample = 27 (right)

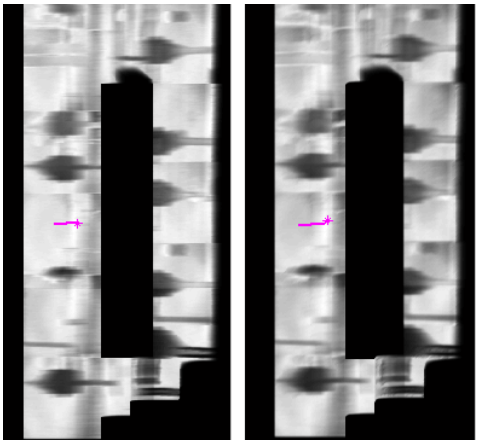

Figure 4.33: MLVOM probabilistic map in sample = 62 (left) and sample = 82 (right)

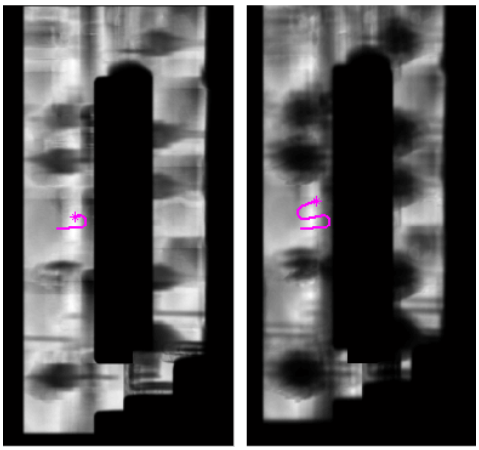

Figure 4.34: MLVOM probabilistic map in sample = 122 (left) and sample = 242 (right)

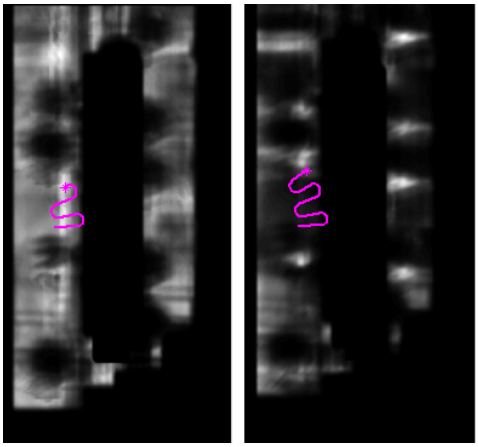

Figure 4.35: MLVOM probabilistic map in sample = 297 (left) and sample = 402 (right)

Just like the MLWO, the MLVOM needs more time to converge to the correct position due to the world map size. Whereas in sample 42 the MLVOM, with small world map, already converged to the correct position, with this world map it is only possible to see a focus near the mobile robot real position in 82 sample (Fig. 4.33). Furthermore, until sample 242 it is not possible to find the mobile correct position without the help of the ground truth marker (Fig. 4.34). When looking to the position estimation graphics, it is expected to see that there was only one point that had high probability during the last part of the trajectory, which is not true. During that part, the MLVOM maximum value even changed from the mobile correct position to another position, but it ended in the correct position (Fig. 4.35).

4.5.2 Results for Longer trajectory

For being the longest trajectory used in all experimental tests, this trajectory is the most difficult for the localization methods. During the trajectory, the mobile robot passes under similar zones more than once, testing the localization method ability to overcome this adversity. Therefore, is expected for the PCA to have more difficulties to estimate the mobile robot correct position, since it is not as good as the Markov localization in overcome the ceiling element repeatability problem. In this trajectory, it is expected to notice the difference between WO and VOM. Taking in account the result showed in figure 4.14, the result obtained by MLVOM should be significantly higher than the MLWO. The influence of wheel slippage depend on the trajectory length and its complexity, and in a long trajectory with many straight lines, the WO position estimate is terrible.

As expected, the PCA estimation is much more chaotic and worst than both Markov localization methods estimation, which is more visible in the position x estimation figure (Fig. 4.36). Surprisingly, the PCA position y estimation is better than the MLWO estimation, especially near the end. However, with a awful position x estimation, it is possible to conclude that the PCA is not the best localization method to estimate the mobile robot correct position with a long trajectory in a large world map. Unlike the others two experimental test, this trajectory is long enough to notice the difference between WO and VOM. In the position x estimation figure (Fig. 4.36) there are no significant difference between both localization methods. On the other hand, in the position y estimation figure (Fig. 4.36) confirms

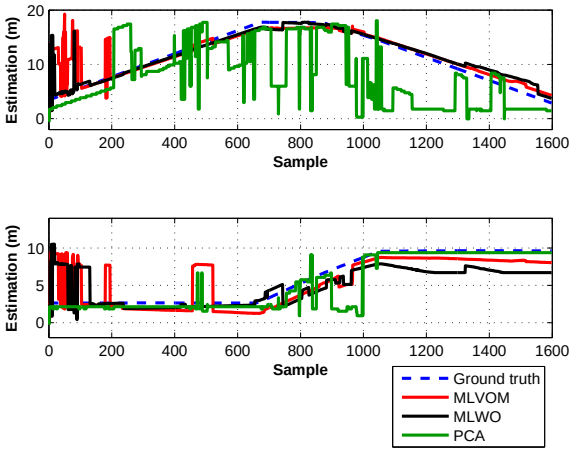

Figure 4.36: x position (top) and y position (bottom) estimation on Longer trajectory

that using VOM instead of WO improves significantly the Markov localization performance. On the second part of the trajectory is visible the WO bad influence, making the Markov localization estimate a wrong position.

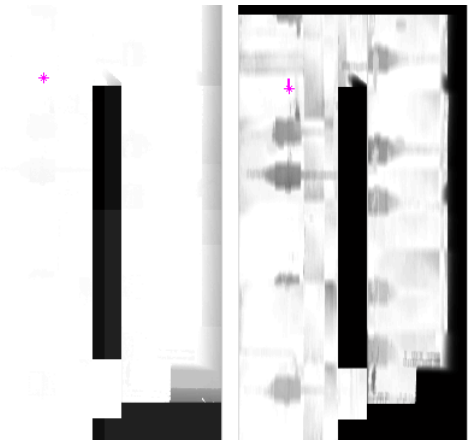

Figure 4.37: MLWO probabilistic map in sample = 1 (left) and sample = 22 (right)

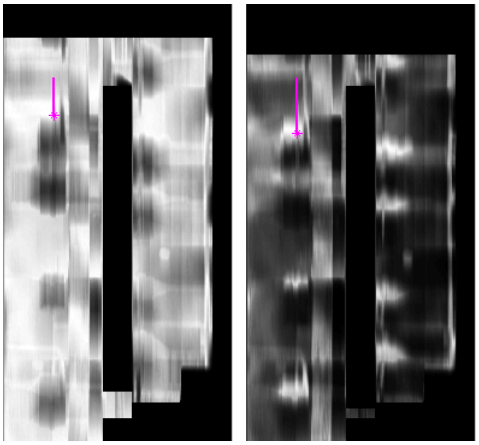

Figure 4.38: MLWO probabilistic map in sample = 82 (left) and sample = 122 (right)

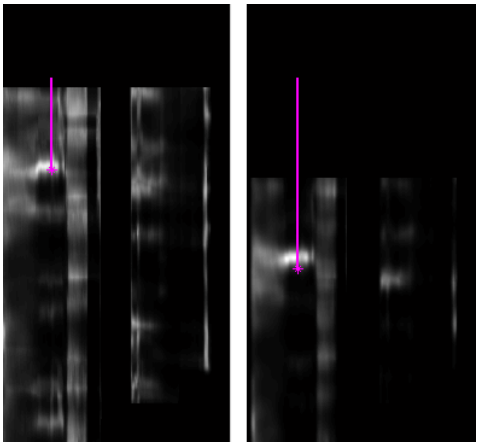

Figure 4.39: MLWO probabilistic map in sample = 202 (left) and sample = 422 (right)

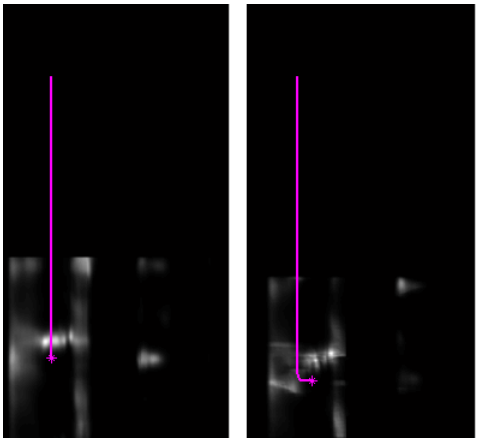

Figure 4.40: MLWO probabilistic map in sample = 622 (left) and sample = 702 (right)

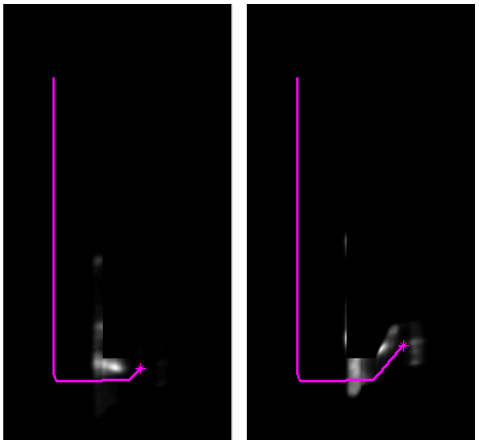

Figure 4.41: MLWO probabilistic map in sample = 902 (left) and sample = 962 (right)

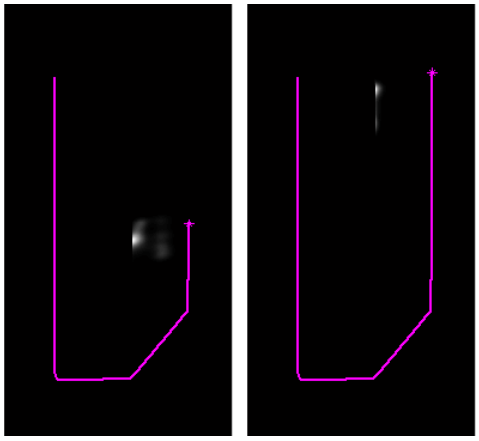

Figure 4.42: MLWO probabilistic map in sample = 1242 (left) and sample = 1582 (right)

As previously stated, this trajectory is the longest trajectory tested and, therefore, it took the most time to start converging to the correct position, 122 samples (Fig. 4.38). However, after pinpointing the mobile robot correct position at sample 202 (4.39), it is able to correctly estimate the mobile robot position until the first turn, with a little delay. During the first turn, the little delay makes the MLWO estimation to jump from one side of the laboratory to another (Fig. 4.41). With this jump, the MLWO position y estimation will never catch up with the mobile correct position, always being near the edge of the map. This is explained by the WO estimation, that is constantly pulling the MLWO estimation to the left after the ”U” turn.

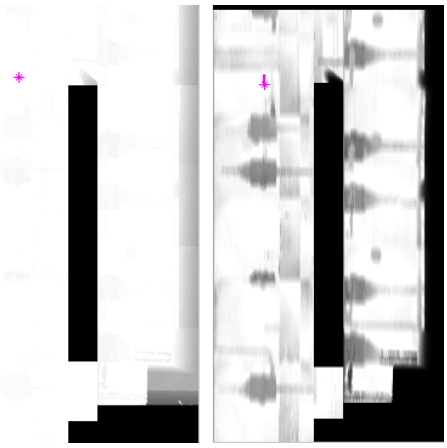

Figure 4.43: MLVOM probabilistic map in sample = 2 (left) and sample = 22 (right)

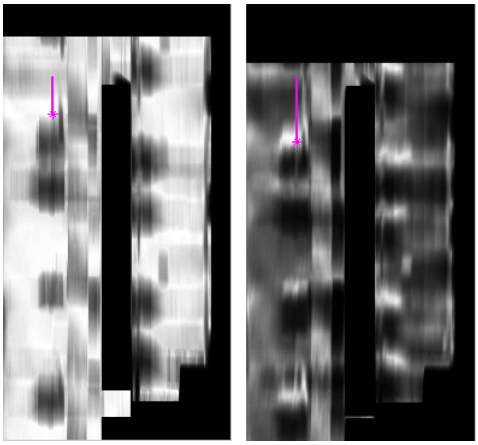

Figure 4.44: MLVOM probabilistic map in sample = 82 (left) and sample = 142 (right)

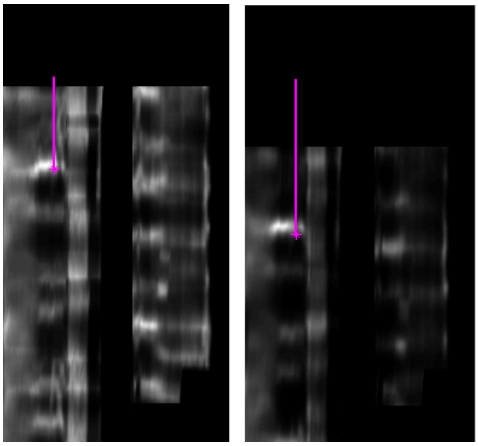

Figure 4.45: MLVOM probabilistic map in sample = 202 (left) and sample = 342 (right)

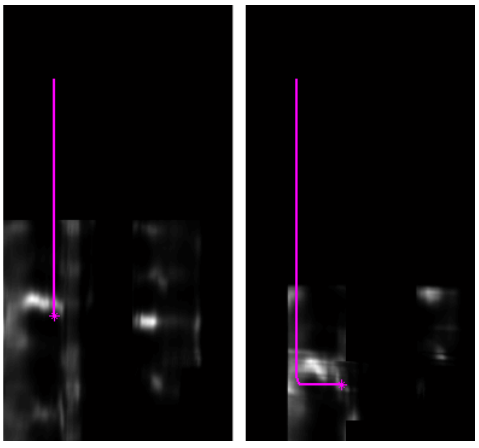

Figure 4.46: MLVOM probabilistic map in sample = 522 (left) and sample = 782 (right)

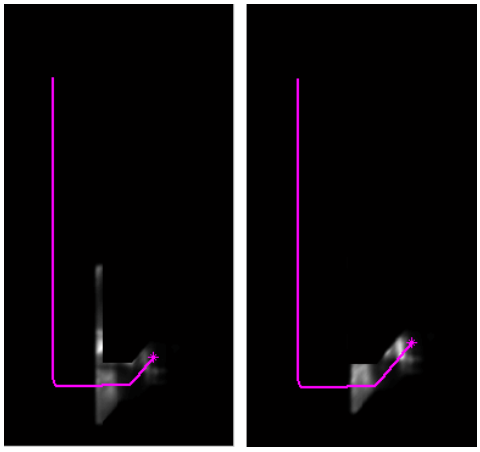

Figure 4.47: MLVOM probabilistic map in sample = 942 (left) and sample = 982 (right)

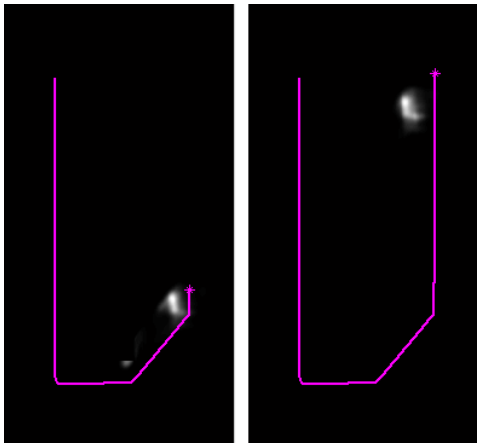

Figure 4.48: MLVOM probabilistic map in sample = 1102 (left) and sample = 1582 (right)

Like the MLWO probabilistic map evolution, it is necessary more time to the MLVOM to start converging to the correct position (Fig. 4.44). After pinpointing the correct position, the MLVOM is able to keep estimating the correct position until the first turn (Fig. 4.46). Since the VOM is not perfect, the MLVOM estimation leaps form one side to another during the ”U” turn (Fig. 4.47). In the last straight line, the MLVOM is able to recover ending with a estimation very near to the mobile robot correct position(Fig. 4.48). Comparing to the MLWO probabilistic map evolution, it can be seen that the MLVOM is able to recover and end with a estimation closer to the mobile robot correct position, due to the building block used.

4.6 MLVOM kidnapping problem experimental results

With the simulation results in section. 3.2.4 it was proved that the Markov localization is a method that can solve the mobile robot kidnapping problem successfully. Therefore, the MLVOM will be compared with MLWO in these experimental tests, in order to confirm that the VOM is better than the WO in this specific situation, which will support the conclusion done in section 4.4 . Similar to the simulation tests, there are two different trajectories, corresponding to two types of kidnapping,

change of position and change of position and direction. Both trajectories wait until sample 300, where the localization method already converge to the mobile robot correct position, to kidnap the mobile robot. In the first type of kidnapping, change of position, the fact that the robot is just placed few meters ahead, without changing the direction, means that the building block should be able to correctly estimate the mobile robot correct attitude even with the kidnapping. On the other hand, when the change of direction it is added to the equation, the building block is expected to face serious difficulties, which may prevent the MLVOM from correctly estimate the mobile robot correct position.

First, it will be presented the results for the first type of kidnapping, the change of position, followed by the second type of kidnapping, change of position and direction. In both cases, it will be showed the position estimation, the building block attitude estimation and the MLVOM probabilistic map evolution.

4.6.1 Results for change of position

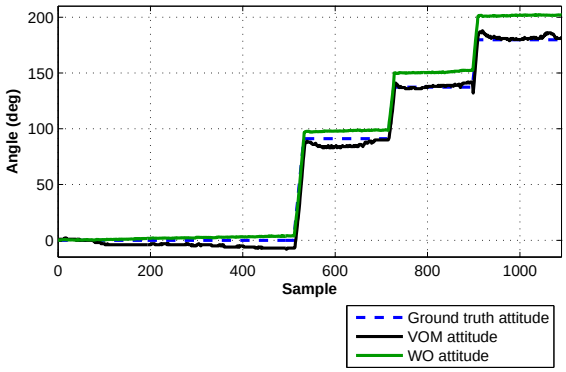

Figure 4.49: Attitude estimation by WO and VOM when position is changed

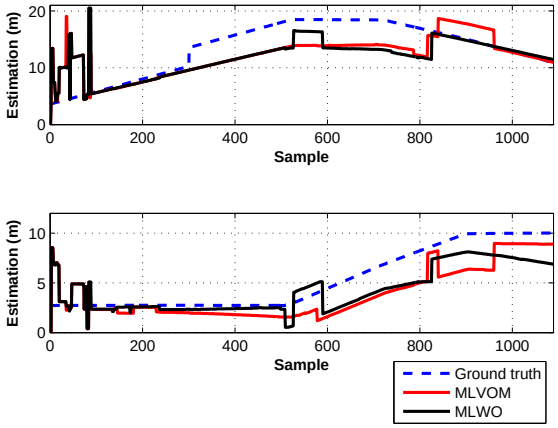

Figure 4.50: x position (top) and y position (bottom) estimation when position is changed

As it was expected, the Markov localization is indeed able to solve the mobile robot kidnapping problem. Both Markov localization methods are able to recover from the mobile robot kidnapping,

however in the y position figure (Fig.4.50) it is possible to see the difference between both building blocks, whereas the MLVOM progressively get closer to the mobile robot correct position, the MLWO gets further away from it each sample. Despite not being as fast as it was in the simulation tests (section 3.2.4), it still is a great result considering the fact that the building block is constantly influencing negatively the localization method, as it is possible to see in Fig. 4.49. In this figure it is possible to see that the VOM attitude estimation is much closer to the mobile robot correct attitude than the WO, which is completely unaware that the mobile robot was kidnapped. However in this case, since the direction is still the same, the building block wont affect the localization performance as much as in the case where the direction change.

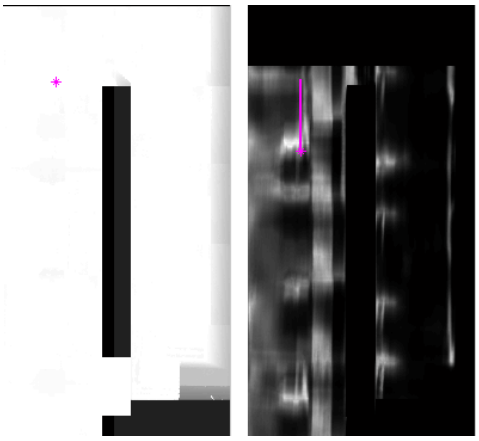

Figure 4.51: MLVOM probabilistic map in sample = 1 (left) and sample = 152 (right)

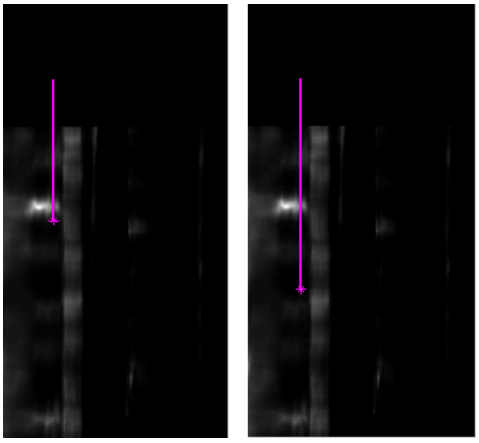

Figure 4.52: MLVOM probabilistic map in sample = 299 (left) and sample = 300 (right)

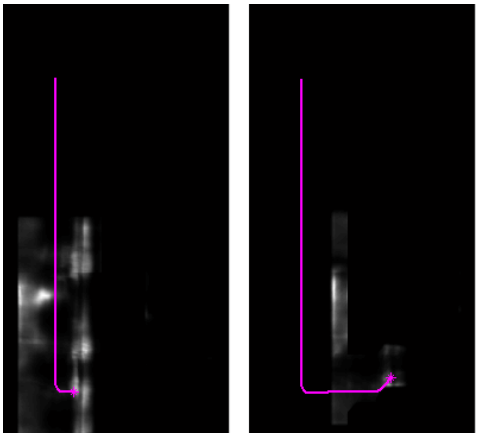

Figure 4.53: MLVOM probabilistic map in sample = 562 (left) and sample = 757 (right)

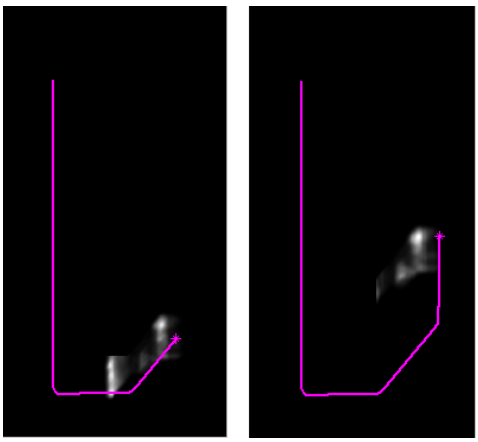

Figure 4.54: MLVOM probabilistic map in sample = 857 (left) and sample = 1087 (right)

In Fig.4.51 shows that the MLVOM is slowly converging to the mobile robot correct position, which is completely reached before sample 300, when the mobile robot is kidnapped (Fig. 4.52). Afterwards, the MLVOM is only able to estimate the mobile robot position after the ”U” turn, ending very near to the mobile robot correct position (Fig. 4.53 and Fig.4.54).

4.6.2 Results for change of position and direction

In this trajectory, the building block influence greatly the localization method performance. As it is possible to see in figure 4.55, the WO is completely unaware that the mobile robot was kidnapping and keep estimating the mobile robot attitude like nothing happened. On the other hand, the VOM is able to recognize that something happened, however its estimation is still far from the mobile robot correct attitude. This means that the MLWO position estimation will be awful where as the MLVOM will be closer to the mobile robot correct position, as it is possible to see in figure 4.56.

In this figure, the MLVOM estimation is aware that the mobile robot was kidnapped and, even with the fact that the VOM is not able to correctly estimate the mobile robot attitude, the MLVOM is slowly recovering from the kidnapping in the x position figure (Fig.4.56). On the other hand, the wrong attitude estimation will push the MLVOM probabilistic map away from the mobile correct position, as

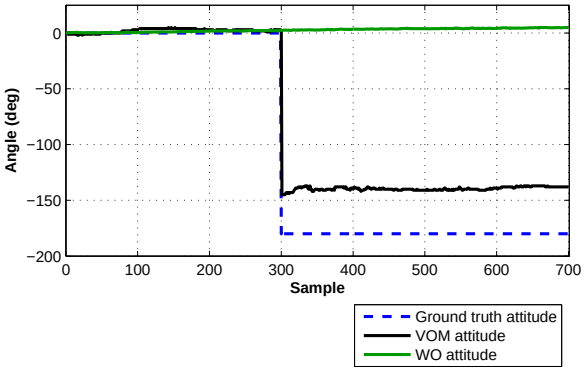

Figure 4.55: Attitude estimation by WO and VOM when position and direction are changed

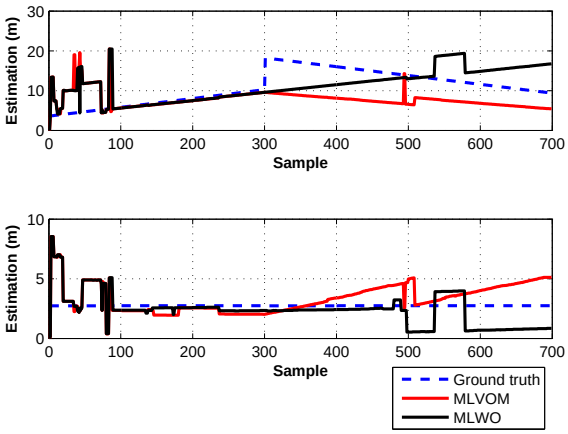

Figure 4.56: x position (top) and y position (bottom) estimation when position and direction are changed

it is possible to see in the y position and in the MLVOM probabilistic map evolution.

Similar to the previous trajectory, the MLVOM estimation is completely converged to the mobile robot correct position just before the mobile robot kidnapping (Fig.4.57 and Fig.4.58). However, unlike the previous trajectory, after the mobile robot kidnapping, the building block influence greatly the MLVOM performance. As it is possible to see in figure 4.59, the probabilistic map is displaced by the attitude estimated by the VOM, which makes impossible for the MLVOM to correctly estimate the mobile robot correct attitude.

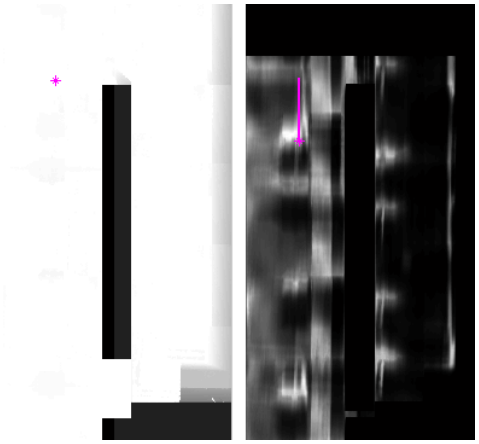

Figure 4.57: MLVOM probabilistic map in sample = 1 (left) and sample = 132 (right)

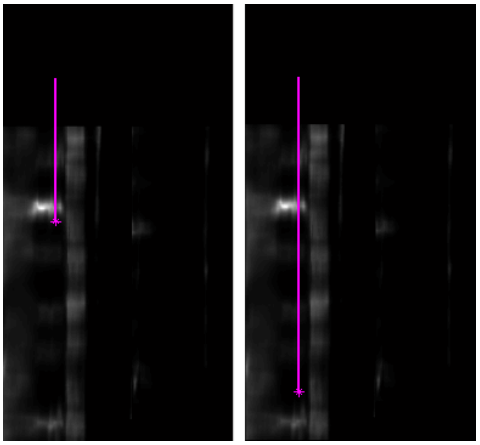

Figure 4.58: MLVOM probabilistic map in sample = 299 (left) and sample = 300 (right)

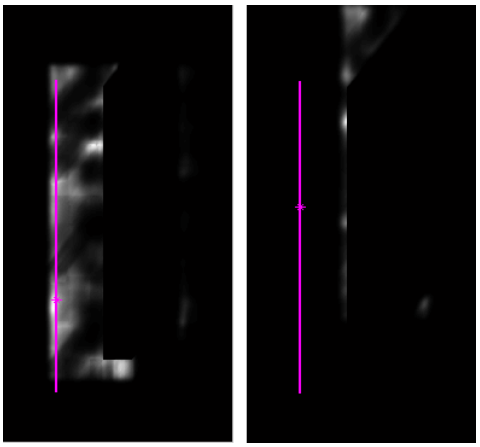

Figure 4.59: MLVOM probabilistic map in sample = 497 (left) and sample = 697 (right)

5

Conclusions and Future Work

Contents

5.1 Conclusion about section 4.3

Analysing the results obtained in section 4.3, it is possible to conclude that the RGB camera, when the image match is almost perfect, can get better results then the Depth camera. In others situations, the RGB camera has an horrible results. This is proves that by using RGB images in your localization method, you are turning it into a fragile method, which only work in specific situations. On the other hand, the Depth camera always have an acceptable result. Even with all variable mixed up, it is possible to see that the estimated position is near the desired position. In other words, the Depth camera is more robust than the RGB camera , which means it is a better sensor in this kind of environment, where some variables are always changing.

5.2 Conclusion about section 4.4

With the results in section 4.4 it is possible to confirm that the VOM is better than VO and, consequently, better than the WO. Regarding the WO performance, it is visible the wheel slippage influence in the WO attitude estimation in every experimental test. In any type of trajectory, the WO is inferior than both visual odometry methods, especially in longer trajectories. However, the VO performance is not acceptable in the longer trajectories. Since the VO not only has difficulties when the mobile robot passes under a ceiling zone with low information, but it also forgets the pass, the poor performance in these type of trajectory is understandable. As for the VOM, it calculates the robot attitude with a good accuracy and very close to the ground truth in the more complex and longer trajectories, unlike the VO. Despite showing a better overall performance than the VO, the VOM method is much slower and the performance difference between both method is only significant with the complete world map.

With a far superior method, the VOM is the ideal replacement for the WO, for two reasons. Due to the VOM output, it is not necessary to make major changes in the Markov localization to replace the building block. In addition to that, the VOM uses a sensor that it is also used in the localization method, which mean that, in the end, one less sensor will be used.

5.3 Conclusion about section 4.5

After analysing all the results obtained in section 4.5, it is possible to confirm that the MLWO is not completely superior than the PCA, however the MLVOM is much better than the MLWO and the PCA. The PCA method shows better results than the Markov localization when the world map is small and do not have many repeatable ceiling elements. However, when comparing the PCA method performance with a small world map against the same method with a complete world map, it is visible a performance decrease. Therefore, the PCA is more indicated for solving the local localization problem where the world map is rich and does not face the ceiling element repeatability problem.

Regarding the results comparing both building blocks for the Markov localization, it is possible to confirm that the WO influence negatively the Markov localization. This is more visible in longer trajectories, where the difference between both odometry method is more significant. In the smallest

trajectory, the WO does not have enough time to mislead the Markov localization, showing as good results as the MLVOM. Therefore, in a specific situation, it is possible to use any method and maintain a decent performance. On the other hand, the MLVOM is much better than the MLWO when the trajectory is big enough, where the WO have enough time to mislead Markov localization. In addition to test with two different trajectories, it was also tested the change of the world map. The results confirmed that this factor is much more important when comparing the main localization method, Markov localization against PCA, and not the building block, MLWO against MLVOM. In conclusion, the VOM is overall better than the WO as building block for the Markov localization, obtaining results as good as WO or much better results than the WO. In addition to use a better building block in the localization method, this change also allowed the removal of one sensor, the wheel encoders.

5.4 Conclusion about section 4.6

After analysing the experimental results in section 4.6, it is possible to see that the Markov localization can indeed solve the mobile robot kidnapping problem successfully. The experimental results also allow to conclude that the VOM is a much better building block than the WO when facing this problem. The WO not only has a poor performance for long trajectories, but also is completely unaware that the mobile robot was kidnapped and the direction changed during the process. As for the VOM, the performance is far from perfect in the cases where the direction change, however, being aware that the mobile robot was kidnapped makes the VOM much better than the VO. In order to successfully solve the mobile robot kidnapped, it might be necessary to use the information of another sensor, for example a digital compass.

5.5 Direction for future research

With the conclusion of this thesis, the first couple steps to the creation of an indoor localization method that uses ceiling depth images were done. By knowing the strengths and weakness of the developed work, it was possible to define the next steps:

- First, improve the quality of the complete world map. Instead of building manually the complete world map, a mosaicing technique can be used in order to obtain a much better result. This technique could also be used to improve the mapping part of the VOM;
- Secondly, the way that VOM estimate the mobile robot attitude and velocity is slow. One way to improve this section is to use the Fourier Shift Theorem. Despite being a more mathematical way to compare two images, it is much faster than the method applied in the VOM;
- In addition to change the how the VOM compares two images, it is also possible to reduce the number of attitudes and velocities tested (see Fig. 4.9). Instead of testing a fixed number of angles or velocities, which is the gridding method, the Newton method can be applied. This method just need two points in each iterations, and with the $f'(x_n)$, it can calculate the x_{n+1}.

In other words, it is possible to estimate the minimum of the function of the iteration $n+1$ only knowing the minimum of iteration n, the $f'(x_n)$ and the $f(x_n)$, reducing drastically the number of tests necessary;

- After improving the VOM method, it is possible to combine the output of VOM with a KF. This way, the building block performance would increase significantly. However, with the addition of a KF, it is necessary to add another sensor, the digital compass;
- Finally, with a very high performance building block, the new localization method can be implemented in the mobile robot control.

In the end it is supposed to have an indoor localization method that uses ceiling depth images, without using features to extract information from that images, requiring a very low number of sensors and have a low computational cost.

Bibliography

[1] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *Journal of Artificial Intelligence Research*, vol. 11, pp. 391–427, August 1999.

[2] J. J. Leonard and H. F. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *1991 IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 376–382, June 1991.

[3] M. Betke and L. Gurvits, "Mobile robot localization using landmarks," *1997 IEEE Transactions on Robotics and Automation*, vol. 13, no. 2, pp. 251–263, April 1997.

[4] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*, ser. Intelligent Robotics and Autonomous Agents. USA: MIT Press, August 2005.

[5] D. Scaramuzza, F. Fraundorfer, and R. Siegwart, "Real-time monocular visual odometry for on-road vehicles with 1-point ransac," in *IEEE International Conference on Robotics and Automation (ICRA2009)*, Kobe, Japan, May 2009, pp. 4293–4299.

[6] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, ser. Springer Tracts in Advanced Robotics. Germany: Springer, 2011, vol. 73.

[7] F. Carreira, C. Christo, D. Valério, M. Ramalho, C. Cardeira, J. M. F. Calado, and P. Oliveira, "Experimental validation of a PCA-based localization system for mobile robots in unstructured environments," in *Robotica - 12th International Conference on Autonomous Robot Systems and Competitions*, Guimarães, Portugal, April 2012, pp. 69–74, iDMEC/CSI Internal Report, http://www1.dem.ist.utl.pt/cardeira/Mesh/Ca12b.pdf.

[8] F. Carreira, J. M. F. Calado, C. Cardeira, and P. Oliveira, "Enhanced pca-based localization using depth maps with missing data," in *Robotica - 13th International Conference on Autonomous Robot Systems and Competitions*, Lisbon, Portugal, April 24 2013, pp. 56–63.

[9] J. Rodrigues, F. Carreira, C. Cardeira, P. Oliveira, and J. M. F. Calado, "Experimental validation of a visual odometry system for indoor unstructured environments," in *ICAR 2013 - The 16th International Conference on Advanced Robotics*, Montevideo, Uruguay, November 2013.

[10] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[11] S. Se, D. G. Lowe, and J. J. Little, "Vision-based global localization and mapping for mobile robots," *2005 IEEE Transactions on Roboticss*, vol. 21, no. 3, pp. 364–375, June 2005.

[12] D. Schleicher, L. M. Bergasa, R. Barea, E. Lopez, M. Ocaa, J. Nuevo, and P. Fernandez, "Real-time stereo visual slam in large-scale environments based on sift fingerprints," in *Proceedings of WISP 2007, the IEEE International Symposium on Intelligent Signal Processing*. Alcala de Hernares, Spain: IEEE, October 3-5 2007, pp. 1–6.

[13] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *9th European Conference on Computer Vision ECCV 2006*. Graz, Austria: Springer, May 7 2006, pp. 404–417.

[14] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[15] A. C. Murillo, J. Guerrero, and C. Sagues, "Surf features for efficient robot localization with omni-directional images," in *2007 IEEE International Conference on Robotics and Automation*. Roma, Italy: IEEE, April 10-14 2007, pp. 3901–3907.

[16] M. Cummins and P. Newman, "Fab-map: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[17] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey vision conference*, vol. 15. Manchester, UK, 1988, pp. 50–55.

[18] H. P. Moravec, "Visual mapping by a robot rover," in *Proceedings of the 6th international joint conference on Artificial intelligence*, vol. 1. Toyko, Japan: Morgan Kaufmann Publishers Inc., August 20-23 1979, pp. 598–600.

[19] J. S. Beis and D. G. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Juan, Puerto Rico: IEEE, June 17-19 1997, pp. 1000–1006.

[20] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[21] P. J. Rousseeuw, "Least median of squares regression," *Journal of the American statistical association*, vol. 79, no. 388, pp. 871–880, 1984.

[22] S. Thrun, "Particle filters in robotics," in *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*. Alberta, Canada: Morgan Kaufmann Publishers Inc., August 1-4 2002, pp. 511–518.

[23] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial intelligence*, vol. 128, no. 1, pp. 99–141, May 2001.

[24] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American statistical association*, vol. 94, no. 446, pp. 590–599, March 1999.

[25] A. Doucet, N. De Freitas, and N. Gordon, *An introduction to sequential Monte Carlo methods*. Springer, 2001.

[26] M. Isard and A. Blake, "Condensation: conditional density propagation for visual tracking," *International journal of computer vision*, vol. 29, no. 1, pp. 5–28, 1998.

[27] S. Lenser and M. Veloso, "Sensor resetting localization for poorly modelled mobile robots," in *2000 IEEE International Conference on Robotics and Automation . ICRA'00.*, vol. 2. San Francisco, USA: IEEE, August 24-28 2000, pp. 1225–1232.

[28] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," in *Proceedings of the National Conference on Artificial Intelligence*. Orlando, USA: JOHN WILEY & SONS LTD, July 18-22 1999, pp. 343–349.

[29] S. Thrun, D. Fox, W. Burgard *et al.*, "Monte carlo localization with mixture proposal distribution," in *Proceedings of the National Conference on Artificial Intelligence*. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2000, pp. 859–865.

[30] R. L. McGreevy, "Reverse monte carlo modelling," *Journal of Physics: Condensed Matter*, vol. 13, no. 46, p. R877, 2001.

[31] A. F. Smith and A. E. Gelfand, "Bayesian statistics without tears: a sampling–resampling perspective," *The American Statistician*, vol. 46, no. 2, pp. 84–88, 1992.

[32] H. Köse and H. Akın, "The reverse monte carlo localization algorithm," *Robotics and Autonomous Systems*, vol. 55, no. 6, pp. 480–489, 2007.

[33] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous robot vehicles*. New York, USA: Springer, 1990, pp. 167–193.

[34] A. J. Davison and D. W. Murray, "Simultaneous localization and map-building using active vision," *2002 IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 865–880, July 2002.

[35] P. Jensfelt, D. Kragic, J. Folkesson, and M. Bjorkman, "A framework for vision based bearing only 3d slam," in *2006 IEEE International Conference on Robotics and Automation. ICRA 2006.* Orlando, USA: IEEE, May 15-19 2006, pp. 1944–1950.

[36] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *2007 IEEE Transactions onPattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[37] T. Lemaire and S. Lacroix, "Monocular-vision based slam using line segments," in *2007 IEEE International Conference on Robotics and Automation*. Rome, Italy: IEEE, April 10-14 2007, pp. 2791–2796.

[38] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, and H. Durrant-Whyte, "Simultaneous localization and mapping with sparse extended information filters," *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 693–716, 2004.

[39] Y. Liu and S. Thrun, "Results for outdoor-slam using sparse extended information filters," in *2003 IEEE International Conference on Robotics and Automation, ICRA'03.*, vol. 1. Taipei, Taiwan: IEEE, Setember 14-19 2003, pp. 1227–1233.

[40] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *Proceedings of the National conference on Artificial Intelligence*. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press;, 2002, pp. 593–598.

[41] T. D. Barfoot, "Online visual motion estimation using fastslam with sift features," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2005*. Alberta, Canada: IEEE, August 2-6 2005, pp. 579–585.

[42] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, August, 9-15 2003, pp. 1151–1156.

[43] I. Jolliffe, *Principal Component Analysis*, 2nd ed., ser. Springer Series in Statistics. Germany: Springer-Verlag, 2002.

[44] P. Oliveira, "Interpolation of signals with missing data using pca," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2006)*, vol. 3, pp. 828–831, May 2006.

[45] P. J. Oliveira, "Mmae terrain reference navigation for underwater vehicles using pca," *International Journal of Control*, vol. 80, no. 7, pp. 1008–1017, 2007.

[46] M. Artač, M. Jogan, and A. Leonardis, "Mobile robot localization using an incremental eigenspace model," in *2002 IEEE International Conference on Robotics and Automation*, Washington, DC, USA, May 2002, pp. 1025–1030.

[47] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *2011 IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, December 08 2011.

[48] C. Cardeira and J. Sá da Costa, "A low cost mobile robot for engineering education," in *31st Annual Conference of IEEE Industrial Electronics Society, 2005. IECON 2005.*, Raleigh, USA, 2005, pp. 2162–2167.

A

Simulation results

A.1 VO and VOM simulation results

A.1.1 Straight line trajectory

Dome world

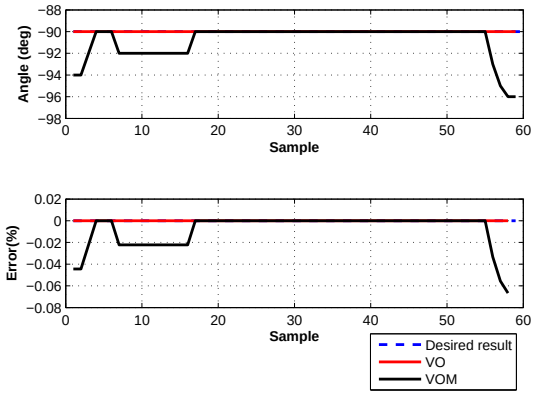

Figure A.1: Attitude estimation(top) and its error(bottom) on the straight line trajectory, in the Dome world

IST world

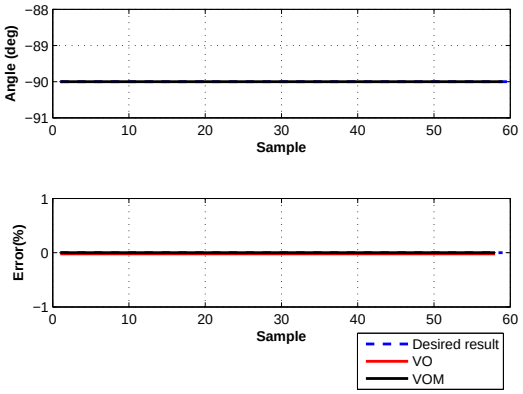

Figure A.2: Attitude estimation(top) and its error(bottom) on the straight line trajectory, in the IST world

A.1.2 Square trajectory

Dome world

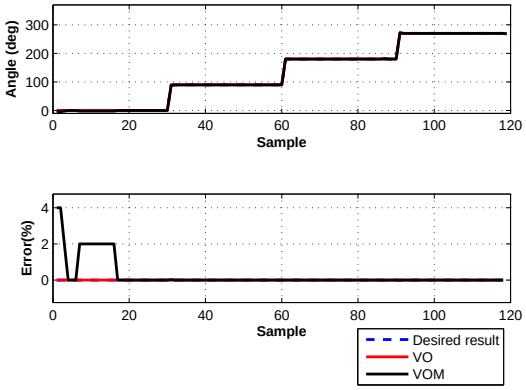

Figure A.3: Attitude estimation(top) and its error(bottom) on the box trajectory, in the Dome world

IST world

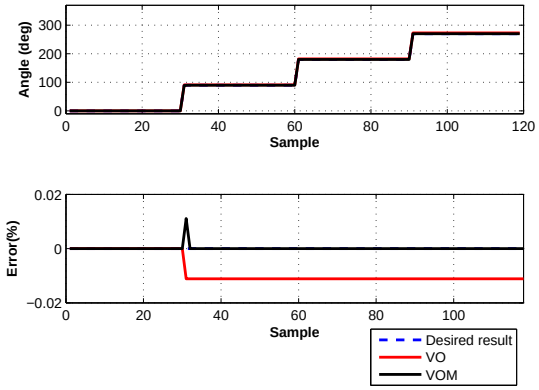

Figure A.4: Attitude estimation(top) and its error(bottom) on the box trajectory, in the IST world

A.1.3 Circle trajectory

Dome world

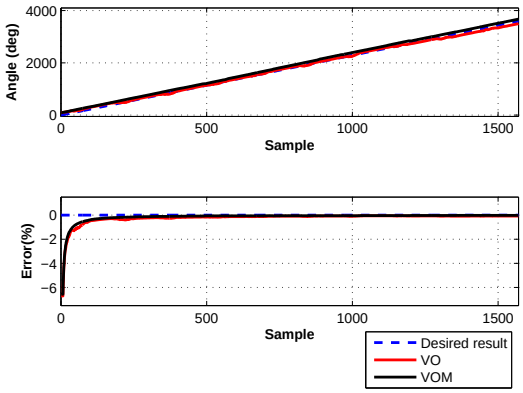

Figure A.5: Attitude estimation(top) and its error(bottom) on the circle trajectory, in the Dome world

IST world

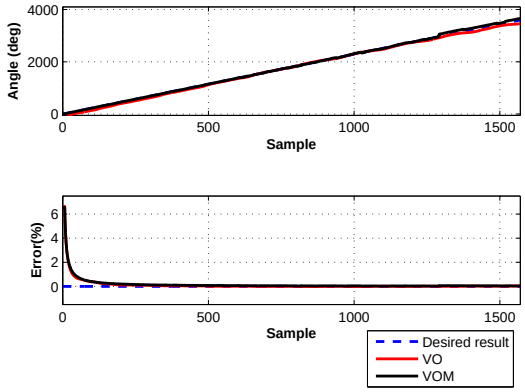

Figure A.6: Attitude estimation(top) and its error(bottom) on the circle trajectory, in the IST world

A.2 Markov localization simulation results

A.2.1 Straight line trajectory

Dome world

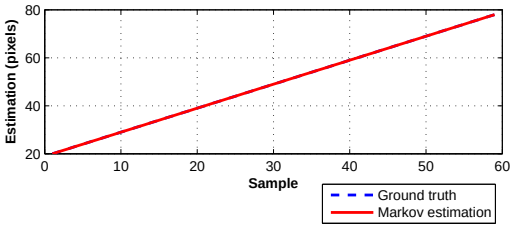

Figure A.7: X position estimation on straight line trajectory in Dome world

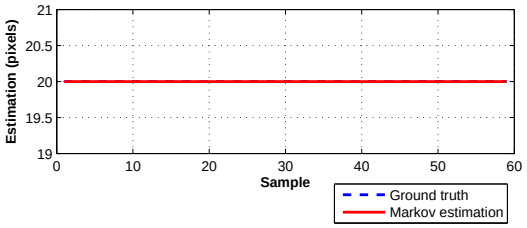

Figure A.8: Y position estimation on straight line trajectory in Dome world

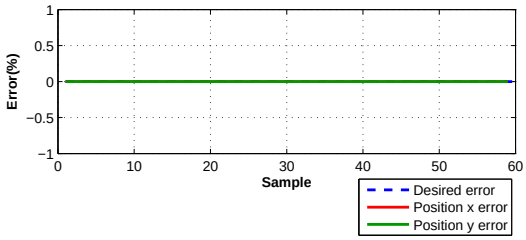

Figure A.9: Error of position estimation on straight line trajectory in Dome world

IST world

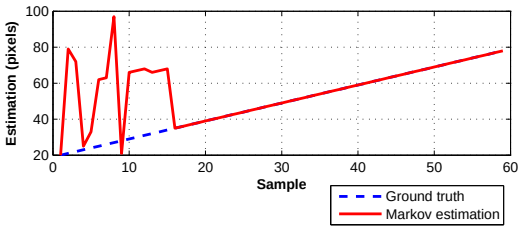

Figure A.10: X position estimation on straight line trajectory in IST world

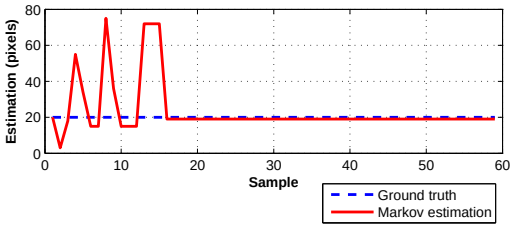

Figure A.11: Y position estimation on straight line trajectory in IST world

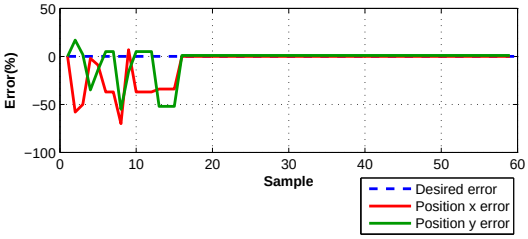

Figure A.12: Error of position estimation on straight line trajectory in IST world

A.2.2 Square trajectory

Dome world

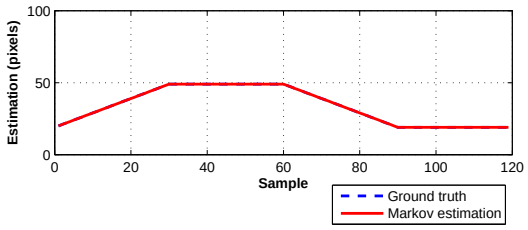

Figure A.13: X position estimation on square trajectory in Dome world

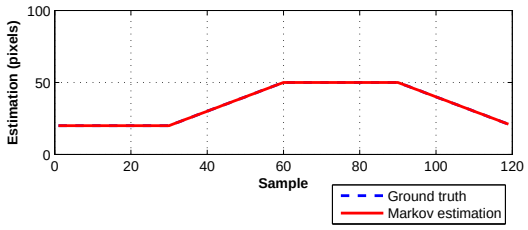

Figure A.14: Y position estimation on square trajectory in Dome world

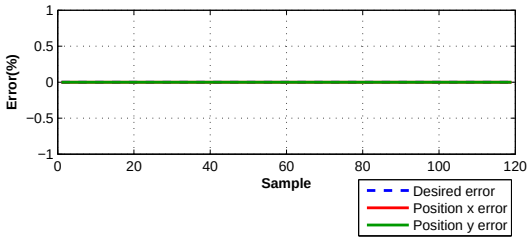

Figure A.15: Error of position estimation on square trajectory in Dome world

IST world

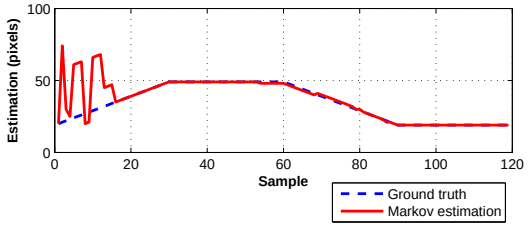

Figure A.16: X position estimation on square trajectory in IST world

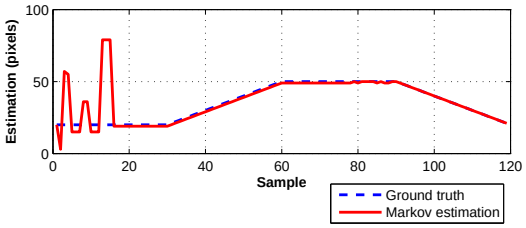

Figure A.17: Y position estimation on square trajectory in IST world

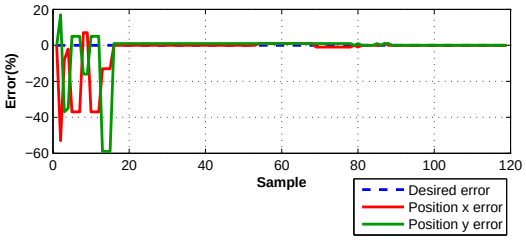

Figure A.18: Error of position estimation on square trajectory in IST world

A.2.3 Circle trajectory

Dome world

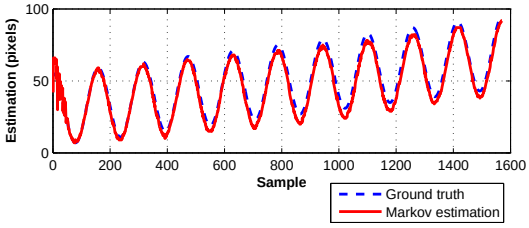

Figure A.19: X position estimation on circle trajectory in Dome world

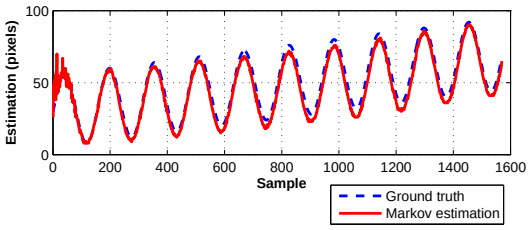

Figure A.20: Y position estimation on circle trajectory in Dome world

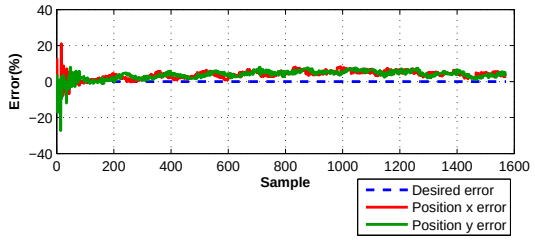

Figure A.21: Error of position estimation on circ trajectory in Dome world

IST world

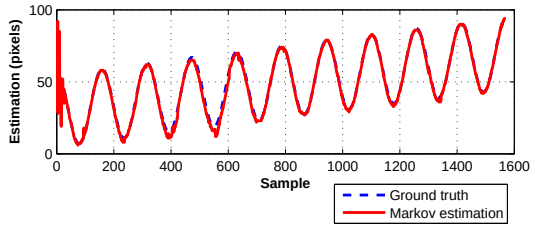

Figure A.22: X position estimation on circle trajectory in IST world

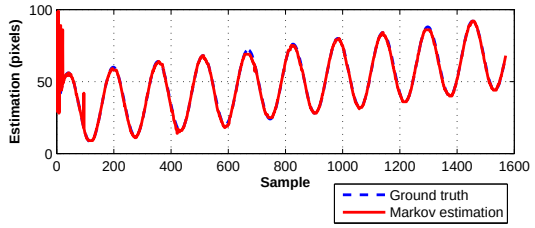

Figure A.23: Y position estimation on circle trajectory in IST world

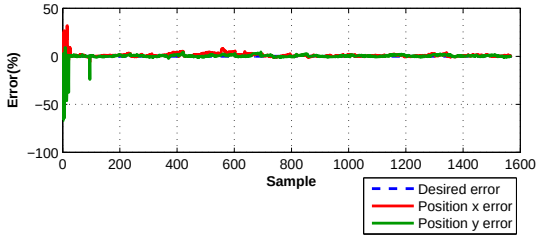

Figure A.24: Error of position estimation on circle trajectory in IST world

A.3 Markov localization Kidnapped simulation results

A.3.1 Change of position

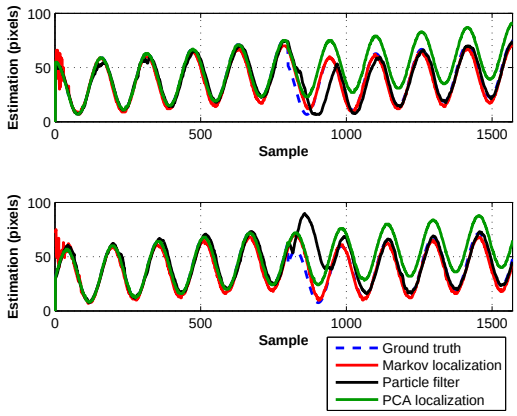

Figure A.25: X(top) and Y(bottom) position estimation on circle trajectory in dome world

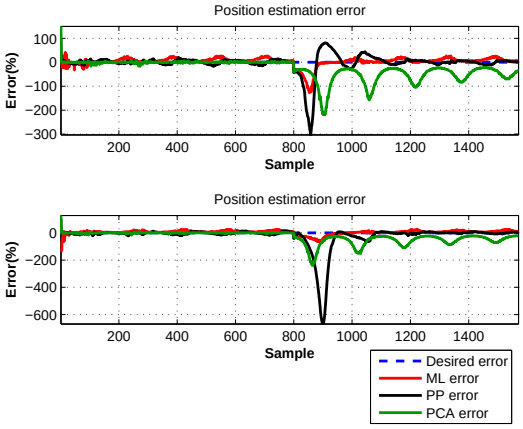

Figure A.26: Position estimation error on circle trajectory in dome world

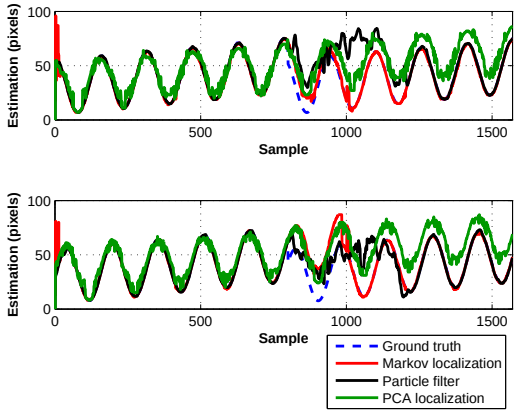

Figure A.27: X(top) and Y(bottom) position estimation on circle trajectory in IST world

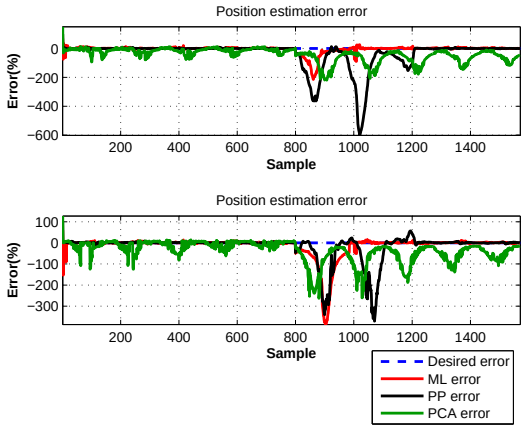

Figure A.28: Position estimation error on circle trajectory in IST world

A.3.2 Change of position and direction

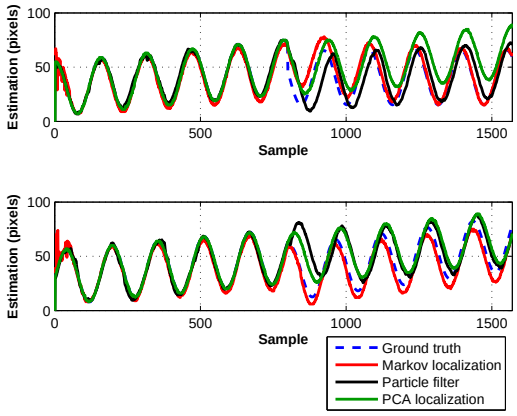

Figure A.29: X(top) and Y(bottom) position estimation on circle trajectory in dome world

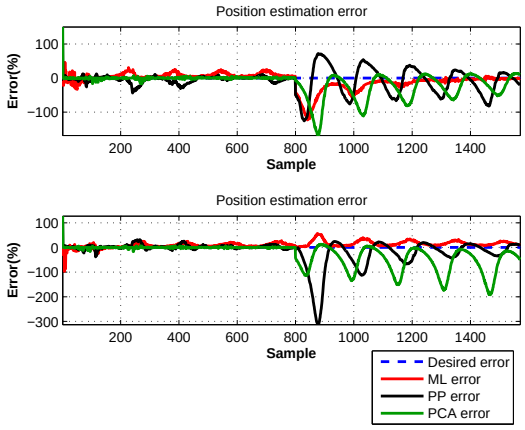

Figure A.30: Position estimation error on circle trajectory in dome world

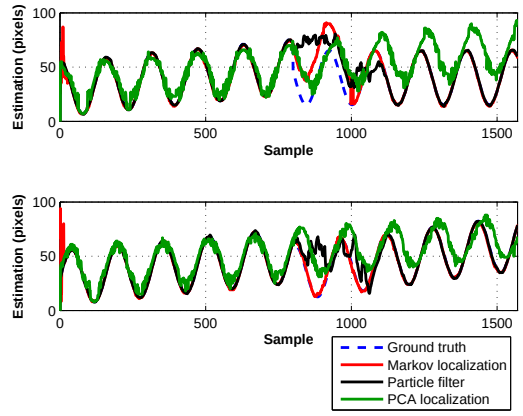

Figure A.31: X(top) and Y(bottom) position estimation on circle trajectory in IST world

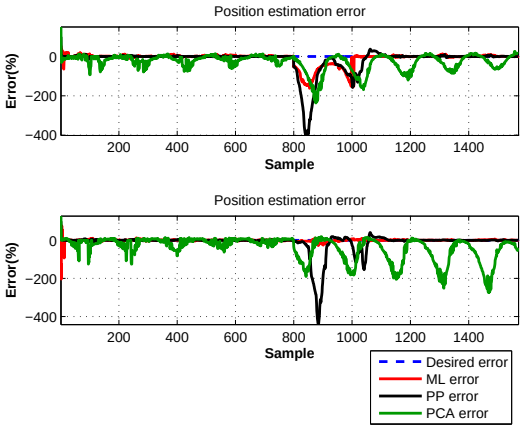

Figure A.32: Position estimation error on circle trajectory in IST world