INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Simultaneous Localization and Mapping using Vision for Search and Rescue Robots

Filipe Miguel Oliveira de Jesus

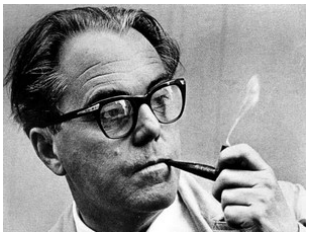
Thesis submitted to the Department of Electrical and Computer Engineering in partial fulfilment of the requirements for the degree of

Master of Science in Electrical and Computer Engineering

Panel

President: Doutor Carlos Filipe Gomes Bispo
Advisor: Doutor Rodrigo Martins de Matos Ventura
Examiner: Doutor Vítor Manuel Ferreira dos Santos

April 2012



"The machine has ***no feelings****, it feels* ***no fear*** *and* ***no hope****...*
It operates according to the pure logic of ***probability****.*
For this reason, I assert that ***the robot perceives more accurately than man.****"*

Max Rudolf Frisch in *Homo Faber* (1957)

Abstract

This thesis addresses an online 6D *Simultaneous Mapping and Localization* technique (SLAM) for a tracked wheel robot in an unknown and unstructured environment. While the robot pose is represented by a 3D position and SO(3) orientation, the environment is mapped with natural landmarks in 3D space, autonomously collected using visual data from feature detectors. An *Extended Kalman Filter* (EKF) is used to compute both pose and landmark estimates over time. A motion model using odometry readings from motor encoders and orientation changes measured with an Inertial Measurement Unit is introduced, as well as an observation model with a novel approach that copes with both mono and stereo observations.

Since the computational complexity of EKF grows over a quadratic order with the increasing number of landmarks in the state, a new landmark classifier using a *Temporal Difference Learning* methodology is introduced. This classifier is applied for landmark removal, identifying all undesired landmarks for estimation. The main novelty introduced is a *Dimensional-bounded EKF* that, with the aforementioned classifier and proper criteria for landmark removal and insertion, forces an upper bound to the number of landmarks in the EKF state, reducing the computational complexity up to a constant while not compromising its integrity.

An inverse depth parametrization is used for landmark representation. Two different feature detectors are presented and compared for this thesis: SURF and ORB. All experimental work was done using real data from RAPOSA-NG, a tracked wheel robot developed for *Search and Rescue* missions.

Keywords: *Simultaneous localisation and mapping, Extended Kalman filter, Feature detector, Inverse depth parametrization, Landmark evaluator, Temporal difference learning*

Resumo

Esta tese trata de uma técnica de *Localização e Mapeamento Simultâneo* (SLAM) online em 6D para um robot terrestre num ambiente desconhecido, cuja pose é representada pela sua posição em 3D e a sua orientação definida em SO(3). O mapa é composto por pontos de referência (*landmarks*) naturais, extraídos automaticamente com um detector de pontos em dados visuais. É utilizado o *Extended Kalman Filter* (EKF) para calcular as estimativas da pose e das *landmarks* ao longo do tempo. Um modelo de movimento que utiliza leituras de odometria e velocidades angulares medidas com um sensor inercial é introduzido, assim como um modelo de observação que lida com informações monoculares e estereo em simultâneo.

Como a complexidade computacional do EKF cresce acima de uma ordem quadrática com o aumento do número de landmarks, um novo classificador de *landmarks* que utiliza técnicas de *Aprendizagem por Diferenças Temporais* é introduzido. Este classificador pode ser utilizado para remoção de *landmarks*, identificando as *landmarks* não desejadas para estimação. A grande novidade introduzida é um *EKF de Dimensão Fechada* que, com o classificador supracitado e critérios próprios para inserção e remoção de *landmarks*, pode limitar o número destas no estado, reduzindo a complexidade computacional do EKF para uma constante sem comprometer o seu desempenho.

Uma parametrização de profundidade inversa é utilizada para representação das *landmarks* no estado. Dois detectores de pontos visuais são apresentados e comparados nesta tese: SURF e ORB. Os resultados experimentais foram obtidos com dados reais do RAPOSA-NG, um robot SAR terrestre para missões de *Busca e Salvamento*.

Palavras Chave: *Localização e mapeamento simultâneo, Extended Kalman filter, , Inverse depth parametrization, Detector de pontos visuais, Classificador de landmarks, Aprendizagem por diferenças temporais*

Acknowledgement

My most sincere thanks to my thesis supervisor, professor Rodrigo Ventura, for all his support during this thesis. Thank you for all the opportunities you have entrusted me and for all the critics that helped me better formalize my knowledge.

I would also like to thank my family, mostly my parents, for putting up with me for so long. Believe me, all material and emotional resources invested on me were not in vain. A special remark to all my father's rides, my mother's "papa Maizena" and to all robot drawings from my cousin Joana that fully inspired me.

Thanks to all my friends and colleagues for being a precious influence during this period of my life. Thanks Henrique Delgado, Jorge Ribeiro and everyone from *"Ventura Group"*. Special thanks for everyone from *"Grupo do Foz, da Dark e dessa gente"*, Marta Lebre, Andreia Rodrigues, Ricardo Doroana and Andreia Santos. Thanks to Margarida Carmello for all times she stayed by my side. Last, but not least, thanks to Rita Santos for motivating me and helping me with english grammar and to a most valuable friend, Sara Cordeiro, for all her support and friendship.

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

SAR	Search and Rescue
RGB	Red, Green, Blue Color Model
XYZ	Representation in X, Y and Z coordinates
R&D	Research and Development
SLAM	Simultaneous Location and Mapping
EKF	Extended Kalman Filter
DBEKF	Dimentional-bounded Extended Kalman Filter
IMU	Inertial Measurement Unit
SURF	Speeded Up Robust Features
ORB	Oriented FAST and Rotated BRIEF
ROS	Robotic Operating System

Contents

Chapter 1

Introduction

This chapter serves as an introduction for this thesis, presenting a problem within the robotics field related to pose and map estimation for most robot platforms and a solution proposal in section 1.1, focused on *Search and Rescue* (SAR) activities and issues. RAPOSA-NG, a SAR semi-autonomous robot used for real tests is then introduced in section 1.2, along with its predecessor, RAPOSA. Section 1.3 reviews the literature and section 1.4 points all major contributions this thesis makes to this area. Finally, section 1.5 ends this chapter by presenting the thesis outline.

1.1 Problem Statement

Nowadays, deploying SAR robots after urban and large-scale catastrophes is of the utmost importance. They intervene within human-inaccessible spaces or environments that may pose any threat to human safety, such as collapsed buildings or environments with radioactive contamination. Their missions can vary from area reconnaissance to finding victims within risky areas, demanding quick responses with extreme caution[1]. As such, most SAR robots require skilful and focused operators to control them successfully. It is then imperative that SAR robots should perform some tasks with an autonomous or semi-autonomous approach. For instance, granting simultaneous localization and mapping capabilities for a SAR robot allows the operator to focus more on different actions, such as operating its course while probing the environment for victims or safer routes without worrying with locating the robot.

It seems to be a *chicken and egg* problem to locate a robot and map the environment in a simultaneous way, as the pose requires a map to be pinpointed but is also needed to build the map layout [1]. In fact, this problem is not trivial at all. Using probabilistic models, one can assume that there is shared uncertainty by both location and map information. If this uncertainty is somehow minimized, both location and mapping estimations will become more

[1] http://spectrum.ieee.org/automaton/robotics/industrial-robots/fukushima-robot-operator-diaries, Accessed on 28/08/2011

Figure 1.1: *iRobot Warrior 710* in Fukushima Dai-1 Nuclear Plant.

accurate [2]. *Simultaneous Location and Mapping* (SLAM) is one of the most promising research fields in robotics, aiming to predict the location of a robot and map its surroundings using both an observation and a motion model with available data from a set of sensors. Probabilistic filters can be applied to reduce uncertainty in an iterative way over time.

Extended Kalman Filter (EKF), when applied to SLAM, proves to work reasonably well with distinct observations and a small state for estimation [3]. However, an optimal solution for SLAM assumes perfect data correspondence over time, and it is quite difficult to accomplish that when dealing with visual observations in real time acquired with perspective variant feature detectors. Also, with the increasing amount of new data for map estimation over time, the computational complexity of EKF grows over a quadratic order, making it obsolete for large amount of data estimation. By upper bounding the state, the computational complexity of EKF becomes constant and, using a well-defined classifier and proper criteria, undesired features are automatically removed in an autonomous way. A side effect from this outlier removal procedure is that the map may become visually sparse, but as long as it meets the SLAM needs for stable predictions, one can use proper techniques to acquire visually more compelling maps or map the surroundings using more appropriate sensors for that effect (e.g. sonars and laser range-finders).

A common configuration for SAR robots uses monocular or stereo vision cameras, as well as rotor encoders for odometry readings and an *Inertial Measurement Unit (IMU)* to measure linear acceleration, angular velocities and orientation changes in three axis. As such, the solution for the SLAM problem proposed on this thesis uses both odometry and IMU readings for movement prediction, as well as monocular and stereo visual data for mapping the environment. The map is represented in state as a set of point landmarks extracted in an autonomous way with feature detectors. By using both monocular and stereo information in a hybrid way, a larger amount of information from visual observations can then be used to solve the SLAM problem.

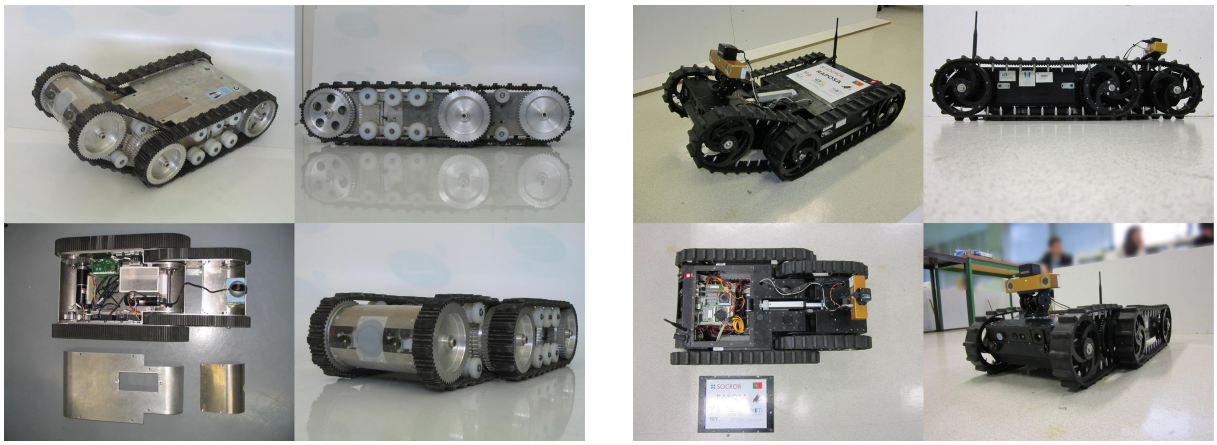


Figure 1.2: Different perspective views and inside view of RAPOSA (Left) and RAPOSA-NG from IST (Right).

1.2 Search and Rescue Robots: RAPOSA and RAPOSA-NG

For this thesis development and experimental results, a real tracked wheel robot named RAPOSA-NG was used. This robot is an upgraded version of RAPOSA, a tracked wheel robot developed in a project with the same name from 2003 to 2005, by a Reseach and Development consortium between a robotics Portuguese company *IdMind©* , *Instituto de Sistemas e Robótica* (ISR) from *Instituto Superior Técnico (IST)*, Lisbon fire-fighters (*Regimento de Sapadores Bombeiros de Lisboa*) and the *Perceptual Robotics Laboratory* from the *University of South Florida* [4]. The goal was to conceive a robot able to intervene and operate in environments hostile or inaccessible to humans for SAR missions. Following the RAPOSA project success, *IdMind©* redesigned it and made this new version for commercial purposes.

Both RAPOSA-NG and its predecessor share the same layout, as shown in figure 1.2. They are both grounded tracked wheel robots with two pairs of tracked wheels, coupled between two different bodies: a base body and a frontal body. IST became the final owner of RAPOSA and acquired the backbones of one RAPOSA-NG from *IdMind©* for research purposes. An *Inertial Measurement Unit* (IMU) was equipped on RAPOSA-NG frontal body, as well as a Stereo Camera for visual data acquisition and to support 3D view with the usage of visual glasses.

Although this work has been configured to be implemented on RAPOSA-NG, it presents a general basis for most grounded robots that share similar traits with RAPOSA-NG.

1.3 Literature Review

It was during the 80s that R.C. Smith and P. Cheeseman presented a stochastic way of measuring a number of spatial relationships with a common world frame by compounding successive states modelled with uncertainty [2]. *Extended Kalman Filter* (EKF) was then applied to es-

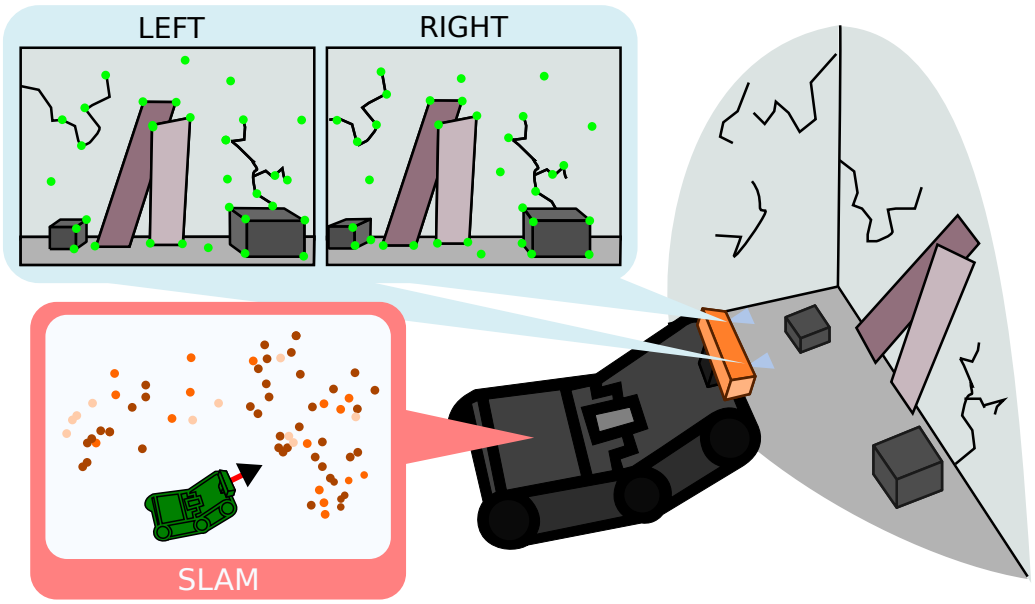


Figure 1.3: Graphical depiction of the proposed SLAM using visual features from a stereo camera.

timate it in an iterative way, computing a proper gain that would weight both observable and prior spatial relationships. Their remarkable work gave rise to a probabilistic approach for the robotics field and many researchers strived to apply it to a real case scenario. J.J. Leonard and H.F. Durrant-Whyte not only accomplished that in 1991 with a real robot in a known environment, but also introduced *Simultaneous Localization and Mapping* (SLAM) terminology to the robotics field and the concept of *geometric beacons*: natural landmarks present in the environment that could be successively observed with reliability and be well described in terms of a concise geometric parametrization (referred in this thesis simply as *landmarks*) [3]. Geometric beacons can be acquired with many different types of sensors, as long as the aforementioned qualities are maintained.

While EKF tends to be the standard and most widely used filter for non-linear state estimates in SLAM, it suffers from several key problems. For instance, the update step for the posterior state covariance demands a full inversion of the innovation covariance and even with few observations, it retains a quadratic order of computational complexity. Also, the Kalman gain obtained is not optimal, as the system is linearised. Many researchers have proposed different techniques to overcome some of these issues in SLAM, such as the usage of Unscented Kalman Filter for a more optimal Kalman gain over non-linear systems [5] [6], particle filters for a posterior state estimation [7] and Extended Information Filter applied to sparse information matrices [8]. Nonetheless, EKF remains a powerful tool for solving the SLAM problem, still currently used for a number of applications.

Most literature regarding SLAM algorithms points out the usage of lasers and/or sonars for map observation [1]. While they prove to be good sensors for the mapping purpose when using techniques such as occupancy grid mapping [9] and ICP [10], cheaper sensors for mapping

can be also used that nonetheless are mandatory in a SAR robot, e.g. visual cameras.

Civera and Davison proposed a real-time algorithm which recovers the location of a monocular camera over time using SLAM with a random walk model [11]. However, feature initialization requires more than one observation and a proper triangulation for an initial depth estimate. Also, it needs to acquire landmarks with known depth for scale initialization. Thus, Civera and Davison presented an *inverse depth parametrization* that represents landmarks uncertainty with more accuracy than the standard XYZ parametrization [12]. The increase of accuracy can be justified by the higher degree of linearity of the inverse depth parametrization over XYZ parametrization. However, this representation parametrizes each landmark with double the parameters, increasing the EKF complexity even further. They also defined a landmark classifier that removes 50% of all predicted landmarks that should be visible but are not detected by any feature detector per iteration. This approach lead to the landmark classifier introduced in this thesis. The usage of a random walk model assumes a well behaved motion with smooth linear and angular velocities for all time, a condition that surely fails when, for instance, a robot climbs up a set of stairs.

Pinies, Lupton, *et al.* added the usage of an IMU to the vision SLAM with inverse depth parametrization [13]. In fact, having orientation changes measured with an IMU, the uncertainty of the camera location is better modelled. However, it does not decrease the uncertainty when only linear motion is observed, which leads to the need of odometry inclusion presented in this thesis. As for the scalability problem, in order to solve it, this thesis extends the inverse depth parametrization usage for stereo vision as well.

BiCamSLAM, introduced in 2007 by J. Sola *et al.*, uses monocular information from each camera of a stereo rig instead of applying directly stereo information from epipolar geometry [14]. For this thesis, a different approach is presented, where both monocular and stereo observations are added to the state with the inverse depth representation but with a known depth (and a proper model for the uncertainty using epipolar geometry) for the stereo case.

Autonomous feature detectors and descriptors such as SIFT [15], SURF [16] and, more recently, ORB [17], contributed for the visual detection of landmarks over the SLAM. By presenting a subset of interest points (features) from each image with unique descriptors in an autonomous way, it allows the Visual SLAM to detect and classify visual landmarks for mapping purposes.

1.4 Main Contributions

This thesis presents two major contributions:

1. The capability to introduce landmarks in the state from either monocular or stereo visualizations using the inverse depth parametrization in an hybrid and elegant way. All landmarks are treated the same way, regardless of their origin. As such, they can be mod-

elled with both monocular and stereo observation models. While monocular observations have no depth information, they allow the estimation of depth through parallax changes during time. Also, the motion model uses sensor fusion between odometry and (IMU) readings, proving to be effective for robots with tracked wheels such as RAPOSA-NG. The SLAM implemented in this thesis requires neither observation initialization, artificial landmarks nor prior movement knowledge from the robot, making all the process for SLAM completely autonomous.

2. A novel approach for the EKF, entitled *Dimensional-bounded EKF (DBEKF)*, is introduced, imposing an upper bound for the number of landmarks present in state to be estimated. This condition is granted by eliminating unwanted landmarks, as well as landmarks with no contribution to the state. In order to classify the utility of each landmark, a landmark classifier computed through a *Temporal Difference Learning* method is defined [18]. If the map is only needed for pose estimation and not for visual analysis from the user, this proves to be highly effective for the the SLAM, since the computational load becomes constant over time;

A comparison between feature detectors ORB and SURF is presented as well, where both performances are evaluated for this problem.

1.5 Thesis Outline

The remainder of this thesis is given as follows: Chapter 2 introduces all theoretical aspects, presenting definitions and formulations related to the 6D EKF SLAM with inverse depth parametrisation and both odometry and IMU readings, feature detectors and a novel dimensional-bounding algorithm with proper classifiers and criteria for EKF. Chapter 3 presents a physical description of RAPOSA-NG, including all sensors and actuators, while covering the system architecture and calibrations made. Chapter 4 presents all tests and respective results from real scenarios with RAPOSA-NG. Finally, Chapter 5 concludes this thesis and presents some ideas and suggestions for future work.

Chapter 2

Theoretical Background

This chapter is focused on all theoretical aspects relevant for this thesis. It starts in section 2.1 with the introduction of *Extended Kalman Filter* (EKF) as a solution for the *Simultaneous Localization and Mapping* (SLAM) problem, how it is computed and both motion and observation models used are presented, as well as new landmark insertion to the state from both mono and stereo observations. Section 2.2 makes reference to two feature detectors and descriptors, SURF and ORB, as well as their usage for data correspondence between landmarks and visual features. Finally, section 2.3 introduces a new Landmark Classifier and how it can infer on the EKF to upper limit the state, resulting in the *Dimensional-bounded Extended Kalman Filter* (DBEKF).

2.1 Extended Kalman Filter for Pose and Map Estimation

2.1.1 Introduction

Kalman Filter became a major reference for the probabilistic robotics field as it iteratively computes an optimal estimate over time from a given set of observations, knowing the stochastic process model. It presents an elegant closed-form solution to the *Bayes Filter* problem, provided that the state estimate is represented as a gaussian distribution and both motion and observation models are linear systems with additive zero-mean uncorrelated multivariate gaussian noise [1].

The *Extended Kalman Filter* (EKF) differs from the *Kalman Filter* by coping with non-linear models via first-order Taylor expansion. It can be applied to solve the *Simultaneous Localization and Mapping* (SLAM) problem in this thesis, as pose changes and observations are not modelled in a linear fashion. However, all properties regarding gaussian assumptions remain true, and as such, the robot state estimation, s_t, is represented as a gaussian distribution for all instants,

$$s_t \sim \mathcal{N}(\mu_t, P_t) \tag{2.1}$$

Given a motion model, f, for state transition between instants $t-1$ and t, with the previous state, s_{t-1}, and transition measures, u_t (e.g. odometry and *Inertial Measurement Unit* (IMU) readings),

$$s_t = f(s_{t-1}, u_t) + \epsilon_t \qquad \text{where} \qquad \epsilon_t \sim \mathcal{N}(0, Q_t), \tag{2.2}$$

one can predict the *a priori* state estimate with EKF *predict instant*:

$$\bar{\mu}_t = f(\mu_{t-1}, u_t) \tag{2.3}$$

$$\bar{P}_t = F_{t-1}\, P_{t-1}\, {F_{t-1}}^\top + Q_t \tag{2.4}$$

where the jacobian F_{t-1} is

$$F_{t-1} = \left.\frac{\partial f}{\partial s_t}\right|_{\mu_{t-1}, u_t}. \tag{2.5}$$

If sensor observations are available, an observation model, g, is used to compute the expected observations from a given state. It is assumed that real observations are covered by the observation model with an added gaussian error noise,

$$z_t = g(s_t) + \delta_t \qquad \text{where} \qquad \delta_t \sim \mathcal{N}(0, R_t). \tag{2.6}$$

In the *update* step, the error between real observations and expected ones, b_t, is given as a gaussian distribution:

$$b_t \sim \mathcal{N}(\hat{b}_t, B_t) \tag{2.7}$$

$$\hat{b}_t = z_t - g(\bar{\mu}_t) \tag{2.8}$$

$$B_t = H_t\, \bar{P}_t\, {H_t}^\top + R_t \tag{2.9}$$

where the jacobian H_t is

$$H_t = \left.\frac{\partial g}{\partial s_t}\right|_{\bar{\mu}_t}. \tag{2.10}$$

Finally, EKF *update* computes a "near-optimal" Kalman Gain, K_t,

$$K_t = \bar{P}_t\, {H_t}^\top\, {B_t}^{-1}, \tag{2.11}$$

and closes the feedback control for an *a posterior* state estimate

$$\mu_t = \bar{\mu}_t + K_t \hat{b}_t, \tag{2.12}$$

$$P_t = (I - K_t H_t) \bar{P}_t. \tag{2.13}$$

2.1.2 State Definition and Quaternion Representation

It is assumed that the robot being located uses both odometry and IMU readings for pose prediction and visual data from a monocular or stereo camera. Two coordinate frames are introduced for a better description of the problem, depicted in figure 2.1:

1. **World frame**, (X^W, Y^W, Z^W): a fixed coordinate frame with respect to both pose and map estimations.
2. **Camera frame**, $(X^{C_t}, Y^{C_t}, Z^{C_t})$: it shares its origin and orientation with the robot camera pose for instant t in the world frame. Note that the camera frame origin may not coincide to the focal point location. In fact, for a stereo camera, the camera frame origin is assumed to be located between the left and the right camera focal points. However, it shares the same orientation as both left and right camera focus, where Z^{C_t} is perpendicular to their image planes. The camera frame at the initial instant is assumed to be equal to the world frame.

It is then imperative that the state at any instant t comprises both camera and map information,

$$s_t = \Big(\underbrace{r_t^\top \; q_t^\top}_{\text{Camera State}} \quad \underbrace{y_1^\top \; \dots \; y_n^\top}_{\text{Map State}} \Big)^\top \tag{2.14}$$

Both vector r_t and the unitary quaternion q_t have the camera position in the world frame and spatial rotation from the world frame to the camera frame at instant t, respectively [19]. This SLAM maps the environment by observing point features from a monocular or stereo visual camera and introducing them on the state as landmarks. Thus, the map state is described by a set of n 3D-point landmarks, y_1 to y_n, each represented with an *Inverse Depth Parametrization* that is briefly introduced in subsection 2.1.3 [12]. It is assumed that the environment surrounding the robot is static, making map representation invariant with time.

Regarding q_t, although less intuitive than euler angles, quaternion representation for orientation suffers no singularity, avoiding the gimbal lock problem. Also, they are advantageous over rotation matrices for real time implementations, as quaternions are more compact, numerically stable and computationally efficient. The quaternion is kept in state in the form of vector, as

$$q_t = \Big(q_{w_t} \; q_{x_t} \; q_{y_t} \; q_{z_t} \Big)^\top \tag{2.15}$$

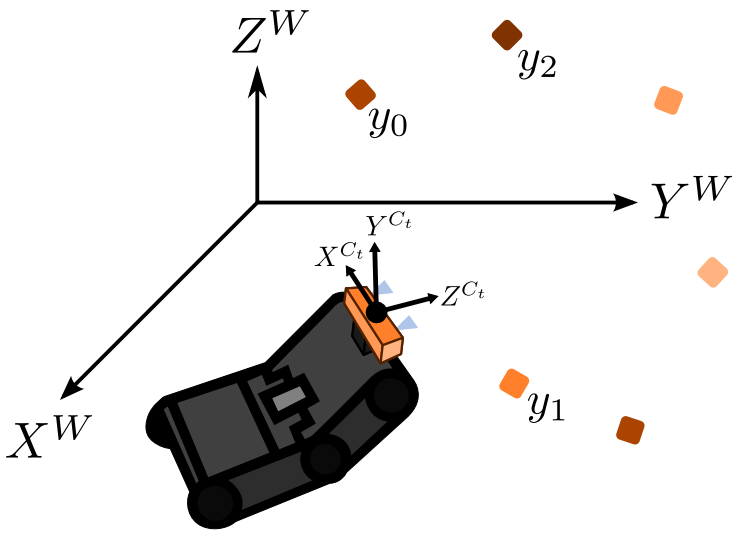


Figure 2.1: Possible state representation at instant t.

Also, the spatial rotation from the camera frame at instant t to the world frame is given by the quaternion conjugate, q_t^*,

$$q_t^* = \begin{pmatrix} q_{wt} & -q_{xt} & -q_{y_t} & -q_{zt} \end{pmatrix}. \tag{2.16}$$

Note that the quaternion has four parameters in state but represents a transform in SO(3). As such, all values are conditioned to

$$q_{wt}^2 + q_{xt}^2 + q_{y_t}^2 + q_{zt}^2 = 1, \tag{2.17}$$

and this condition must be always satisfied for a proper representation of a rotation.

For cases where an orthogonal matrix representation is needed, an unitary quaternion q can be easily converted to a rotation matrix, denoted R_q, by

$$R_q = \begin{bmatrix} q_w^2 + q_x^2 - q_y^2 - q_z^2 & 2\,(q_x\,q_y - q_w\,q_z) & 2\,(q_x\,q_z + q_w\,q_y) \\ 2\,(q_x\,q_y + q_w\,q_z) & q_w^2 - q_x^2 + q_y^2 - q_z^2 & 2\,(q_y\,q_z - q_w\,q_x) \\ 2\,(q_x\,q_z - q_w\,q_y) & 2\,(q_y\,q_z + q_w\,q_x) & q_w^2 - q_x^2 - q_y^2 + q_z^2 \end{bmatrix}. \tag{2.18}$$

Figure 2.2 depicts a flowchart of this SLAM. It assures that the prediction and update instant are computed when motion and observation data are available, respectively. It also depicts the state initialization right on start and landmark inclusion to the state whenever it is needed.

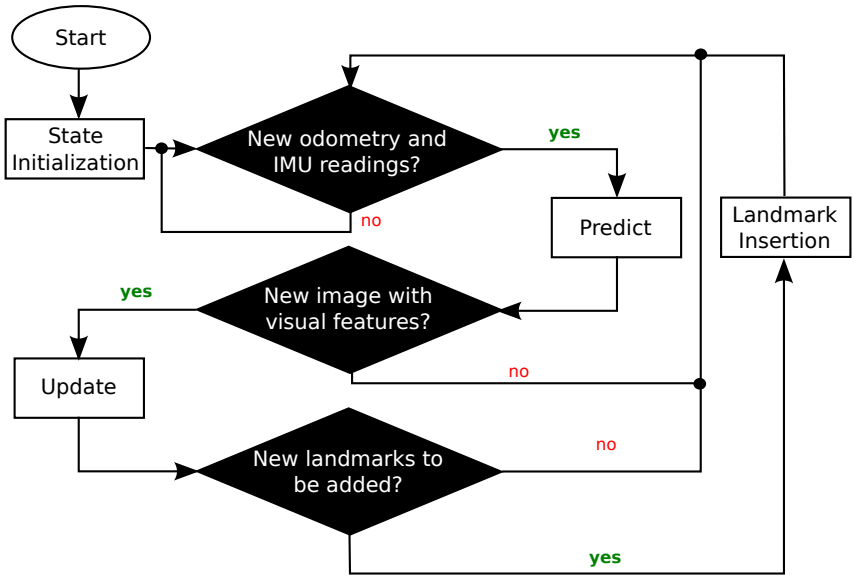


Figure 2.2: EKF flowchart

2.1.3 Inverse Depth Parametrization

EKF demands a close proximity between the current state and the linearization point. While it can be assumed that the robot pose has small changes between two consecutive EKF instants, drastic changes over observation estimates can turn EKF unstable. To overcome this problem, a more linear parametrization for the observation model is proposed by Civera et al [12], called Inverse Depth Parametrization, which is applied for a monocular Visual SLAM. In this work, however, this parametrization is applied for both monocular and stereo visual observations.

Using this parametrization, a landmark y_i in state is represented by

$$y_i = \left(o_{xi}\ o_{y_i}\ o_{zi}\ \theta_i\ \phi_i\ p_i\right)^\top, \tag{2.19}$$

where o_i is the observation point, θ_i and ϕ_i are the azimuth and elevation of the semi-ray that crosses both the o_i and the landmark in the world frame, and p_i is the inverse of the euclidean distance from o_i to the landmark.

To convert a landmark i in XYZ parametrization, y_i^{XYZ}, to the inverse depth counterpart, y_i, an observation point must be chosen from

$$o_i \in \mathbb{R}^3 \setminus \{y_i^{XYZ}\}. \tag{2.20}$$

For this work, o_i will always correspond to the camera location in the world frame where landmark i was first observed. Having o_i, one can define the vector h_i,

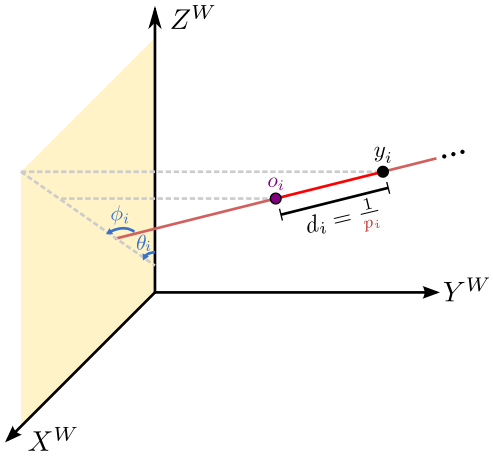


Figure 2.3: Inverse Depth Parametrization for landmark y_i ($d_i = |y_i^{XYZ} - o_i|$).

$$h_i = y_i^{XYZ} - o_i \tag{2.21}$$

and compute the remaining parameters,

$$\begin{pmatrix} \theta_i \\ \phi_i \\ p_i \end{pmatrix} = \begin{pmatrix} arctan(h_{xi}, h_{zi}) \\ arctan(-h_{y_i}, \sqrt{{h_{xi}}^2 + {h_{zi}}^2}) \\ \frac{1}{|h_i|} \end{pmatrix}. \tag{2.22}$$

Conversely,

$$y_i^{XYZ} = o_i + \frac{1}{p_i} m_i \qquad \text{where} \qquad m_i = \begin{pmatrix} \cos\phi_i \, \sin\theta_i \\ -\sin\phi_i \\ \cos\phi_i \, \cos\theta_i \end{pmatrix}. \tag{2.23}$$

2.1.4 Motion Model

For the Motion Model, it is assumed that the robot moves in a 3D space and uses an IMU and odometry to help predict its location over time. This model allows the EKF to predict the state transition from two consecutive instants t and $t+1$. If no IMU nor odometry are used, a random walk approach can alternatively be applied at a cost of more inaccuracies to the estimates.

To help model the robot movement over time, two extra frames are defined for the robot for each instant t. Figure 2.4 shows a possible distribution for all new frames plus camera frame given by the state.

1. **Body frame**, $(X^{B_t}, Y^{B_t}, Z^{B_t})$: it represents the robot body pose at instant t. The spatial translation from B_t to B_{t+1} in the world frame, $r_{B_{t+1}}^{B_t}$, can be obtained using a simple

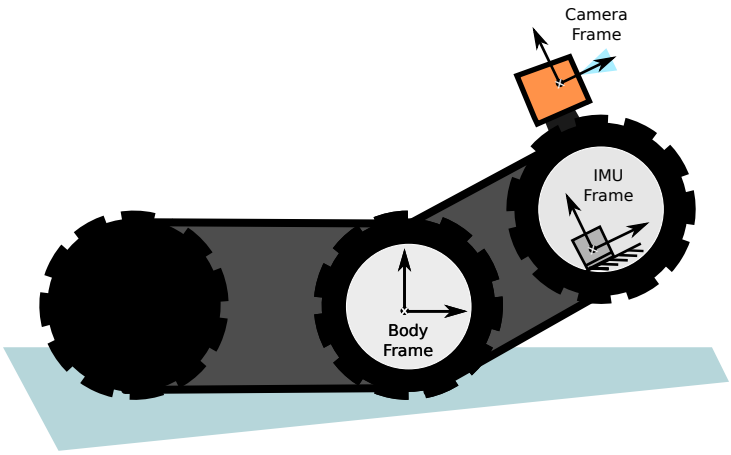


Figure 2.4: A possible Body, IMU and Camera frames configuration for a robot.

odometry motion model and by integrating rotary encoders information from each wheel $(r_\tau^{right}, r_\tau^{left})$. As EKF is solved iteratively in a discrete way, each instant t corresponds to a time τ_t. Knowing the time difference between instants t and $t+1$, $\Delta\tau_{(t,t+1)}$,

$$\Delta\tau_{(t,t+1)} = \tau_{t+1} - \tau_t, \tag{2.24}$$

both linear motions from each wheel can be modelled with addictive uncorrelated zero-mean gaussian noise,

$$\begin{cases} r_\tau^{right} = r_{t+1}^{right} - r_t^{right} + r_t^{\epsilon\, right}, & r_t^{\epsilon\, right} \sim \mathcal{N}(0, (\Delta\tau_{(t,t+1)}\, \sigma_{odo}^{right})^2) \\ r_\tau^{left} = r_{t+1}^{left} - r_t^{left} + r_t^{\epsilon\, left}, & r_t^{\epsilon\, left} \sim \mathcal{N}(0, (\Delta\tau_{(t,t+1)}\, \sigma_{odo}^{left})^2) \end{cases} \tag{2.25}$$

For instance, assuming a two-wheel robot with movement $r_{B_{t+1}}^{B_t}$ coplanar to the plane $(ZoX)^{B_t}$,

$$r_t^{odo} = r_{B_{t+1}}^{B_t} = T_\tau^{odo} \begin{pmatrix} \sin(\alpha_\tau^{odo}) \\ 0 \\ \cos(\alpha_\tau^{odo}) \end{pmatrix}, \tag{2.26}$$

$$\alpha_\tau^{odo} = \frac{r_\tau^{right} - r_\tau^{left}}{L^{odo}}, \qquad T_\tau^{odo} = \frac{r_\tau^{right} + r_\tau^{left}}{2}, \tag{2.27}$$

where L^{odo} is the size of the line segment that crosses both wheels and the robot frame origin. Unfortunately, a simple two-wheeled motion model for movement is not accurate enough for tracked wheel robots to grant more kinectic friction with inclined surfaces and stairs, varying L^{odo} with unmeasured external factors over time. As such, since this model fuses information from an IMU, it will not consider the resulting body frame orientation

from odometry. Instead, it will use IMU readings, as explained below.

2. **IMU frame**, $(X^{I_t}, Y^{I_t}, Z^{I_t})$: it represents the IMU pose at instant t. The angular velocity w^{imu} can be modelled through the IMU gyroscopes,

$$w_t^{imu} = w_t^{gyro} + w_t^{bias} + w_t^{\epsilon}, \tag{2.28}$$

where w_t^{gyro} is the angular velocity retrieved from the IMU, w_t^{bias} is the bias error normally associated with most IMUs (if the IMU uses optical or MEMS technology and is calibrated, it can be assumed no w_t^{bias} for some period of time [20]) and w_t^{ϵ} is a normally distributed error with zero-mean,

$$w_t^{\epsilon} \sim \mathcal{N}(0, \sigma_{imu}^2). \tag{2.29}$$

Knowing the time difference between instants t and $t+1$, $\Delta\tau_{(t,t+1)}$, and assuming constant velocity w_t^{imu}, one can compute the spatial rotation from frame I_t to I_{t+1} using quaternion notation (q_t^{imu}) [19],

$$q_{t_w}^{imu} = \cos(2\,|w_t^{imu}|\,\Delta\tau_{(t,t+1)}), \tag{2.30}$$

$$o_t^{sin} = \frac{\sin(2\,|w_t^{imu}|\,\Delta\tau_{(t,t+1)})}{|w_t^{imu}|} \qquad \begin{pmatrix} q_{t_x}^{imu} & q_{t_y}^{imu} & q_{t_z}^{imu} \end{pmatrix}^{\top} = o_t^{sin}\, w_t^{imu}, \tag{2.31}$$

where $|w_t^{imu}|$ is the L2-norm of w_t^{imu}.

This model assumes an IMU installed at any part of the robot, given that all transformations from the IMU to the robot camera and body frames are well known for each instant. It can be retrieved through servo feedback and proper encoders. Using homogeneous transformation matrices, the state transition can be computed as

$$T_{C_{t+1}}^{W} = T_{C_t}^{W}\,[T_{I_t}^{C_t}\,T_{I_{t+1}}^{I_t}\,[T_{I_{t+1}}^{C_{t+1}}]^{-1}], \tag{2.32}$$

where both $T_{I_t}^{C_t}$ and $T_{I_{t+1}}^{C_{t+1}}$ are well known transforms from the camera frame to the IMU frame in their respective instants. Regarding $T_{I_{t+1}}^{I_t}$,

$$T_{I_{t+1}}^{I_t} = T_{B_t}^{I_t}\,T_{B_{t+1}}^{B_t}\,[T_{B_{t+1}}^{I_{t+1}}]^{-1}, \tag{2.33}$$

$$r_{I_{t+1}}^{I_t} = r_{B_t}^{I_t} + R_{B_t}^{I_t}\,r_t^{odo} - R_{q_t^{imu}}\,r_{B_{t+1}}^{I_{t+1}} \tag{2.34}$$

while the rotation $R_{I_{t+1}}^{I_t}$ is available by converting q_t^{imu} to an orthogonal rotation matrix,

$$R_{I_{t+1}}^{I_t} = R_{q_t^{imu}}, \tag{2.35}$$

So, to calculate the state transition from two consecutive instants t and $t+1$,

$$\begin{cases} r_{t+1} = r_t + R_{q_t} \left(r_{B_t}^{C_t} + R_{B_t}^{C_t} \, r_t^{odo} - R_{I_t}^{C_t} \, R_{q_t^{imu}} \left[R_{I_{t+1}}^{C_{t+1}} \right]^\top r_{B_{t+1}}^{C_{t+1}} \right) \\ q_{t+1} = q_t \times (q_t^{cam} \times q_t^{imu} \times q_{t+1}^{*cam}) \end{cases} \tag{2.36}$$

where q_t^{cam} and q_{t+1}^{cam} are quaternion representations of $R_{I_t}^{C_t}$ and $\left(R_{I_{t+1}}^{C_{t+1}}\right)^\top$, respectively, and both $T_{B_t}^{C_t}$ and $T_{B_{t+1}}^{C_{t+1}}$ are also well known transforms from the camera frame to the body frame in their respective instants.

When implementing this model for RAPOSA, it can be assumed that both IMU frame and camera frame share the same orientation and $r_{B_t}^{C_t}$ for all time,

$$\forall_t, \quad R_{I_t}^{C_t} = I_{(3\times3)}, \quad r_{B_t}^{C_t} = r^{inc} \tag{2.37}$$

and due to a linear actuator installed on RAPOSA (refer to subsection 3.1.3), $R_{B_t}^{C_t}$ is given by

$$R_{B_t}^{C_t} = R_t^{inc} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\left(\alpha_t^{inc}\right) & \sin\left(\alpha_t^{inc}\right) \\ 0 & -\sin\left(\alpha_t^{inc}\right) & \cos\left(\alpha_t^{inc}\right) \end{bmatrix} \tag{2.38}$$

where α_t^{inc} is the smallest angular displacement around the X^{O_t} axis between the body frame and the camera frame orientation.

The state transition from two consecutive instants t and $t+1$ for RAPOSA simplifies to

$$\begin{cases} r_{t+1} = r_t + R_{q_t} \, r_t^{motion} \\ q_{t+1} = q_t \times q_t^{imu} \end{cases} \tag{2.39}$$

where

$$r_t^{motion} = r^{inc} + R_t^{inc} \, r_t^{odo} - R_{q_t^{imu}} \, r^{inc}. \tag{2.40}$$

A jacobian matrix F_t is needed to compute the predict step in EKF, as stated in equation (2.5). The jacobian is given by

$$F_t = \left[\begin{array}{cc|c} I & 0 & \\ \left.\frac{\partial r_{t+1}}{\partial q_t}\right|_{\mu_t,u_t} & \left.\frac{\partial q_{t+1}}{\partial q_t}\right|_{\mu_t,u_t} & 0 \\ \hline 0 & & I \end{array}\right] \tag{2.41}$$

.

where, from equation (2.39),

$$\left.\frac{\partial r_{t+1}}{\partial q_t}\right|_{\mu_t,u_t} = 2\left[\begin{array}{c|c|c|c} M_w\, r_t^{motion} & M_x\, r_t^{motion} & M_y\, r_t^{motion} & M_z\, r_t^{motion} \end{array}\right], \tag{2.42}$$

$$\begin{aligned} M_w &= \begin{bmatrix} q_{w_t} & -q_{z_t} & q_{y_t} \\ q_{z_t} & q_{w_t} & -q_{x_t} \\ -q_{y_t} & q_{x_t} & q_{w_t} \end{bmatrix}, \quad M_x = \begin{bmatrix} q_{x_t} & q_{y_t} & q_{z_t} \\ q_{y_t} & -q_{x_t} & -q_{w_t} \\ q_{z_t} & q_{w_t} & -q_{x_t} \end{bmatrix} \\ M_y &= \begin{bmatrix} -q_{y_t} & q_{x_t} & q_{w_t} \\ q_{x_t} & q_{y_t} & q_{z_t} \\ -q_{w_t} & q_{z_t} & -q_{y_t} \end{bmatrix}, \quad M_z = \begin{bmatrix} -q_{z_t} & -q_{w_t} & q_{x_t} \\ q_{w_t} & -q_{z_t} & q_{y_t} \\ q_{x_t} & q_{y_t} & q_{z_t} \end{bmatrix}. \end{aligned} \tag{2.43}$$

and

$$\left.\frac{\partial q_{t+1}}{\partial q_t}\right|_{\mu_t,u_t} = \begin{bmatrix} q_{w_t}^{imu} & -q_{x_t}^{imu} & -q_{y_t}^{imu} & -q_{z_t}^{imu} \\ q_{x_t}^{imu} & q_{w_t}^{imu} & q_{z_t}^{imu} & -q_{y_t}^{imu} \\ q_{y_t}^{imu} & -q_{z_t}^{imu} & q_{w_t}^{imu} & q_{x_t}^{imu} \\ q_{z_t}^{imu} & q_{y_t}^{imu} & -q_{x_t}^{imu} & q_{w_t}^{imu} \end{bmatrix} \tag{2.44}$$

Contrary to equation (2.2), this model assumes no additive error on the output. A proper first order Taylor linearisation is then needed to approximate the model to a linear form in order to use it for the EKF, where

$$Q_t = J_t^Q\, Q'_t\, [J_t^Q]^\top \tag{2.45}$$

where

$$Q'_t = \left[\begin{array}{ccc|c} (\Delta\tau_{(t,t+1)}\,\sigma_{odo}^{left})^2 & 0 & 0 & \\ 0 & (\Delta\tau_{(t,t+1)}\,\sigma_{odo}^{right})^2 & 0 & 0 \\ 0 & 0 & (\sigma_{inc})^2 & \\ \hline & 0 & & (\sigma_{imu})^2\, I_{(3\times 3)} \end{array}\right] \tag{2.46}$$

and

$$J_t^Q = \left[\begin{array}{ccc|c} \left.\frac{\partial r_{t+1}}{\partial r_{odo}^{left}}\right|_{\mu_t,u_t} & \left.\frac{\partial r_{t+1}}{\partial r_{odo}^{right}}\right|_{\mu_t,u_t} & \left.\frac{\partial r_{t+1}}{\partial \alpha_t^{inc}}\right|_{\mu_t,u_t} & \left.\frac{\partial r_{t+1}}{\partial q_t^{imu}}\right|_{\mu_t,u_t} \cdot \left.\frac{\partial q_t^{imu}}{\partial w_t^{imu}}\right|_{\mu_t,u_t} \\ 0 & 0 & 0 & \left.\frac{\partial q_{t+1}}{\partial q_t^{imu}}\right|_{\mu_t,u_t} \cdot \left.\frac{\partial q_t^{imu}}{\partial w_t^{imu}}\right|_{\mu_t,u_t} \\ \hline \multicolumn{4}{c}{0} \end{array}\right] \tag{2.47}$$

The jacobian matrix Q'_t encompasses error variance from both left and right odometry readings, $\Delta\tau_{(t,t+1)}\,\sigma_{odo}^{left}$ and $\Delta\tau_{(t,t+1)}\,\sigma_{odo}^{right}$ (equation (2.25)), the inclination arm, σ_{inc}, and angular velocity readings (equation (2.28)). The partial derivatives related to the odometry and inclination are given by

$$\left.\frac{\partial r_{t+1}}{\partial r_{odo}^{left}}\right|_{\mu_t,u_t} = R_{q_t}\,R^{inc}\begin{pmatrix} \frac{1}{2}\,\sin(\alpha_\tau^{odo}) - \frac{T_\tau^{odo}}{L^{odo}}\cos(\alpha_\tau^{odo}) \\ 0 \\ \frac{1}{2}\,\cos(\alpha_\tau^{odo}) + \frac{T_\tau^{odo}}{L^{odo}}\sin(\alpha_\tau^{odo}) \end{pmatrix}, \tag{2.48}$$

$$\left.\frac{\partial r_{t+1}}{\partial r_{odo}^{right}}\right|_{\mu_t,u_t} = R_{q_t}\,R^{inc}\begin{pmatrix} \frac{1}{2}\,\sin(\alpha_\tau^{odo}) + \frac{T_\tau^{odo}}{L^{odo}}\cos(\alpha_\tau^{odo}) \\ 0 \\ \frac{1}{2}\,\cos(\alpha_\tau^{odo}) - \frac{T_\tau^{odo}}{L^{odo}}\sin(\alpha_\tau^{odo}) \end{pmatrix}, \tag{2.49}$$

and

$$\left.\frac{\partial r_{t+1}}{\partial \alpha_t^{inc}}\right|_{\mu_t,u_t} = R_{q_t}\begin{pmatrix} 0 \\ -T_\tau^{odo}\,\cos(\alpha_\tau^{odo})\,\cos(\alpha_t^{inc}) \\ T_\tau^{odo}\,\cos(\alpha_\tau^{odo})\,\sin(\alpha_t^{inc}) \end{pmatrix} \tag{2.50}$$

The partial derivatives related to the IMU readings are

$$\left.\frac{\partial q_{t+1}}{\partial q_t^{imu}}\right|_{\mu_t,u_t} = \begin{bmatrix} q_{w_t} & -q_{x_t} & -q_{y_t} & -q_{z_t} \\ q_{x_t} & q_{w_t} & -q_{z_t} & q_{y_t} \\ q_{y_t} & q_{z_t} & q_{w_t} & -q_{x_t} \\ q_{z_t} & -q_{y_t} & q_{x_t} & q_{w_t} \end{bmatrix} \tag{2.51}$$

$$\left.\frac{\partial r_{t+1}}{\partial q_t^{imu}}\right|_{\mu_t,u_t} = -2\,R_{q_t}\left[\begin{array}{c|c|c|c} M_w^{imu}\,r^{inc} & M_x^{imu}\,r^{inc} & M_y^{imu}\,r^{inc} & M_z^{imu}\,r^{inc} \end{array}\right], \tag{2.52}$$

$$
\begin{aligned}
M_w^{imu} &= \begin{bmatrix} q_{w_t}^{imu} & -q_{z_t}^{imu} & q_{y_t}^{imu} \\ q_{z_t}^{imu} & q_{w_t}^{imu} & -q_{x_t}^{imu} \\ -q_{y_t}^{imu} & q_{x_t}^{imu} & q_{w_t}^{imu} \end{bmatrix}, \quad M_x^{imu} = \begin{bmatrix} q_{x_t}^{imu} & q_{y_t}^{imu} & q_{z_t}^{imu} \\ q_{y_t}^{imu} & -q_{x_t}^{imu} & -q_{w_t}^{imu} \\ q_{z_t}^{imu} & q_{w_t}^{imu} & -q_{x_t}^{imu} \end{bmatrix} \\
M_y^{imu} &= \begin{bmatrix} -q_{y_t}^{imu} & q_{x_t}^{imu} & q_{w_t}^{imu} \\ q_{x_t}^{imu} & q_{y_t}^{imu} & q_{z_t}^{imu} \\ -q_{w_t}^{imu} & q_{z_t}^{imu} & -q_{y_t}^{imu} \end{bmatrix}, \quad M_z^{imu} = \begin{bmatrix} -q_{z_t}^{imu} & -q_{w_t}^{imu} & q_{x_t}^{imu} \\ q_{w_t}^{imu} & -q_{z_t}^{imu} & q_{y_t}^{imu} \\ q_{x_t}^{imu} & q_{y_t}^{imu} & q_{z_t}^{imu} \end{bmatrix}.
\end{aligned} \tag{2.53}
$$

and, from both equations (2.30) and (2.31),

$$
\left.\frac{\partial q_t^{imu}}{\partial w_t^{imu}}\right|_{\mu_t, u_t} = \begin{bmatrix} -\Delta\tau_{(t,t+1)}\, o_t^{sin}\, (w_t^{imu})^\top \\ (\Delta\tau_{(t,t+1)}\, q_{w_t}^{imu} - o_t^{sin})\, \frac{w_t^{imu}\,(w_t^{imu})^\top}{|w_t^{imu}|^2} \;-\; o_t^{sin}\, I_{(3\times 3)} \end{bmatrix} \tag{2.54}
$$

2.1.5 Observation Model

The Observation Model adopted for this approach, from equation (2.6), computes z_t, a vector with all n features correspondent to each landmark in state, given the camera pose at instant t,

$$
z_t = \left({z_{1t}}^\top \; ... \; {z_{nt}}^\top \right)^\top. \tag{2.55}
$$

Assuming that the camera focal point is located over the camera frame origin and shares the same orientation, this model is composed of two steps:

1. For each landmark y_i in state, compute a directional vector in the camera frame that points from the camera position to the landmark position. All landmarks within the interval $\mathfrak{H}^C$ result in the same semi-ray for the same camera pose, and therefore, share the same visual observation. The interval $\mathfrak{H}^C$ in the camera frame can be parametrized as

$$
\mathfrak{H}^C = \{ R_{q_t^*}\, j\, (o_i - r_t + \frac{1}{p_i}\, m_i) \mid j \in [0, +\infty[\}, \tag{2.56}
$$

where $R_{q_t^*}$ is the rotation matrix that rotates the world frame to the camera frame (from q_t conjugate, q_t^*).

A possible directional vector that can be easily computed is h_{it}^C, defined as,

$$
h_{it}^C = R_{q_t^*}\, h_{it}, \qquad \text{where} \qquad h_{it} = (p_i\, (o_i - r_t) + m_i). \tag{2.57}
$$

Choosing h_{it}^C from $\mathfrak{H}^C$ allows the observation model to measure landmarks with infinite depth, since

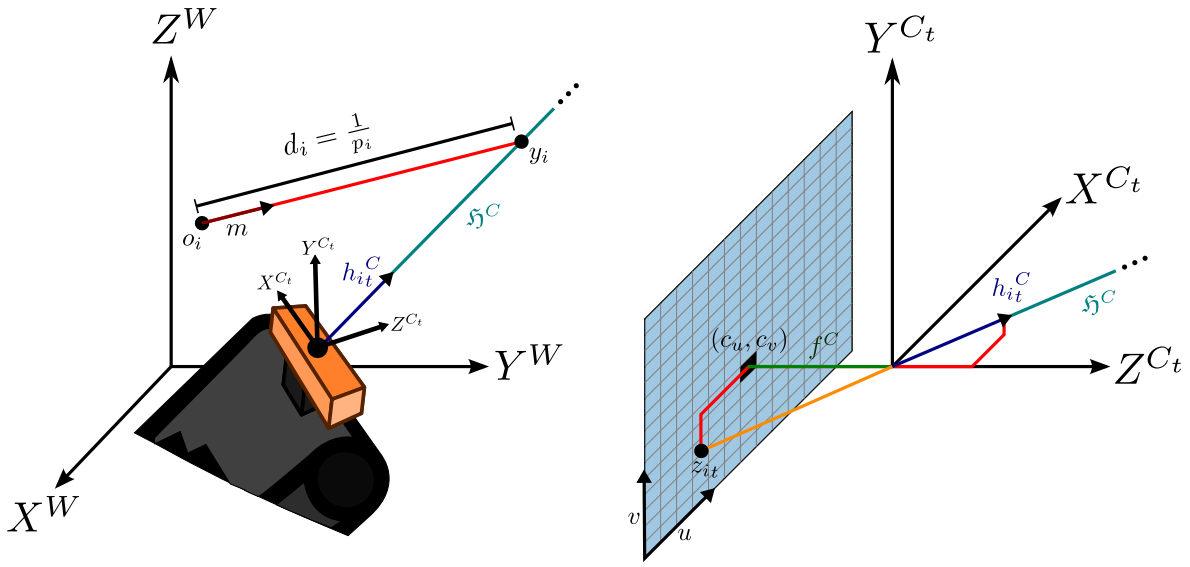


Figure 2.5: Observation Model for landmark y_i.

$$p_i = 0: \qquad h_{it}^C = R_{q_t^*}\, m_i. \tag{2.58}$$

2. Using the Pinhole Camera Model, for each landmark i situated in front of the camera, project h_{it}^C on the camera's image plane that is orthogonal to the camera's optical axis. Knowing the focal length, f_C, the scale factors relating pixel coordinates to meters in the image plane, m_u and m_v, and the location of the principal point in pixels, c_u and c_v, the directional vector h_{it}^C can be directly mapped onto pixel coordinates, z_{it}, by

$$\begin{pmatrix} z_{uit} \\ z_{vit} \end{pmatrix} = \begin{pmatrix} c_u \\ c_v \end{pmatrix} - \frac{f_C}{h_{zi\,t}^C} \begin{pmatrix} m_u\, h_{xi\,t}^C \\ m_v\, h_{yi\,t}^C \end{pmatrix} + \delta_{it} \qquad \text{where} \qquad \delta_{it} \sim \mathcal{N}(0, \sigma_{pixel}^2\, I_{(2\times 2)}). \tag{2.59}$$

It would be more accurate to round z_{it} to their nearest integers, modelling the discretization of visual information to the image output. However, this model would lose its continuity and would no longer be differential for all the domain. Instead, it is assumed that the model does not perform any discrete operation but introduces an error associated with the approximation. This error is defined in conformity with all aforementioned EKF conditions, being an added zero-mean uncorrelated multivariate gaussian noise.

Note also that the image output is a cropped region from the camera's image plane. As such, all landmarks that are projected outside of the image plane margins or with negative depth are not visible. This evaluation will be used further on section 2.3.

The pinhole model assumes a single camera with no lenses, nor aperture radius. It does not model any type of image distortion or blur present in every camera. For this thesis, information retrieved for observation analysis passed through a correction process using camera's proprietary

software before being used by the EKF, returning an undistorted image with known intrinsic parameters, while maintaining a wide visual range. This software also rectifies each pair of stereo images, by projecting them to a common image plane [21]. If no software correction is available, distortion can be compensated with proper models using distortion parameters intrinsic to the camera, retrieved through calibration methods.

An horizontal stereo camera is used in this thesis to acquire image data from two different sources. Since all images are properly rectified, a given pair of features from both cameras only correspond to the same landmark if they both share the same horizontal axis. This rectification also results in a pair of images with the same size and intrinsic parameters.

A set of coordinate frames, $(X^{CL_t}, Y^{CL_t}, Z^{CL_t})$ and $(X^{CR_t}, Y^{CR_t}, Z^{CR_t})$, are defined for the left and right camera, respectively. After the rectification process, both right and left camera frames share the same orientation as the camera frame and are displaced by b along the X^{C_t} axis.

To simplify the formalism, a parameter k^{LR} is introduced, where

$$k^{LR} = \begin{cases} 0, & \text{from left (L) camera} \\ 1, & \text{from right (R) camera} \end{cases} \tag{2.60}$$

Both directional vectors from left and right cameras, $h_{i_t}^{CL}$ and $h_{i_t}^{CR}$, can easily computed from $h_{i_t}^{C}$,

$$h_{i_t}^{CL/CR} = R_{q_t^*}\left(p_i\left(o_i - \left(r_t + (-1)^{k^{LR}} R_{q_t}\,\bar{b}\right)\right) + m_i\right) = h_{i_t}^{C} - (-1)^{k^{LR}}\, h'_{i_t} \tag{2.61}$$

where

$$h'_{i_t} = p_i\,\bar{b} \qquad \text{and} \qquad \bar{b} = \begin{pmatrix} b & 0 & 0 \end{pmatrix}^\top. \tag{2.62}$$

With the pinhole model, one can either model an observation from both cameras,

$$z_{i_t}^{Stereo} = \begin{pmatrix} z_{u_{i_t}}^{L} \\ z_{u_{i_t}}^{R} \\ z_{v i_t} \end{pmatrix} = \begin{pmatrix} z_{u_{i_t}} \\ z_{u_{i_t}} \\ z_{v i_t} \end{pmatrix} - \frac{f_C\, m_u\, p_i}{h_{zi}^{C}{}_t} \begin{pmatrix} -b \\ b \\ 0 \end{pmatrix}, \tag{2.63}$$

or from left or right camera only,

$$z_{i_t}^{Left} = \begin{pmatrix} z_{u_{i_t}}^{L} \\ z_{v i_t} \end{pmatrix} \qquad z_{i_t}^{Right} = \begin{pmatrix} z_{u_{i_t}}^{R} \\ z_{v i_t} \end{pmatrix}. \tag{2.64}$$

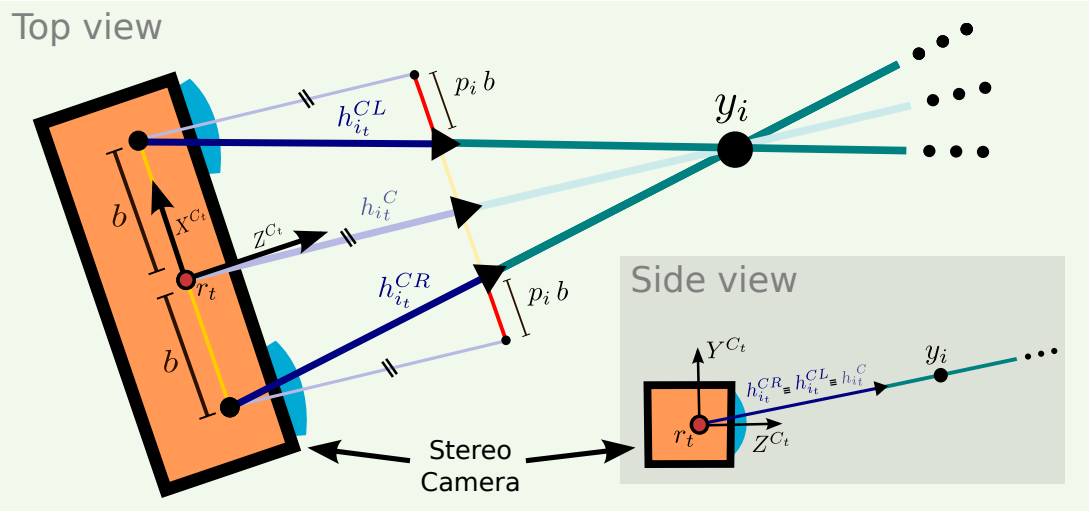


Figure 2.6: Horizontal stereo camera representation.

From equation (2.7) and (2.10), a jacobian H_t is needed to compute the update step from EKF,

$$H_t = \begin{bmatrix} H_{1_t} & \cdots & H_{i_t} & \cdots & H_{n_t} \end{bmatrix}^\top, \tag{2.65}$$

where

$$H_{i_t} = \begin{bmatrix} \frac{\partial z_{i_t}}{\partial r_t}\Big|_{\bar{\mu}_t, z_{i_t}} & \frac{\partial z_{i_t}}{\partial q_t}\Big|_{\bar{\mu}_t, z_{i_t}} & 0 & \dots & 0 & \frac{\partial z_{i_t}}{\partial y_i}\Big|_{\bar{\mu}_t, z_{i_t}} & 0 & \dots & 0 \end{bmatrix} \tag{2.66}$$

and

$$\frac{\partial z_{i_t}}{\partial y_i}\Big|_{\bar{\mu}_t, z_{i_t}} = \begin{bmatrix} \frac{\partial z_{i_t}}{\partial o_i}\Big|_{\bar{\mu}_t, z_{i_t}} & \frac{\partial z_{i_t}}{\partial \theta_i}\Big|_{\bar{\mu}_t, z_{i_t}} & \frac{\partial z_{i_t}}{\partial \psi_i}\Big|_{\bar{\mu}_t, z_{i_t}} & \frac{\partial z_{i_t}}{\partial p_i}\Big|_{\bar{\mu}_t, z_{i_t}} \end{bmatrix}. \tag{2.67}$$

To compute the partial derivatives related to the robot state, from equation (2.57),

$$\frac{\partial z_{i_t}}{\partial r_t}\Big|_{\bar{\mu}_t, z_{i_t}} = \frac{\partial z_{i_t}}{\partial h_{i_t}^C}\Big|_{\bar{\mu}_t, h_{i_t}^C} \frac{\partial h_{i_t}^C}{\partial r_t}\Big|_{\bar{\mu}_t, z_{i_t}} \quad \text{and} \quad \frac{\partial z_{i_t}}{\partial q_t}\Big|_{\bar{\mu}_t, z_{i_t}} = \frac{\partial z_{i_t}}{\partial h_{i_t}^C}\Big|_{\bar{\mu}_t, h_{i_t}^C} \frac{\partial h_{i_t}^C}{\partial q_t}\Big|_{\bar{\mu}_t, z_{i_t}}. \tag{2.68}$$

For the partial derivative related to the robot position,

$$\frac{\partial h_{i_t}^C}{\partial r_t}\Big|_{\bar{\mu}_t, z_{i_t}} = -p_i\, R_{q_t^*}. \tag{2.69}$$

As for the partial derivative related to the robot orientation, knowing the relation between

a quaternion and its conjugate from equation (2.16),

$$\left.\frac{\partial h^C_{i_t}}{\partial q_t}\right|_{\bar{\mu}_t, z_{i_t}} = 2 \left[\begin{array}{c|c|c|c} M^*_w \, h_{it} & -M^*_x \, h_{it} & -M^*_y \, h_{it} & -M^*_z \, h_{it} \end{array} \right], \tag{2.70}$$

where

$$\begin{aligned} M^*_w &= \begin{bmatrix} q^*_{wt} & -q^*_{zt} & q^*_{yt} \\ q^*_{zt} & q^*_{wt} & -q^*_{xt} \\ -q^*_{yt} & q^*_{xt} & q^*_{wt} \end{bmatrix}, \quad M^*_x = \begin{bmatrix} q^*_{xt} & q^*_{yt} & q^*_{zt} \\ q^*_{yt} & -q^*_{xt} & -q^*_{wt} \\ q^*_{zt} & q^*_{wt} & -q^*_{xt} \end{bmatrix}, \\ M^*_y &= \begin{bmatrix} -q^*_{yt} & q^*_{xt} & q^*_{wt} \\ q^*_{xt} & q^*_{yt} & q^*_{zt} \\ -q^*_{wt} & q^*_{zt} & -q^*_{yt} \end{bmatrix}, \quad M^*_z = \begin{bmatrix} -q^*_{zt} & -q^*_{wt} & q^*_{xt} \\ q^*_{wt} & -q^*_{zt} & q^*_{yt} \\ q^*_{xt} & q^*_{yt} & q^*_{zt} \end{bmatrix}. \end{aligned} \tag{2.71}$$

To compute the partial derivatives related to the correspondent landmark state, applying the chain rule in higher dimensions,

$$\left.\frac{\partial z_{i_t}}{\partial y_i}\right|_{\bar{\mu}_t, z_{i_t}} = \left.\frac{\partial z_{i_t}}{\partial h^C_{i_t}}\right|_{\bar{\mu}_t, h^C_{i_t}} \left.\frac{\partial h^C_{i_t}}{\partial y_i}\right|_{\bar{\mu}_t, z_{i_t}} + \left.\frac{\partial z_{i_t}}{\partial h'_{i_t}}\right|_{\bar{\mu}_t} \left.\frac{\partial h'_{i_t}}{\partial y_i}\right|_{\bar{\mu}_t} \tag{2.72}$$

where

$$\left.\frac{\partial h^C_{i_t}}{\partial y_i}\right|_{\bar{\mu}_t, z_{i_t}} = R_{q^*_t} \left[\begin{array}{c|cc|c} p_i \, I_{(3\times 3)} & \cos\psi_i \cos\theta_i & -\sin\psi_i \cos\theta_i & o_i - r_t \\ & 0 & -\cos\psi_i & \\ & -\cos\psi_i \sin\theta_i & -\sin\psi_i \sin\theta_i & \end{array} \right] \tag{2.73}$$

and

$$\left.\frac{\partial h'_{i_t}}{\partial y_i}\right|_{\bar{\mu}_t, z_{i_t}} = \begin{bmatrix} 0 & \cdots & 0 & \bar{b} \end{bmatrix} \tag{2.74}$$

Finally, for a monocular observation,

$$\left.\frac{\partial z_{i_t}}{\partial h^C_{i_t}}\right|_{\bar{\mu}_t, h^C_{i_t}} = \frac{f^C}{h^C_{z_{i_t}}} \begin{bmatrix} -m_u & 0 & m_u \frac{h^C_{x_{i_t}} - (-1)^{k^{LR}} h'_{i_t}}{h^C_{z_{i_t}}} \\ 0 & -m_v & m_v \frac{h^C_{y_{i_t}}}{h^C_{z_{i_t}}} \end{bmatrix}, \tag{2.75}$$

$$\left.\frac{\partial z_{i_t}}{\partial h'_{i_t}}\right|_{\bar{\mu}_t} = \frac{f^C}{h^C_{z_{i_t}}} \begin{bmatrix} (-1)^{k^{LR}} m_u & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{2.76}$$

and for a stereo observation,

$$\left.\frac{\partial z_{i_t}}{\partial h^C_{i_t}}\right|_{\bar{\mu}_t, h^C_{i_t}} = \frac{f^C}{h^C_{z_{i_t}}} \begin{bmatrix} -m_u & 0 & m_u \frac{h^C_{x_{i_t}} - h'_{i_t}}{h^C_{z_{i_t}}} \\ -m_u & 0 & m_u \frac{h^C_{x_{i_t}} + h'_{i_t}}{h^C_{z_{i_t}}} \\ 0 & -m_v & m_v \frac{h^C_{y_{i_t}}}{h^C_{z_{i_t}}} \end{bmatrix} \tag{2.77}$$

$$\left.\frac{\partial z_{i_t}}{\partial h'_{i_t}}\right|_{\bar{\mu}_t} = \frac{f^C}{h^C_{z_{i_t}}} \begin{bmatrix} m_u & 0 & 0 \\ -m_u & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{2.78}$$

2.1.6 Feature Initialization

Over time, visual observations are made and new landmarks are attached to the state from observed visual features. Many criteria can be used to establish when new landmarks should be inserted and how many. For instance, one can add a new landmark every time a visual feature is observed that does not correspond to any landmark in state, but it proves to be computationally ineffective as it fills the state in a short time if no landmark removal procedure is performed.

Assuming the usage of the stereo camera depicted in figure 2.6, one can acquire monocular features either from the left or from the right camera. Also, some features acquired from both cameras correspond to the same landmark, resulting in a stereo feature. Depending on whether the new landmark in state results from a monocular feature or from a stereo feature, two different initializations are introduced:

1. **From a monocular observation:** If a new landmark y_n is to be attached to the state with $n-1$ landmarks from a feature ${z_n^{Left}}_t$ or ${z_n^{Right}}_t$ alone, first a directional vector for the respective camera frame is computed using the Pinhole Camera Model from equation (2.59).

$${h_n^{CL/CR}}_t = \begin{pmatrix} \frac{1}{f_C\, m_u} \left(c_u - z^{L/R}_{u_{n_t}}\right) \\ \frac{1}{f_C\, m_v} \left(c_v - z_{v_{n_t}}\right) \\ 1 \end{pmatrix} \tag{2.79}$$

Having R_{q_t} from q_t in state and knowing that both left and right camera frames share the same orientation from the robot state, the directional vector can be related to the world frame by

$$h_{n_t} = R_{q_t}\, {h_n^{CL/CR}}_t \tag{2.80}$$

and

$$\begin{pmatrix} o_n \\ \theta_n \\ \phi_n \\ p_n \end{pmatrix} = \begin{pmatrix} r_t + (-1)^{k^{LR}} R_{q_t} \bar{b} \\ arctan(h_{xnt}, h_{znt}) \\ arctan(-h_{y_{nt}}, |h_{xznt}|) \\ p_{initial} \end{pmatrix} \tag{2.81}$$

where

$$|h_{xznt}| = \sqrt{(h_{xnt})^2 + (h_{xnt})^2} \tag{2.82}$$

It is impossible to gain depth information from just one observation, thus an initial arbitrary value $p_{initial}$ serves as an initial estimation for the inverse depth given enough uncertainty. This parametrization is approximately linear along the corresponding semi-ray, allowing the EKF to sustain big errors for the depth estimation.

2. **From a stereo observation:** If a new landmark y_n is to be attached to the state with $n-1$ landmarks from a stereo pair of features, $z_{n_t}^{Stereo}$, first a directional vector from the camera frame is computed using the Pinhole Camera Model from equation (2.59). Since both left and right camera frames share the same distance from the camera frame but in opposite directions, the directional vector can be calculated as

$$h_{n_t}^C = \begin{pmatrix} \frac{1}{f_C\, m_u} \left(c_u - \frac{1}{2}(z_{u_{n_t}}^L + z_{u_{n_t}}^R)\right) \\ \frac{1}{f_C\, m_v} \left(c_v - z_{v_{n_t}}\right) \\ 1 \end{pmatrix}. \tag{2.83}$$

In the same fashion as equation (2.80),

$$h_{n_t} = R_{q_t}\, h_{n_t}^C \tag{2.84}$$

and

$$\begin{pmatrix} o_n \\ \theta_n \\ \phi_n \\ p_n \end{pmatrix} = \begin{pmatrix} r_t \\ arctan(h_{xnt}, h_{znt}) \\ arctan(-h_{y_{nt}}, |h_{xznt}|) \\ p_{epipolar} \end{pmatrix}. \tag{2.85}$$

where $p_{epipolar}$ can be computed using epipolar geometry [22],

$$p_{epipolar} = \frac{z_{u_{n_t}}^L - z_{u_{n_t}}^R}{2\, b\, f_C\, m_u\, |h_{n_t}^C|}. \tag{2.86}$$

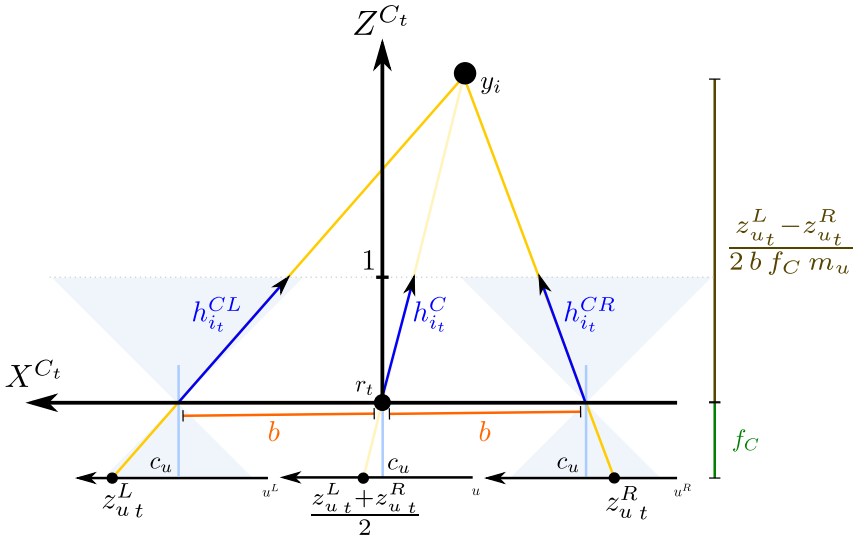


Figure 2.7: Top view of camera frame for stereo vision of a landmark y_i.

For a state with $n-1$ landmarks, s_t estimation (as a normal distribution) can be updated by adding a new landmark y_n initial estimate [12],

$$\mu_t^{new} = \begin{pmatrix} \mu_t \\ y_n \end{pmatrix}, \qquad P_t^{new} = J_t^{new} \begin{bmatrix} P_t & 0 \\ 0 & A_{nt} \end{bmatrix} (J_t^{new})^\top \tag{2.87}$$

where

$$J_t^{new} = \left[\begin{array}{ccccc|c} I & & & & & 0 \\ \hline \left.\frac{\partial y_n}{\partial r_t}\right|_{\mu_t, z_{n_t}} & \left.\frac{\partial y_n}{\partial q_t}\right|_{\mu_t, z_{n_t}} & 0 & \cdots & 0 & J_{y_n}^{new} \end{array}\right]. \tag{2.88}$$

From equations (2.81) and (2.85), by applying the chain rule, the partial derivatives related to the robot state are

$$\left.\frac{\partial y_n}{\partial r_t}\right|_{\mu_t, z_{n_t}} = \begin{bmatrix} I_{(3\times 3)} \\ 0_{(3\times 3)} \end{bmatrix} \qquad \text{and} \qquad \left.\frac{\partial y_n}{\partial q_t}\right|_{\mu_t, z_{n_t}} = \begin{bmatrix} \left.\frac{\partial o_n}{\partial q_t}\right|_{\mu_t, z_{n_t}} \\ \left.\frac{\partial \theta_n}{\partial h_{n_t}}\right|_{\mu_t, h_{n_t}} \cdot \left.\frac{\partial h_{n_t}}{\partial q_t}\right|_{z_{n_t}} \\ \left.\frac{\partial \phi_n}{\partial h_{n_t}}\right|_{\mu_t, h_{n_t}} \cdot \left.\frac{\partial h_{n_t}}{\partial q_t}\right|_{z_{n_t}} \\ 0_{(1\times 4)} \end{bmatrix}, \tag{2.89}$$

where

$$\left.\frac{\partial \theta_n}{\partial h_{n_t}}\right|_{\mu_t, h_{n_t}} = \frac{1}{|h_{xznt}|^2} \begin{pmatrix} h_{znt} \\ 0 \\ -h_{xnt} \end{pmatrix}^\top, \qquad \left.\frac{\partial \phi_n}{\partial h_{n_t}}\right|_{\mu_t, h_{n_t}} = \frac{1}{|h_{nt}|^2 \, |h_{xznt}|} \begin{pmatrix} h_{xnt} \, h_{y_{nt}} \\ -|h_{xznt}|^2 \\ h_{y_{nt}} \, h_{znt} \end{pmatrix}^\top \tag{2.90}$$

and, having the conversion from quaternion to a rotation matrix given by equation (2.18), if the landmark is extracted from a monocular feature,

$$\left.\frac{\partial o_n}{\partial q_t}\right|_{\mu_t, z_{n_t}} = 2\, b\, (-1)^{k^{LR}} \begin{bmatrix} q_{wt} & q_{xt} & -q_{y_t} & -q_{zt} \\ q_{zt} & q_{y_t} & q_{xt} & q_{wt} \\ -q_{y_t} & q_{zt} & -q_{wt} & q_{xt} \end{bmatrix} \tag{2.91}$$

or, if the landmark is extracted from a stereo pair of features,

$$\left.\frac{\partial o_n}{\partial q_t}\right|_{\mu_t, z_{n_t}} = 0_{(3\times 4)} \tag{2.92}$$

From equations (2.80), (2.84),

$$\left.\frac{\partial h_{n_t}}{\partial q_t}\right|_{z_{n_t}} = 2 \left[\; M_w \, h_{nt} \; \middle| \; M_x \, h_{nt} \; \middle| \; M_y \, h_{nt} \; \middle| \; M_z \, h_{nt} \; \right], \tag{2.93}$$

where M_w, M_x, M_y and M_z are from equation (2.43).

Regarding $J_{y_n}^{new}$ and A_{nt}, they assume a different structure depending if the landmark is extracted from a monocular or stereo source. For the monocular case,

$$A_{nt} = \begin{bmatrix} \sigma_{pixel}^2 \, I_{(2\times 2)} & 0 \\ 0 & \sigma_{i.d.}^2 \end{bmatrix} \qquad \text{and} \qquad J_{y_n} = \begin{bmatrix} \left.\frac{\partial y_n}{\partial z_n^{Mono}{}_t}\right|_{\mu_t, z_{n_t}^{Mono}} & \left.\frac{\partial y_n}{\partial p_n}\right|_{\mu_t, z_{n_t}^{Mono}} \end{bmatrix}. \tag{2.94}$$

where $\sigma_{i.d.}^2$ is the variance associated to the initial inverse depth (which is assumed to be high enough to cover all uncertainty). The partial derivatives are

$$\left.\frac{\partial y_n}{\partial p_n}\right|_{\mu_t, z_{n_t}^{Mono}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}^\top \tag{2.95}$$

and

$$\left.\frac{\partial y_n}{\partial p_n}\right|_{\mu_t, z_{n_t}^{Mono}} = \begin{bmatrix} 0_{(3\times 2)} \\ \left.\frac{\partial \theta_n}{\partial h_{n_t}}\right|_{\mu_t, h_{n_t}} \cdot \left.\frac{\partial h_{n_t}}{\partial z_{n_t}^{Mono}}\right|_{\mu_t, z_{n_t}^{Mono}} \\ \left.\frac{\partial \psi_n}{\partial h_{n_t}}\right|_{\mu_t, h_{n_t}} \cdot \left.\frac{\partial h_{n_t}}{\partial z_{n_t}^{Mono}}\right|_{\mu_t, z_{n_t}^{Mono}} \\ 0_{(1\times 2)} \end{bmatrix}, \tag{2.96}$$

where

$$\left.\frac{\partial h_{n_t}}{\partial z_{n_t}}\right|_{\mu_t, z_{n_t}^{Mono}} = \frac{-1}{f^C} R_{q_t} \begin{bmatrix} \frac{1}{m_u} & 0 \\ 0 & \frac{1}{m_v} \\ 0 & 0 \end{bmatrix}. \tag{2.97}$$

For the stereo case,

$$A_{nt} = \sigma_{pixel}^2 I_{(3\times 3)} \qquad \text{and} \qquad J_{y_n} = \left.\frac{\partial y_n}{\partial z_{n_t}^{Stereo}}\right|_{\mu_t, z_{n_t}^{Stereo}} \tag{2.98}$$

$$\left.\frac{\partial y_n}{\partial p_n}\right|_{\mu_t, z_{n_t}^{Stereo}} = \begin{bmatrix} 0_{(3\times 3)} \\ \left.\frac{\partial \theta_n}{\partial h_{n_t}}\right|_{\mu_t, h_{n_t}} \cdot \left.\frac{\partial h_{n_t}}{\partial z_{n_t}^{Stereo}}\right|_{\mu_t, z_{n_t}^{Stereo}} \\ \left.\frac{\partial \psi_n}{\partial h_{n_t}}\right|_{\mu_t, h_{n_t}} \cdot \left.\frac{\partial h_{n_t}}{\partial z_{n_t}^{Stereo}}\right|_{\mu_t, z_{n_t}^{Stereo}} \\ \left.\frac{\partial p_n}{\partial z_{n_t}^{Stereo}}\right|_{\mu_t, z_{n_t}^{Stereo}} \end{bmatrix}, \tag{2.99}$$

where

$$\left.\frac{\partial h_{n_t}}{\partial z_{n_t}^{Stereo}}\right|_{z_{n_t}^{Stereo}} = \frac{-1}{f^C} R_{q_t} \begin{bmatrix} \frac{1}{2\,m_u} & \frac{1}{2\,m_u} & 0 \\ 0 & 0 & \frac{1}{m_v} \\ 0 & 0 & 0 \end{bmatrix}. \tag{2.100}$$

and finally, from equation (2.86),

$$\left.\frac{\partial p_n}{\partial z_{n_t}^{Stereo}}\right|_{\mu_t, h_{n_t}} = p_{epipolar} \begin{pmatrix} \frac{h_{x_{n_t}}^C}{2\,f^C\,m_u\,|h_{n_t}^C|^2} + \frac{1}{z_{u_{n_t}}^L - z_{u_{n_t}}^R} \\ \frac{h_{x_{n_t}}^C}{2\,f^C\,m_u\,|h_{n_t}^C|^2} - \frac{1}{z_{u_{n_t}}^L - z_{u_{n_t}}^R} \\ \frac{h_{y_{n_t}}^C}{f^C\,m_v\,|h_{n_t}^C|^2} \end{pmatrix}^\top. \tag{2.101}$$

2.2 Feature Detection, Description and Matching

2.2.1 Introduction

When observing a camera image, one might ask how to use this data as a measurement for algorithms such as SLAM. As explained in subsection 2.1.2, the map is kept in state as a set of natural landmarks. These landmarks are extracted from different visual points in different images, called features. However, not all visual points in an image serve as good features. Their neighbouring region should be as distinctive as possible from other regions. For instance, unique corners and blob-like regions are known as good feature holders. Also, it is important that all features possess an unique identification, invariant to some changes such as scale and rotation and resistant to small distortions. This would allow the SLAM to associate all landmarks in state with their visual counterparts, by associating a descriptor to the landmark and matching it with all observed features from camera images.

In this thesis, each landmark is described by the descriptor from the last visual feature observed and matched. This presents some advantages, such as the ability to match more accurately in continuous images, but also presents problems regarding loop closure. If the landmark is not observed in the same perspective as the last time it was perceived, it will be extremely difficult for the descriptors to match. Also, false positive matches infer wrong descriptors to the landmarks. While it may seem critical, proper measures in the matching process described bellow can reduce *de facto* a number of false positives.

There are many feature detectors and descriptors available in literature that can be used to autonomously identify visual features within an image and extract a descriptor for each feature in such a way that all matched descriptors correspond to the same feature. This thesis covers two feature detectors that require no prior data about the environment: SURF [16] and ORB [17]. Although there are more reliable feature detectors such as SIFT, this work is to be processed in real time, requiring more efficient techniques.

2.2.2 SURF: Speeded Up Robust Features

SURF (Speeded Up Robust Features) is a scale and rotation invariant feature detector and descriptor that extracts blob-like features using integral images I_t, defined from an image I_m by

$$I_t(u,v) = \sum_{\bar{u}=1}^{u} \sum_{\bar{v}=1}^{v} I_m(\bar{u},\bar{v}). \tag{2.102}$$

With I_t, one can compute the integral of any rectangular section (u, v, du, dv) within I_m easily:

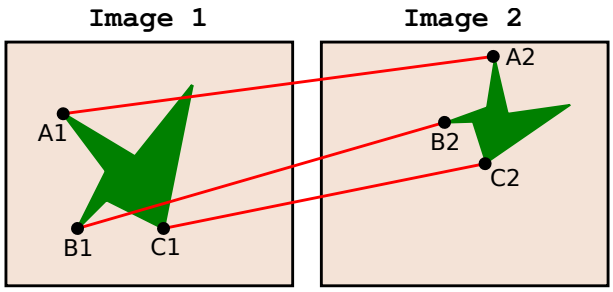


Figure 2.8: Visual depiction of matched feature correspondences.

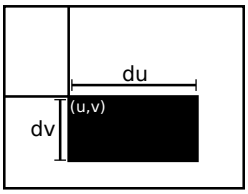


Figure 2.9: The integral within the black section of Im is given by just three operations.

$$\sum_{\bar{u}=u}^{u+du}\sum_{\bar{v}=v}^{v+dv} I_m(\bar{u},\bar{v}) = I_t(u+du,v+dv) + I_t(u,v) - I_t(u+du,v) - I_t(u,v+dv). \tag{2.103}$$

Since SURF uses the determinant of the Hessian matrix to identify all interest points, equation (2.103) becomes fruitful when computing image convolutions over box filters representing second-order Gaussians in a rough way, allowing a scale-space analysis for feature detection. After computing all determinants, the local maximums above a certain threshold are selected as features. The determinant of the Hessian Matrix can also be used as a good measure for the feature's strength.

For the descriptors, it is first assigned an orientation to each feature using Haar-wavelets, thus making SURF invariant to rotations. Knowing the orientation and the scale at which the feature was detected, a neighbour region is selected and the descriptor is computed from it.

Most SURF implementations return each descriptor as a vector of 64 or 128 floats. While 128 sized descriptors are more precise, their computation is slower.

The chances that two descriptors correspond to the same feature is bigger when the euclidean distance between those same descriptors is small. Any matching process that minimizes the euclidean distances between descriptors can be used, but there are always chances of getting bad matches.

To prevent false-positives, SURF calculates the sign of the laplacian of each feature. If two matching candidates have different signs, they will surely not match. Also, one can valid a match only when the best match distance is q times smaller than the second best match to avoid mismatches between different features with similar descriptors (e.g. $q = 0.7$).

Algorithm 1 Matching process for SURF

```
for each landmark in state do
    if landmark is visible then
        m1 ← +∞, m2 ← +∞
        for each feature from current image do
            if landmark.laplacian == feature.laplacian then
                dist ← |landmark.descriptor − feature.descriptor|
                if m1 > dist then
                    m2 ← m1
                    m1 ← dist
                    best_candidate ← feature
                else if m2 > dist then
                    m2 ← dist
                end if
            end if
        end for
        if m1 < q ∗ m2 then
            landmark has a match with best_candidate
        end if
    end if
end for
```

Algorithm 1 presents the matching procedure between landmarks in state and features from when a new image is acquired and processed with SURF.

2.2.3 ORB: Oriented FAST and Rotated BRIEF

ORB (Oriented FAST and Rotated BRIEF) is a rotation-only invariant feature detector and descriptor. Although less reliable than SURF and no scale invariant, it still behaves with great accuracy for small scale changes. While both methods have good performances, ORB is faster but less accurate than SURF regarding scale changes. However, if those changes are considered small, ORB accuracy suffices the SLAM needs. ORB computes FAST features with added orientation information from the intensity centroid. For the descriptors, it uses BRIEF descriptors, calculated from binary intensity tests and rotated using the orientation assigned.

The ORB implementation from Willow Garage in OpenCV returns each descriptor as a set of binary values (32 bytes). A good matching process minimizes the Hamming distance between two descriptors, as presented in algorithm 2.

Also, as with SURF, one can valid a match only when the best match distance is q times smaller than the second best match. Algorithm 3 presents the matching procedure between landmarks in state and features from when a new image is acquired and processed with ORB.

Algorithm 2 Hamming distance for ORB descriptors

```
function DISTANCEORB(descriptorA, descriptorB)
    dist ← 0
    for i = 0 to 32 do
        if descriptorA[i] ≠ descriptorB[i] then
            dist ← dist + 1
        end if
    end for
    return dist
end function
```

Algorithm 3 Matching process for ORB

```
for each landmark in state do
    if landmark is visible then
        m1 ← +∞, m2 ← +∞
        for each feature from current image do
            dist ← distanceORB(landmark.descriptor, feature.descriptor)
            if m1 > dist then
                m2 ← m1
                m1 ← dist
                best_candidate ← feature
            else if m2 > dist then
                m2 ← dist
            end if
        end for
        if m1 < q * m2 then
            landmark has a match with best_candidate
        end if
    end if
end for
```

2.3 Dimensional-bounded Extended Kalman Filter

2.3.1 Introduction

One of the major problems regarding the Extended Kalman Filter is the fact that its computational complexity increases over a quadratic order with the number of entries in the state. As stated on subsection 2.1.2, the state is represented by the camera pose and set of landmarks representing a map. The number of entries in state vector s_t is given by

$$size_{s_t} = 7 + 6\, n_{landmarks} \tag{2.104}$$

where 7 parameters have respect to the robot pose and $n_{landmarks}$ correspond to the number of landmarks represented in state, each having 6 parameters. It has almost double the number of entries if all landmarks were to be represented in XYZ coordinates. It is suggested by Civera et

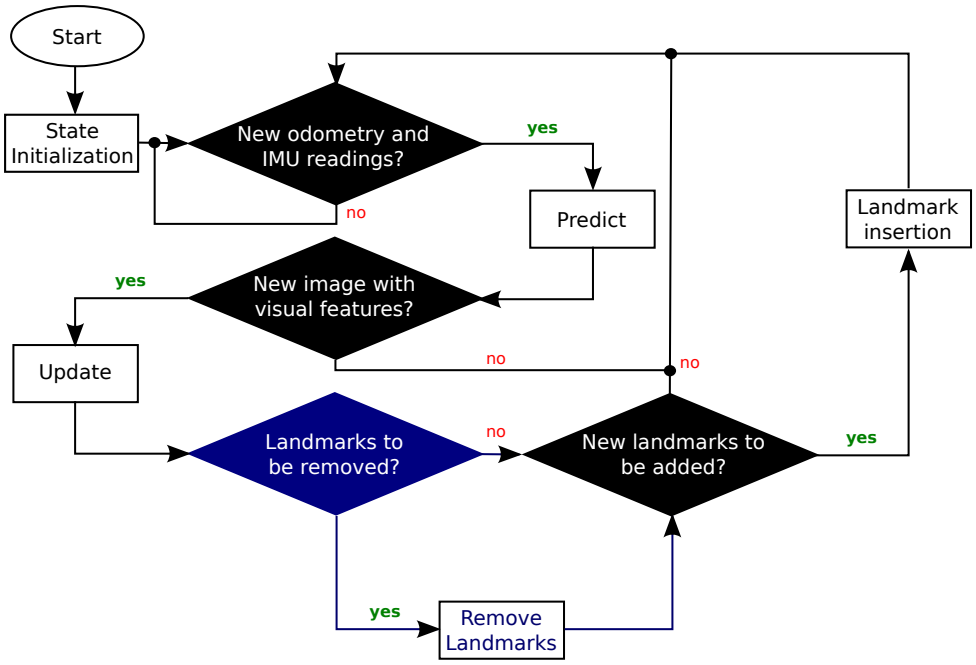


Figure 2.10: DBEKF flowchart

al [12] to convert a landmark's inverse depth parametrization to XYZ in state when the error covariance is low, but it does not interfere with the computational complexity of EKF.

By upper limiting the number of landmarks in state by a value $M_{landmarks}$, EKFs computational complexity becomes upper bounded. However, only upper limiting without any criterion and without removing old landmarks prevents the EKF filter to acquire new features. As such, new landmarks are only added when needed and those state landmarks that are not contributing to reduce the uncertainty should be removed. Since in practice, it is very difficult (although possible) to process loop closure with visual data retrieved from feature detectors when revisited in a different perspective, it is of no priority to keep old landmarks in state. However, it does not have to eliminate any landmarks unless it is needed, and a criterion for when new landmarks are needed must be defined as well.

This thesis introduces a *Dimensional-bounded Extended Kalman Filter* (DBEKF) which encompasses EKF with special criterion for landmarks insertion and removal. For that, a new Landmark Classifier is defined.

The flowchart in figure 2.10 shows how DBEKF is processed, having in blue the differences from normal EKF in figure 2.2.

2.3.2 Landmark Classifier

For the DBEKF, a landmark y_i is said to be *visible* in state s_t, $y_i \in \mathbf{V_{s_t}}$, if it is observable within the field of view of the camera from state s_t. Also, y_i is *detected* at instant t, $y_i \in \mathbf{D_t}$, if the feature detector points out a corresponding feature. In a perfect scenario, assuming that no

landmarks have physical occlusions

$$\mathbf{V_{s_t}} \subset \mathbf{D_t}, \tag{2.105}$$

that is, if the landmark is *visible* then it should be *detected*•. However, feature detectors are far from having a perfect behaviour: descriptors can fail to point out some correspondences and miss features from being detected. This inaccuracies are crucial to classify each landmark's usability in state. Since it is assumed that no landmarks have physical occlusions, a visible but not detected landmark can only represent a failed match.

Since the state in EKF has only gaussian distributions representing landmark pose estimates, a *Temporal Difference Learning* approach [7] is used to predict a measure of the utility, $util_t^i$, at instant t:

$$util_t^i = \begin{cases} G\, util_{t-1}^i + (1-G)\, \mathbf{1}^i_{\mathbf{D_t}} & \text{if} \quad y_i \in \mathbf{V_{s_t}} \\ util_{t-1}^i & \text{else} \end{cases}, \qquad G \in [0,1] \tag{2.106}$$

where G is an adjustable weight factor and the indicator function, $\mathbf{1}^i_{\mathbf{D_{s_t}}}$, is defined for detectability,

$$\mathbf{1}^i_{\mathbf{D_{s_t}}} = \begin{cases} 1 & \text{if} \quad y_i \in \mathbf{D_{s_t}} \\ 0 & \text{else} \end{cases}. \tag{2.107}$$

The initial value for utility is

$$util_0^i = 1. \tag{2.108}$$

From equations (2.106) and using the condition from (2.108), it can be shown that

$$\forall_{G\in[0,1]}\ \forall_t, \quad util_t^i \in [0,1]. \tag{2.109}$$

Regarding the weight factor G, it represents the influence at instant t of $\mathbf{1}^i_{\mathbf{D_{s_t}}}$ over $util_t^i$. The lower G is, higher the influence. If $G = 0$, the utility at instant t assumes the same value as $\mathbf{1}^i_{\mathbf{D_{s_t}}}$. If $G = 1$, the utility stays equal to the initial value $util_0^i$, having no influence from $\mathbf{1}^i_{\mathbf{D_{s_t}}}$.

$$\text{if } G = 1: \qquad \forall_t \quad util_t^i = 1, \tag{2.110}$$

$$\text{if } G = 0: \qquad \forall_t \quad util_t^i = \{0,1\}, \tag{2.111}$$

$$\text{if } G \in]0,1[: \qquad \forall_t \quad util_t^i =]0,1]. \tag{2.112}$$

Algorithm 4 Landmark Removal

for each *landmark* in state **do**
  $k \leftarrow 0$
  if *landmark* is visible **then**
    if *landmark* was detected **then**
      $util_t^i \leftarrow 1 + G * (util_{t-1}^i - 1)$
      $k \leftarrow k + 1$
    else
      $util_t^i \leftarrow G * util_{t-1}^i$
    end if
  end if
end for

$n \leftarrow m_{observations} - k$
for each *landmark* in state **do**
  if $n > 0$ **then**
    landmark is removed
    $n \leftarrow n - 1$
  end if
end for

for each *landmark* in state, **sorted in ascending order of** $util_t$ **do**
  if $util_t^i \leq T$ **or** $p_i \leq 0$ **then**
    landmark is removed
  end if
end for

2.3.3 Landmark Removal

Assuming $M_{landmarks}$ as the maximum number of landmarks imposed by the user to the DBEKF, the Landmark removal procedure is composed of three criteria simultaneously imposed to all landmarks:

1. **Utility Threshold:** Regarding the utility $util_t^i$, one can interpret $util_t^i = 1$ as a maximum score and $util_t^i = 0$ as a minimal score (although it can only be equal to zero if $G = 0$). A simple but effective approach for a landmark removal criterion is that when $util_t^i$ reaches a value bellow T at instant t, it is discarded from the state, where $T \in [0, 1]$.

2. **Negative Inverse Depth:** From the first-order linearisation nature of EKF, it may happen that a landmark estimation in state gets a negative inverse depth. This situation, although not common, can damage the whole process. As such, all features with negative inverse depth are automatically discarded from the state.

3. **Emergency Removal**: Although the aforementioned two criteria are enough for a landmark quality control, the risk of filling the state and not being able to add new, important, landmarks for the observation of new areas maintains. As such, a used defined value $m_{observations}$ is presented, stating the minimal number of matched landmarks per

Algorithm 5 New Landmark Insertion

$n \leftarrow M_{landmarks} - n_{landmarks}$
for each $feature$ in observation, **sorted in descending order of** $strengh$ **do**
  if $n > 0$ **and** did not match any landmark in state **then**
    $state \leftarrow newlandmark$ from $feature$
    $n \leftarrow n - 1$
  end if
end for

observation. Note that

$$m_{observations} \leq M_{landmarks}. \tag{2.113}$$

If only n landmarks were matched and $n < m_{observations}$, the number of landmarks to eliminate should be, at least, of $m_{observations} - n$. Older landmarks are picked for removal.

Algorithm (4) shows in pseudocode how the *Landmark Removal* is processed.

2.3.4 New Landmark Insertion

As of new landmark insertion, the only criterion is to add $n_{newlandmarks}$ new landmarks to the state

$$n_{newlandmarks} = min(M_{landmarks} - n_{landmarks},\ n_{features}), \tag{2.114}$$

where $n_{landmarks}$ is the current number of landmarks in state and $n_{features}$ is the number of features without landmark correspondence given an observation at instant t. Algorithm (5) shows in pseudocode how the *New Landmark Insertion* is processed.

Chapter 3

Implementation

This chapter presents with more detail the grounded *Search and Rescue* (SAR) robot used for this thesis, as well as all work implemented, from system architecture to sensors calibration. Section 3.1 covers RAPOSA-NG specifications, as well as all sensors and actuators from RAPOSA-NG relevant for this thesis, and section 3.2 describes the software and communication architecture implemented on RAPOSA-NG.

3.1 RAPOSA-NG Specifications

3.1.1 Introduction

As stated in subsection 1.2, *IdMind*© upgraded the SAR robot RAPOSA and made it available for commercial purposes as RAPOSA-NG. An unit was acquired by Instituto Superior Técnico for research purposes without any sensors except for encoder and battery sensors readout.

The main body contains all of the electronic and motor boards responsible for the robot control and power management. It has two DC motors for differential drive motion with rotary encoders and one linear actuator with potentiometer to elevate the frontal body. The frontal body includes diverse sensors, some of them used for this thesis: a stereo camera and an *Inertial Measurement Unit* (IMU). Figure 3.1 identifies the main body and the frontal body, and shows RAPOSA-NG layout with some specifications regarding size measures.

3.1.2 Differential Drive Motors with Rotary Encoders

Although RAPOSA-NG uses two pairs of tracked wheels, it works in a similar fashion to a differential drive, as shown in figure 3.2. Each pair of tracks are coupled with a *Maxon*® DC rotor. To control them, a serial message must be sent to the motors microprocessor with values v_{left} and v_{right}, for the left and right tracks, respectively. Both v_{left} and v_{right} values range from -1100 to 1100 and are proportional to the linear velocity from the respective track.

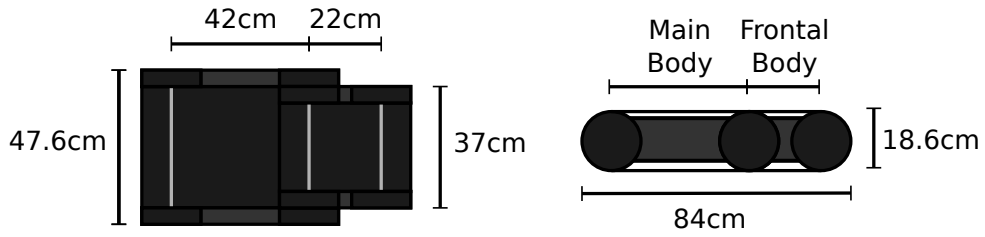


Figure 3.1: RAPOSA-NG size measurements from top view (left) and side view (right)

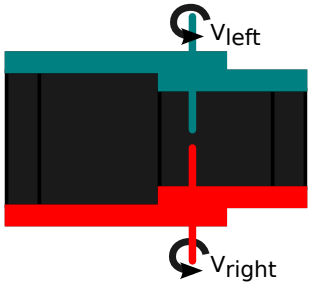


Figure 3.2: RAPOSA-NG differential drive

Each rotor has an incremental rotary encoder attached to it that sends a pulse to the motors microprocessor each time a fixed angular displacement is done. When asked through serial communication, the motors microprocessor returns the number of pulses counted from the last query to the current one. The number of pulses from each encoder, n_{left}^{pulses} and n_{right}^{pulses}, are directly proportional to the distance travelled in meters, r_{left} and r_{right}, by a respective constant K_{left} and K_{right},

$$r_{left} = K_{left}\, n_{left}^{pulses} \qquad \text{and} \qquad r_{right} = K_{right}\, n_{right}^{pulses} \tag{3.1}$$

To get an estimation of K_{left} and K_{right}, a set of pairs of different distances and pulse counts were acquired for each wheel. Since the relationship between pulses and travelled distance is purely linear, a simple linear regression with no constant term is enough to get proper estimates for K_{left} and K_{right}. The values obtained for K_{left} and K_{right} are present in table 3.2.

Table 3.1: Estimated values of K_{left} and K_{right} in meters per pulse

$\mathbf{K_{left}}$	$\mathbf{K_{right}}$
1.61×10^{-5}	1.48×10^{-5}

To have an absolute measure of distance travelled by each wheel from a given time, r_t^{left} and r_t^{right}, the driver used for odometry readings initializes them to zero and increments whenever new measures are acquired by the software driver that communicates via serial with the motors microprocessor.

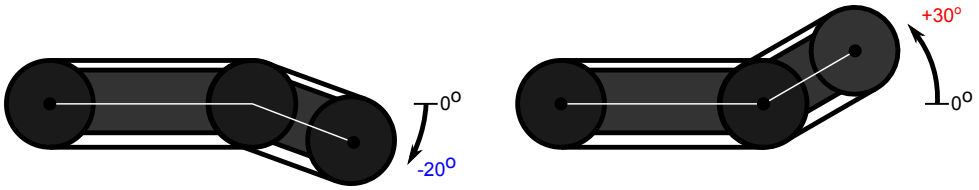


Figure 3.3: RAPOSA-NG inclination arm limits

3.1.3 Linear Actuator with Potentiometer

RAPOSA-NG has a linear actuator from *Firgelli Automations*® that controls, α^{inc}, the shortest angle from the intersection of the planes defined by the two wheel axis from the base body and the two wheel axis from the frontal body. Figure 3.3 depicts the minimum and maximum physical inclination limits in degrees. It also possesses a potentiometer for inclination feedback connected to the motors microprocessor to return a value T^{inc} that is related to α^{inc} by a first-order polynomial function,

$$\alpha^{inc} = A^{inc}\, T^{inc} + b^{inc} \tag{3.2}$$

where A^{inc} and b^{inc} are constant values. In order to obtain A^{inc} and b^{inc}, a linear regression was made from multiple measures of different T^{inc} and hand-measured α^{inc} pairs. The values obtained for A^{inc} and b^{inc} are present in table 3.2.

Table 3.2: Estimated values of A^{inc} and b^{inc}

$\mathbf{A^{inc}}$	$\mathbf{b^{inc}}$
-0.2	129.1

To control the linear actuator, a serial message must be sent to the motors microprocessor with T^{inc}_{des}, a value calculated from the desired inclination α^{inc}_{des} by

$$T^{inc}_{des} = \frac{\alpha^{inc}_{des} - b_{inc}}{A_{inc}}. \tag{3.3}$$

3.1.4 Stereo Camera with Rectification Software

A stereo camera *Bumblebee 2*® from *Point Grey*® was acquired for RAPOSA-NG for vision. It uses IEEE-1394 interface with firewire for communication and is able to process 640x480 color images triggered at same time from both cameras at 48fps. A software library named *Triclops*®, property of *Point Grey*®, uses internal information kept with the stereo camera, regarding the intrinsic parameters from each camera and their baseline, to rectify both cameras and calculate their disparity.

For this thesis, both rectified cameras are used. Most rectification processes reduce each

Figure 3.4: *Bumblebee 2*®(left) and *Microstrain 3DM-GX2*® (right)

camera's field of view in a way that is preferable to use unrectified images and compensate any distortion inside the Extended Kalman Filter (EKF). However, this software allows a proper rectification without losing much field of view, and as such there is no need to add more non-linearities to the EKF. Table 3.3 shows all intrinsic parameters for both cameras with 640×480 rectified images, acquired from the *Bumblebee 2*® using the *Triclops*® library.

Table 3.3: Both rectified intrinsic parameters for both cameras with image size of 640×480 from *Triclops*®

$\mathbf{m_u\, f_C}$	$\mathbf{m_v\, f_C}$	$\mathbf{c_u}$	$\mathbf{c_v}$
285.0663	285.0663	319.3656	254.4078

3.1.5 Inertial Measurement Unit

For an Inertial Measurement Unit (IMU), RAPOSA-NG has a *Microstrain 3DM-GX2*® which uses MEMS sensor technology. It combines a triaxial accelerometer, triaxial gyroscope and triaxial magnetometer on an onboard microprocessor for unbiased, well calibrated measurements of angular velocities, linear accelerations and orientation in different representations. It uses serial communications for data acquisition, as well as writing configuration parameters on an EEPROM inside the IMU.

One of the biggest problems with gyroscope readings comes with the bias associated to each axis. This can prove to be quite problematic to the SLAM if the bias is not included to the EKF state. However, this IMU is able of self calibration as long as neither the IMU or the magnetic map surrounding the IMU is disturbed during the calibration procedure (a proper calibration takes aproximately 10 seconds). Since magnetometer readings can prove to be problematic, e.g. due to some proximity with metal, one can always use only the gyroscope readings for orientation changes measurement and use the magnetometer for calibration purposes only.

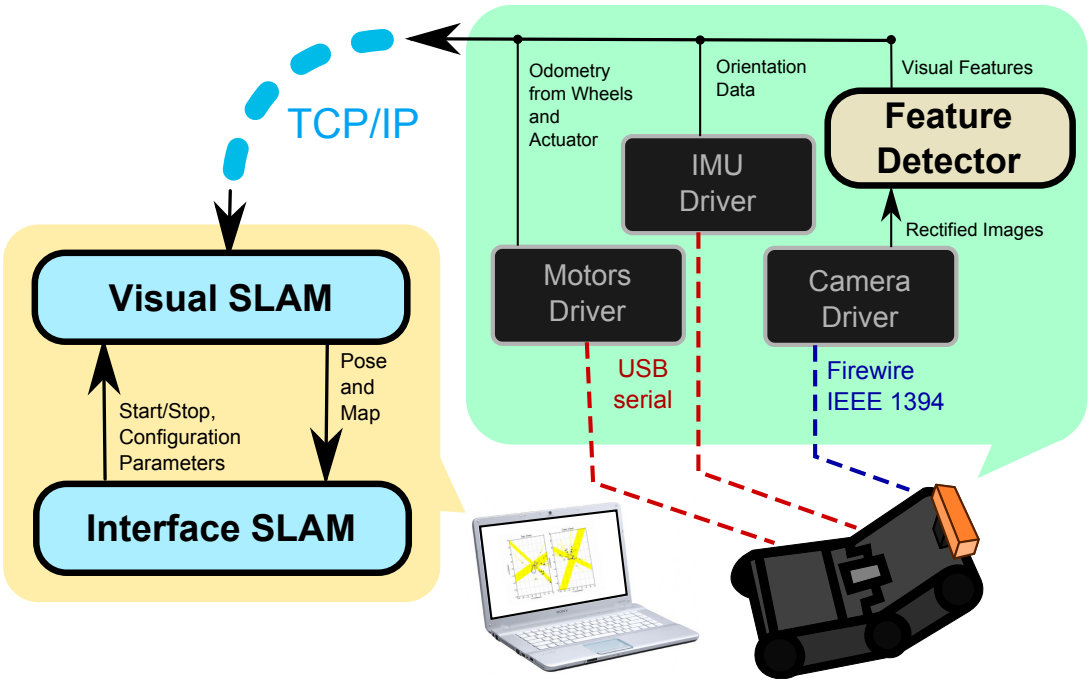


Figure 3.5: Software and Communications Architecture for SLAM on RAPOSA-NG

3.2 System Architecture

The RAPOSA-NG used for this thesis has a motherboard Commell LV-679, a Mini-ATX with an Intel Core 2 Duo 1.6Ghz processor welded. It has two Kingston 1GB 667Mhz DDR2 for a total of 2GB of RAM and a Solid State Drive from Kingston with 64GB. It communicates serial messages to the motors microprocessor through USB, using a FTDI Serial-USB converter.

Figure 3.5 shows how the software and communications architectures needed for SLAM on RAPOSA-NG are implemented. Each gray block represents a driver application and both blue blocks represent all of SLAM software. As for the others:

- *Feature Detector* processes each pair of images retrieved through the Camera Driver from the stereo camera and returns all feature points from either ORB or SURF detector, as well as their respective descriptors.

- *Visual SLAM* is the key application that processes all measurement informations and observations from RAPOSA-NG and runs the SLAM presented in this thesis, returning a new pose and an updated map at each step.

- *Interface SLAM* is responsible for the startup configuration of *Visual SLAM*, where all user-defined parameters are given in a XML format, as well as for monitoring all of its activity. It is done on a different computer that can be connected to RAPOSA-NG either with a LAN cable or wireless.

All software nodes were implemented using C/C++ and Python with the *Robotic Oper-*

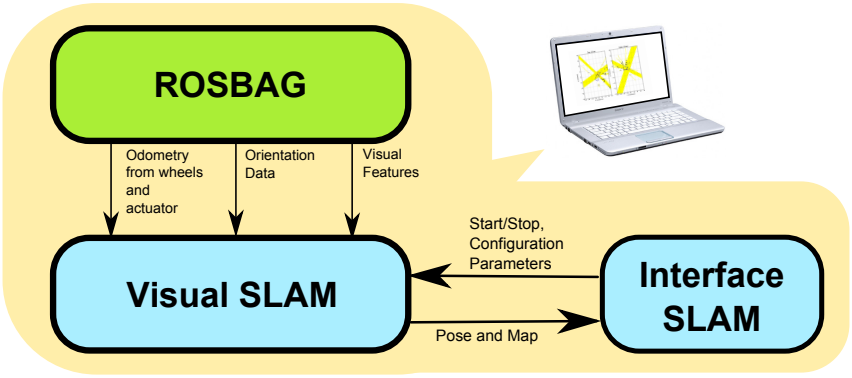


Figure 3.6: Software and Communications Architecture for SLAM using offline data.

ating System (ROS) framework[1], where most data communications use publish–subscribe messaging design and includes a vast repository with different drivers and libraries for free usage (although for this thesis, both Motors drive and Camera drive had to be implemented).

Figure 3.6 shows how the software and communications architectures needed for SLAM using offline datasets are implemented. ROSBAG, for ROS, has a set of tools that allows the recording and playing back of all observation data needed for the SLAM to work. It respects all messages serialization and emulates data transfer times from when it was recording, producing near-real datasets for experimentation.

[1] http://www.ros.org

Chapter 4

Experimental Results

This section presents all experimental results obtained using robot RAPOSA-NG. Section 4.1 introduces this chapter by presenting two datasets used for the test purposes. Section 4.2 shows a comparison between two different feature detectors and descriptors, SURF and ORB, when applied for this *Simultaneous Location and Mapping* (SLAM) algorithm. Section 4.3 shows how each experiment behaves with *Dimensional-bounded EKF* (DBEKF) using monocular only, stereo only and with the novelistic approach of both monocular and stereo features in a hybrid way. Section 4.5 shows DBEKF SLAM regarding the utility weight with two extrema situations. Finally, section 4.6 shows DBEKF executing without *Inertial Measurement Unit* (IMU) and odometry readings, functioning as a random walk for motion model.

4.1 Introduction

All test results presented in this thesis are from two different datasets made with RAPOSA-NG, both recorded in the *Laboratório de Robotica Móvel* from *Instituto Superior Técnico (IST)* at *Campus Alameda*. Both were recorded using ROSBAG application from middleware ROS. Also, the stereo camera is attached to a Pan&Tilt structure in RAPOSA-NG that is not being used for this thesis work but makes the camera sway a little vertically when moving. While slightly observed with the camera, these small camera movements are not caught by the IMU and may incur against the condition presented by the motion model that requires both camera and IMU frames to share the same orientation for all time.

Each dataset contains odometry readings from left track, right track and inclination arm position at 10Hz each, angular velocity readings from IMU at 30Hz, rectified images from both cameras at 15Hz and all features retrieved from image readings using feature detector and descriptor ORB at 15Hz. In order to run and replicate the experiments, the architecture presented in subsection 3.2 and shown in figure 3.6 is used. However, results may slightly vary for the same dataset in different runs. Unless it is said otherwise, all tests performed with DBEKF have an upper bound of $M_{landmarks} = 60$ landmarks in state, an utility weight factor of $G = 0.8$, an

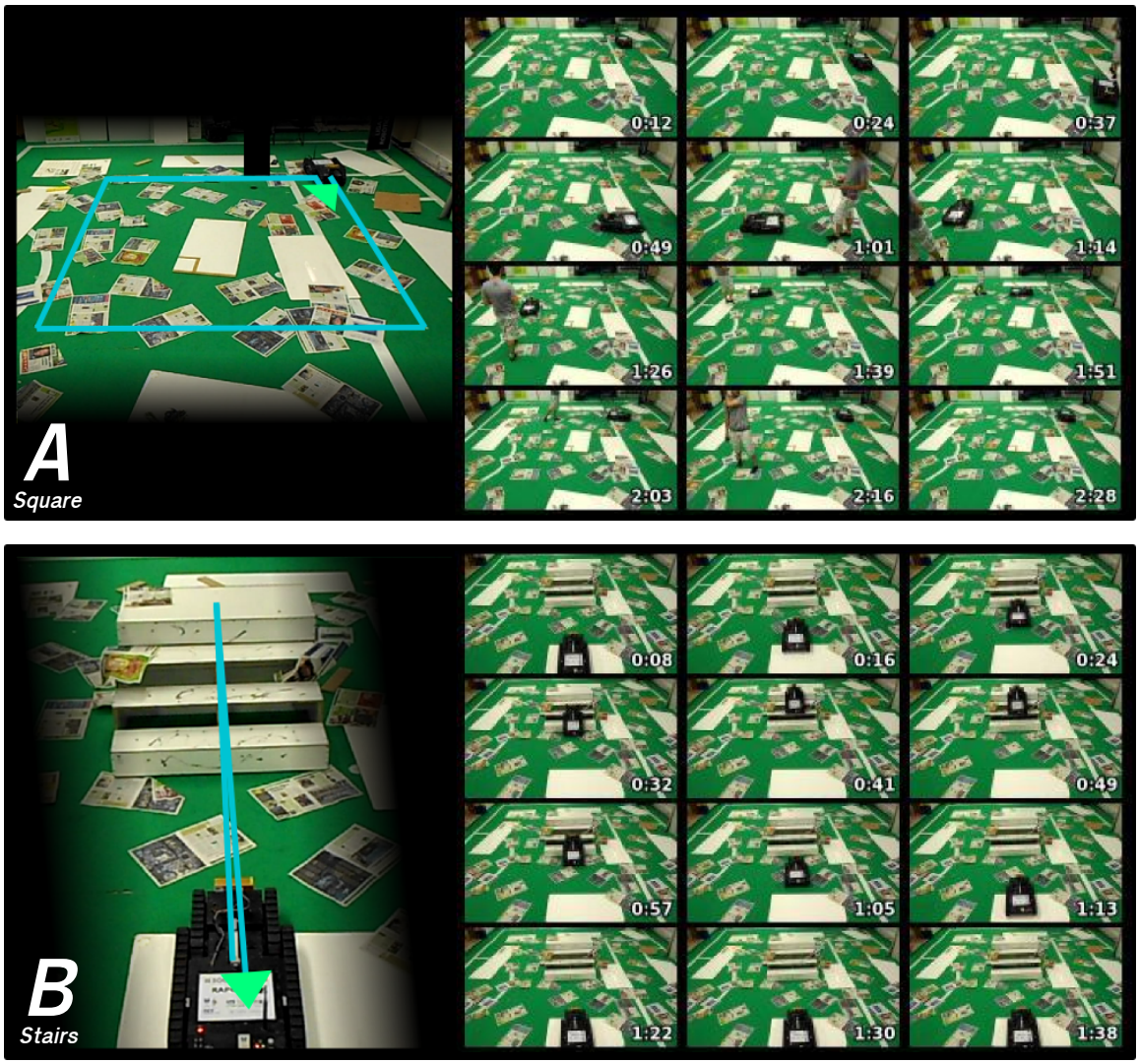


Figure 4.1: Datasets "A" (top) and "B" (bottom). The blue line represents the trajectory travelled by RAPOSA-NG during the experiment, in the direction pointed by the green arrowhead.

utility threshold of $T = 0.01$ and a minimal number of matched landmarks per observation of $M_{landmarks} = 10$.

The datasets are as follows, both depicted in figure 4.1:

- *Dataset "A" - Square Trip*

  On this dataset, RAPOSA-NG performs a near-squared trip of 3×3 meters in a soccer field full of newspaper pages, wooden planks and other sort of objects, simulating debris. Newspapers were scattered around the floor to present a larger spectrum of landmark candidates from feature detection, while messing with odometry readings when turning. The inclination arm angle is zero for all time, and the stereo camera is always parallel to the floor. The robot never stops moving, even when rotating every 90º degrees. Also, it finishes the trajectory facing the same plane from the starting point. This dataset helps to evaluate how the SLAM deals with planar trajectories when performing 3D pose

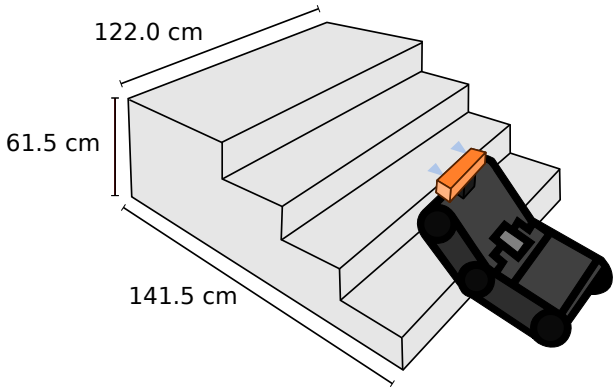


Figure 4.2: Stairs used for dataset "B".

estimation, as well as how the SLAM behaves when, after turning, faces an entirely new plan that requires new observations. It has a duration of 2 minutes and 48 seconds.

- *Dataset "B" - Stairs Trip*

  On this dataset, RAPOSA-NG climbs up and down a set of stairs, placed in the same scenario as dataset "A". As depicted on figure 4.2, the stairs set has 0.615 meters of height. The distance in Z^W from the starting point to the farthest point the robot reaches is of approximately 2.5 meters. Upon reaching the stairs top, RAPOSA-NG drives backwards along the same path until it reaches the starting point again. Stair climbing offers a number of challenges for SLAM: it highly affects odometry readings, it offers a huge amount of feature occlusions while climbing and the robot movement has no smoothness. This dataset has a duration of 1 minute and 38 seconds.

4.2 Comparison between SURF and ORB

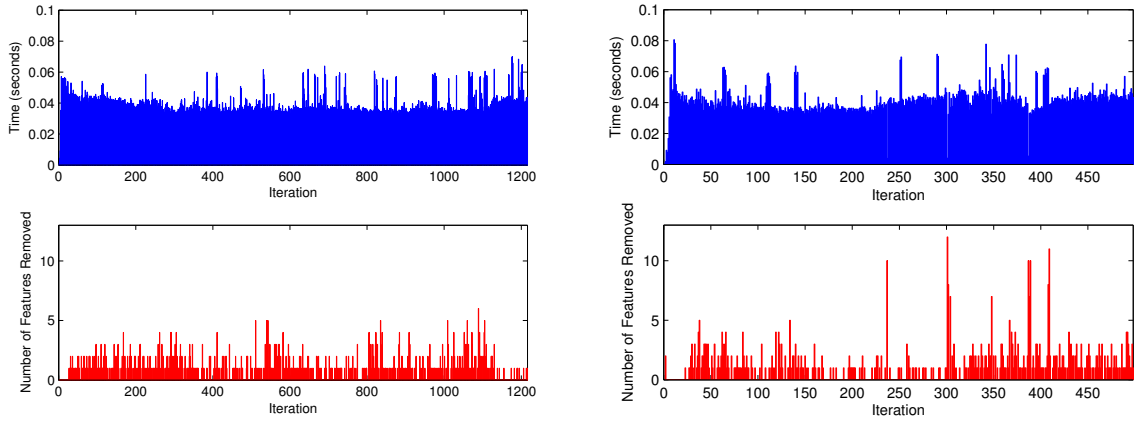
Both ORB and SURF were properly introduced in section 2.2. The speed and performance of those feature detectors is compared by performing a DBEKF with both datasets and using each of them. Both SURF and ORB algorithms used are available from OpenCV image processing libraries in C++ [1].

Table 4.1: Average time and landmarks removed by DBEKF using ORB and SURF with datasets "A" (top) and "B" (bottom).

Dataset "A"	**Time** (average)	**Number of landmarks removed** (average)
ORB	0.015 seconds	1.13 landmarks per iteration
SURF	0.26 seconds	2.27 landmarks per iteration

Dataset "B"	**Time** (average)	**Number of landmarks removed** (average)
ORB	0.013 seconds	1.75 landmarks per iteration
SURF	0.18 seconds	1.79 landmarks per iteration

[1] http://opencv.willowgarage.com/wiki/



(a) Dataset "A" (b) Dataset "B"

Figure 4.3: Time duration (top) and number of removed features (bottom) per DBEKF iteration for datasets "A" (a) and "B" (b) with both monocular and stereo observations.

Table 4.1 presents the average value per iteration of both feature detector processing time and number of landmarks removed by DBEKF using ORB and SURF with datasets "A" and "B". It is clear from processing times that ORB takes much less computational time than SURF. Since one of the DBEKF removal criteria stands for the utility classifier introduced in subsection 2.3.2, where a landmark is said to be useful if it has a corresponding feature match every time it is visible, the fact that the number of landmarks removed from state with ORB is less or similar than with SURF clarifies that ORB shows greater performance over SURF for this application. Note that SURF loses some performance by taking too much time being processed for each iteration, prejudicing EKF and consequent estimates in the process.

For this test, only monocular observations from the left camera were used. If stereo observations were used, the feature detector would take at least double the time per iteration for each pair of stereo observations than with monocular observations. For the remainder of this chapter, all tests will be performed using ORB, as it showed to be faster and more efficient than SURF for this application.

4.3 Computational load and feature removal with DBEKF

As aforementioned in section 2.3.1, by upper limiting the number of landmarks in state by a value $M_{landmarks}$, EKF computational complexity becomes upper bounded as well. If there are enough observations to grant $M_{landmarks}$ in state for estimation at every iteration, the computational load should be constant for all time. This situation happens to all experiments presented on this thesis.

Figure 4.3 shows the time duration and number of removed features per DBEKF iteration with both monocular and stereo observations from datasets "A" and "B", respectively. As

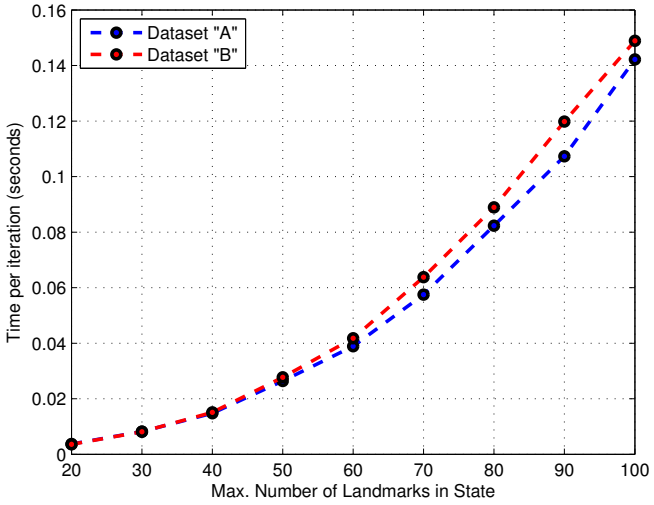


Figure 4.4: Average time per iteration for different upper bounds in DBEKF for datasets "A" and "B".

expected, the computational complexity is near constant for all time during both experiments, presenting some peaks and fluctuations due to new landmark initialization, the number of feature observations per update and other processing tasks unrelated to this software. Regarding dataset "A", one can notice some time intervals where a larger number of features are removed. These intervals happen when the robot finishes rotating 90 degrees and faces a new plane of observations, requiring space for new landmarks in state. It then discards older landmarks in order to acquire new ones. Dataset "B" presents some peaks regarding the number of feature observations as well, where the robot experienced drastic observation changes due to the rough movement of RAPOSA when finishing climbing up or starting to climb down the stairs.

From the average time reading in both experiments, it is clear that the SLAM algorithm fully performs in real time. However, it does not take into account the time needed for feature acquisition using feature detectors such as SURF or ORB. With the system architecture presented in subsection 3.2, one can use processing power from RAPOSA-NG onboard CPU for feature acquisition and use an external computer fully dedicated to SLAM.

Figure 4.4 shows the average time per iteration for different upper bounds in DBEKF with both datasets. As expected, despite the dataset, the time per iteration rises near a cubic order. It is important that, although the computational power becomes constant, a reasonable upper bound is chosen to avoid large time intervals for the EKF that can otherwise reveal linearity problems.

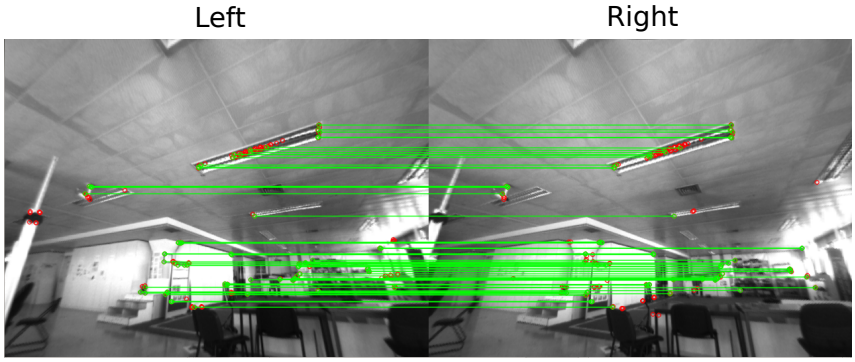


Figure 4.5: Pair of stereo image with visual features. Monocular features are represented with a red dot, while stereo features are represented with a green dot and matched with a green line.

4.4 Mono, Stereo and Hybrid SLAM using DBEKF

Both monocular and stereo observations can be used to correct the *a priori* state estimation in an elegant way using the novelistic approach presented in this thesis. This section compares this solution over monocular only and stereo only solutions. Note that for the monocular only solution, the left camera was chosen arbitrarily for feature acquisition. Using epipolar geometry and feature descriptors, one can identify a stereo feature by matching a pair of monocular features that share the same horizontal axis.

Figure 4.6 shows the number of stereo and monocular features acquired with the stereo camera *Bumblebee2* using ORB for both datasets "A" and "B". While for dataset "A" the stereo acquisition seems constant during all experiment, a huge increase of stereo features (and decrease of monocular features) can be observed during the middle of dataset "B" experiment, while climbing up and down the stairs. During this period, the robot observes the upper part of the scenario, contributing with observations that suffer no perspective problems, unlike the stairs that are in close proximity with the robot. This results suggest the need of coupling both monocular only and stereo observations for state estimation.

Figures 4.7 and 4.8 present SLAM graphical results for datasets "A" and "B" using monocular only, stereo only and both monocular and stereo observations with DBEKF. Figure 4.9 shows the pose covariance trace through time for all the experiences aforementioned. Regarding monocular observations only, both datasets suffer from scalability problems when estimating the trajectory. While it strives to correct the pose and orientation using pallarax changes only, the pose covariance following the robot trajectory tends to increase more over time due to the disparity between odometry measurements and the assumed scale from the observations.

As for stereo observations only experiments, it outperforms SLAM with monocular only in both datasets. Unfortunately, stereo observations seem to be not enough to cover the observations space, and as such some errors in pose estimation may still occur.

Finally, using both monocular and stereo observations, it performs better to stereo ob-

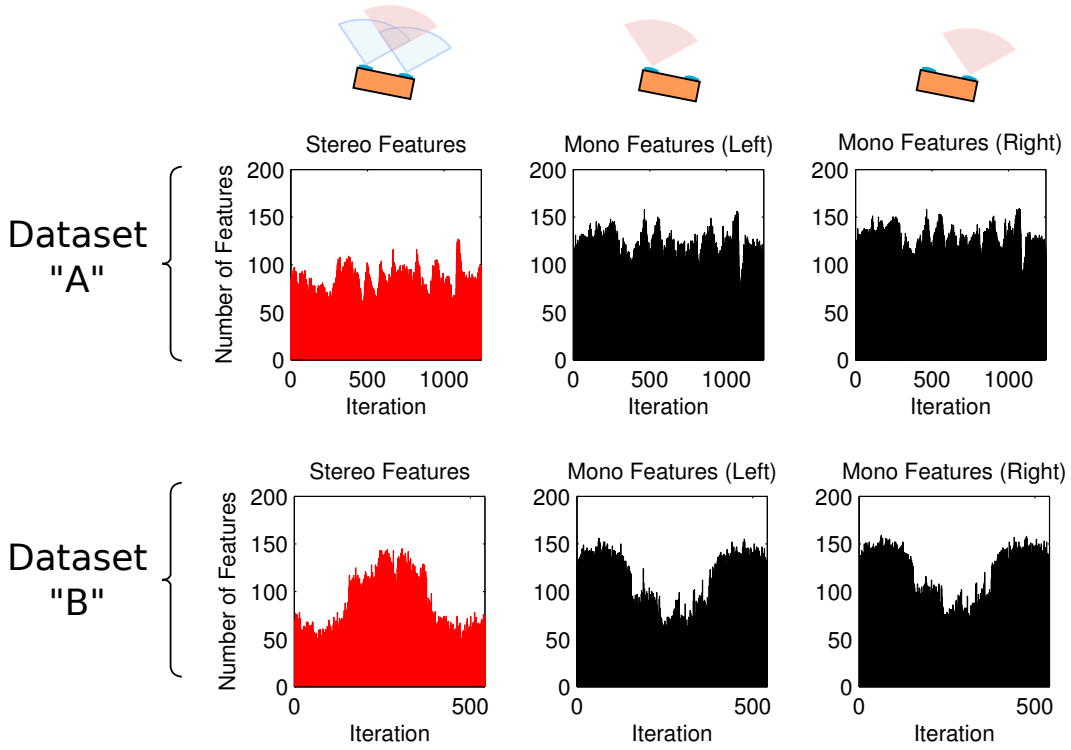


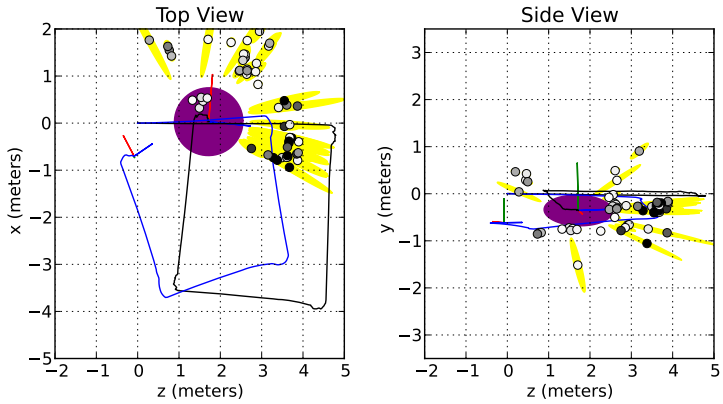
Figure 4.6: Number of stereo and mono features acquired with ORB for each dataset "A" and "B".

servations only by correcting further some trajectory errors. While with dataset "A" the pose covariance trace is always inferior to both monocular and stereo only experiments, with dataset "B" it is inferior to the monocular experiment but similar to the stereo experiment. The number of available features is superior when using both stereo and monocular observations, allowing for a better observation coverage when stereo observations are missing. Also, with this approach, one can use monocular features while suffering no scalability problem, proving to be advantageous over using monocular or stereo observations only.

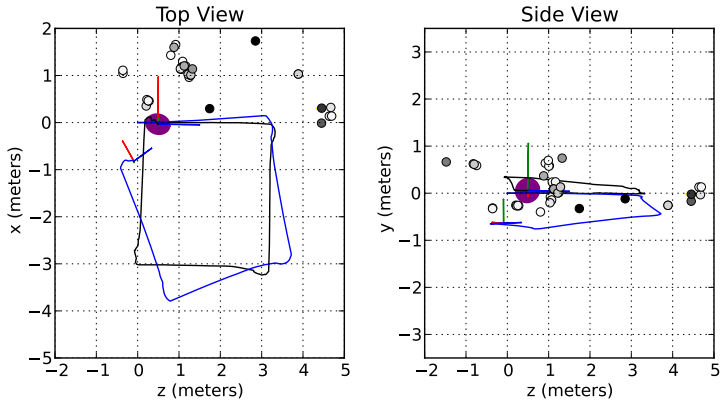
4.5 DBEKF with different parameters

As previously mentioned in section 2.3, the utility weight factor G can vary from 0 to 1. When $G = 0$, a landmark is immediately eliminated if for once it should be visible but is not detected. When $G = 1$ no landmark is removed regardless of its utility. However, the other two criteria are still applied, removing both landmarks with negative inverse depth and older landmarks when required.

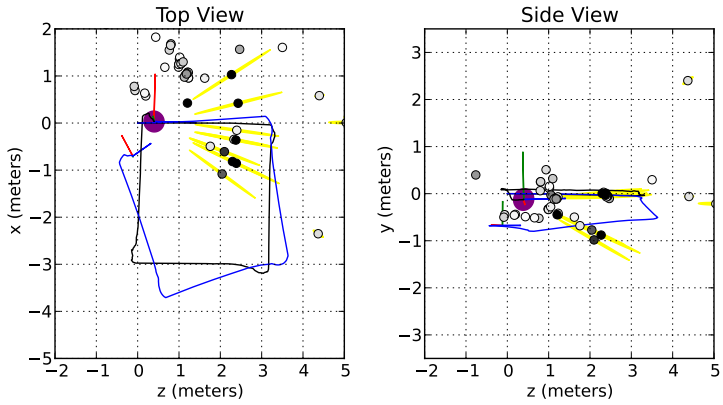
Figures 4.10 and 4.11 present SLAM graphical results for dataset "A" and "B" with $G = 0$ and $G = 1$, respectively, while figure 4.12 shows the pose covariance evolution for those conditions. Regarding test results with $G = 0$, it is clear that the pose covariance rises significantly over time. This is due to the fact that it only takes one simple mismatch or occlusion



(a) Mono Only

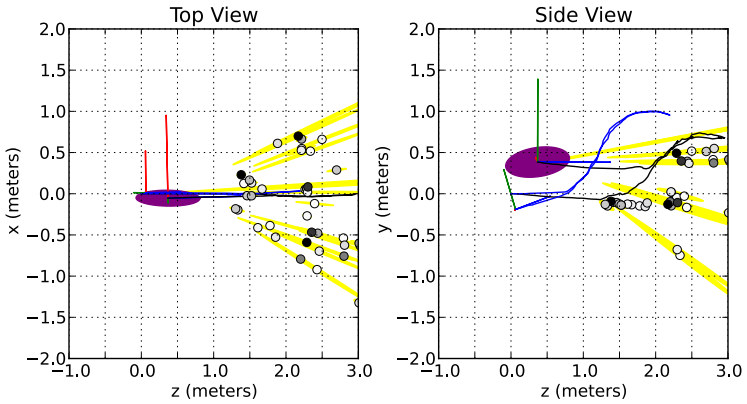


(b) Stereo Only

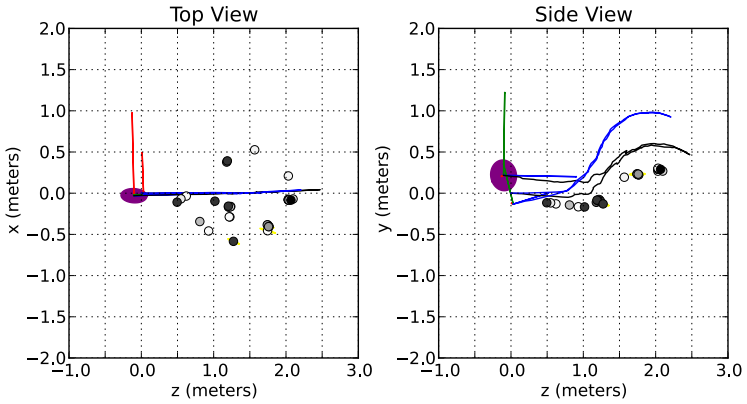


(c) Mono and Stereo

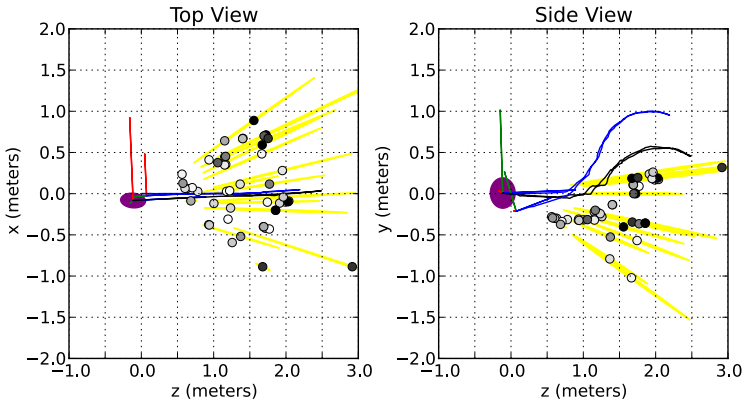
Figure 4.7: SLAM results using DBEKF with dataset "A", using only monocular observations from left camera (a), only stereo observations (b) and both monocular and stereo observations (c). Two different camera trajectories are presented in each graph: in blue is the camera trajectory with only odometry and IMU, while the black one used SLAM estimation. Both final positions have their orientation represented using RGB colors for each XYZ orientation axes, respectively. All landmarks are represented as small circumferences, and their color intensity is proportional to their utility from DBEKF (black=1, white=0). Finally, the areas in purple and yellow represent the final pose covariance and landmarks covariance, respectively.



(a) Mono Only



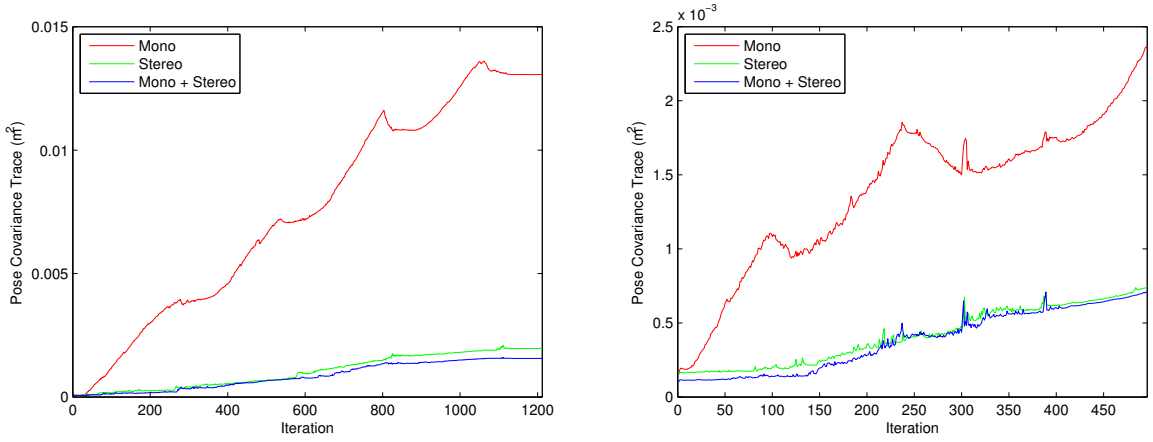
(b) Stereo Only



(c) Mono and Stereo

Figure 4.8: SLAM results using DBEKF with dataset "B", using only monocular observations from left camera (a), only stereo observations (b) and both monocular and stereo observations (c). Refer to figure 4.7 for graphical notation.

to remove a landmark from the state, forcing the removal of landmarks that could otherwise be helpful for the state estimation. For test results with $G = 1$, results are near as good as with $G = 0.8$ and $T = 0.01$, presented in section 4.3. However, it does not discard unwanted landmarks by their utility criteria from state, and if many of them are in state, they can risk



(a) (b)

Figure 4.9: Trace of the pose covariance from SLAM using DBEKF for different types of observations with dataset "A" (a) and "B" (b).

the EKF estimation with dubious information regarding the map layout.

4.6 DBEKF without odometry or IMU

If no odometry nor IMU readings are available, one can apply a *random walk* strategy for the SLAM problem. With *random walk*, the motion model assumes an additive Gaussian distributed impulse to the angular and linear velocities,

$$\begin{bmatrix} r_{t+1} \\ q_{t+1} \\ v_{t+1} \\ w_{t+1} \end{bmatrix} = \begin{bmatrix} r_t + (v_t + V_{t+1})\,\Delta\tau_{(t,t+1)} \\ q_t \times q_{((w_t+W_{t+1})\,\Delta\tau_{(t,t+1)})} \\ v_t + V_{t+1} \\ w_t + W_{t+1} \end{bmatrix}, \tag{4.1}$$

$$V_t \sim \mathcal{N}(0, \sigma_v^2\, I_{(3\times3)}) \qquad \text{and} \qquad W_t \sim \mathcal{N}(0, \sigma_w^2\, I_{(3\times3)}), \tag{4.2}$$

where σ_v and σ_w are standard deviations for linear and angular velocities, respectively. Both linear velocity v_{t+1} and angular velocity w_{t+1} are added to the camera state for this model. $q_{((w_t+W_{t+1})\,\Delta\tau_{(t,t+1)})}$ stands for the quaternion correspondent to the rotation of $((w_t+W_{t+1})\,\Delta\tau_{(t,t+1)})$.

Figure 4.13 shows SLAM results using DBEKF with *Random Walk* with dataset "A" and "B". With dataset "A", as the model assumes smooth changes to both velocities, the estimate is in conformity with the trajectory executed. In fact, using visual observation only it is possible to compute the trajectory developed, as long as the motion model has correct information regarding motion behaviour and velocity changes during dataset "A" were smooth as well. However, with dataset "B" it has some problems dealing with the roughness of the robots movement while

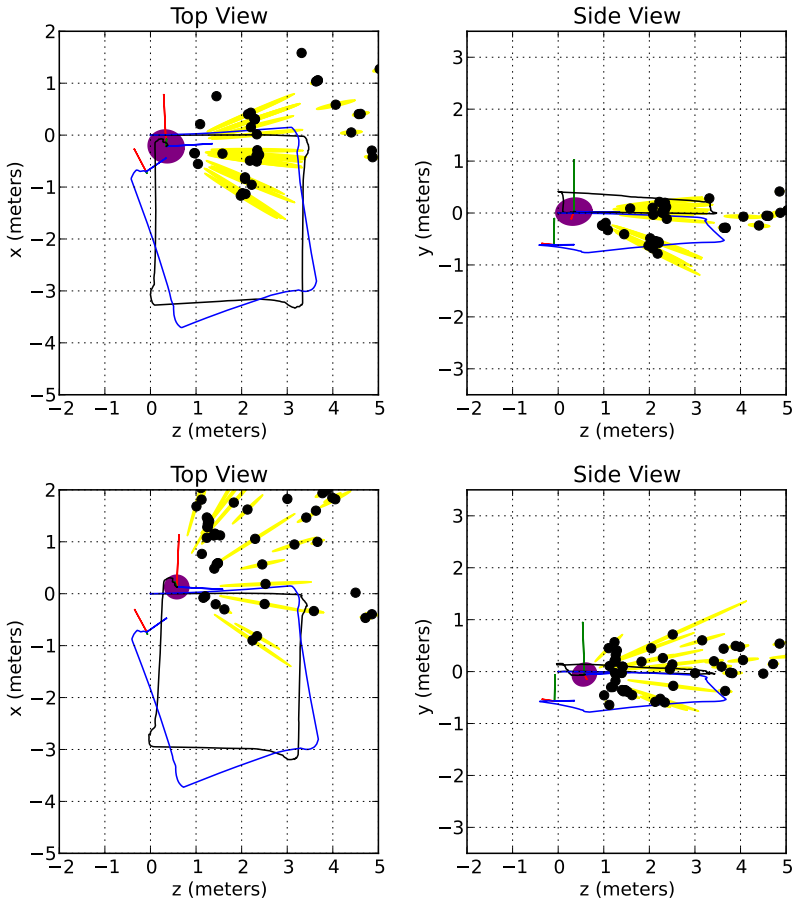


Figure 4.10: SLAM results using DBEKF with different parameters with dataset "A", where $G = 0$ (top) and $G = 1$ (bottom). Refer to figure 4.7 for graphical notation.

climbing the stairs, producing some peaks over the position estimated path.

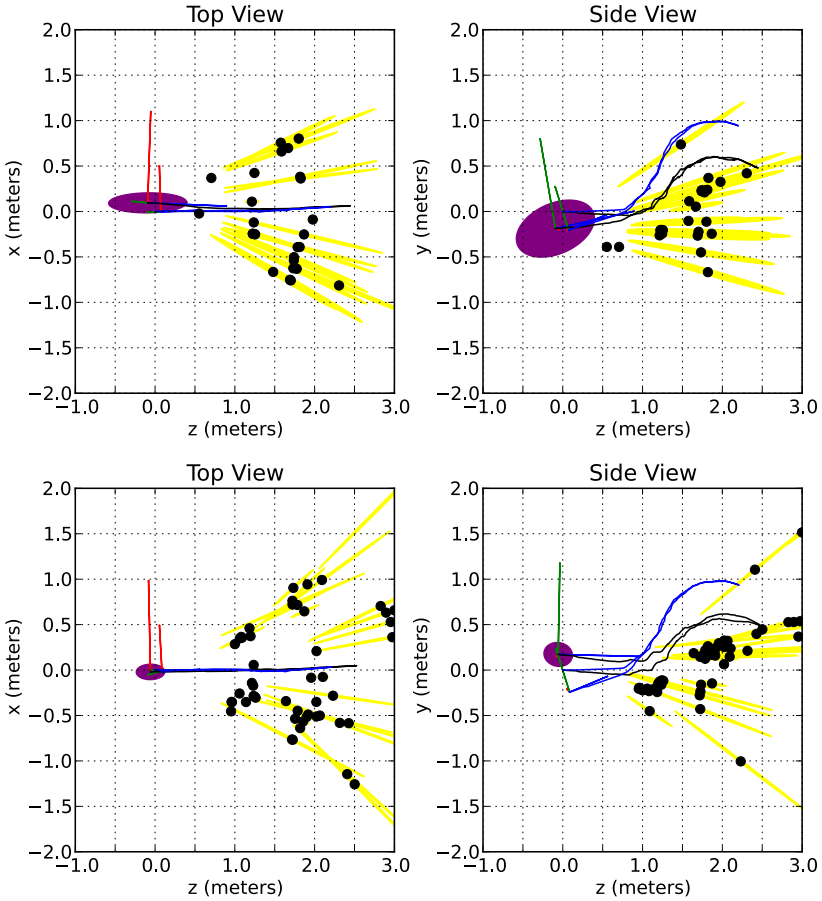
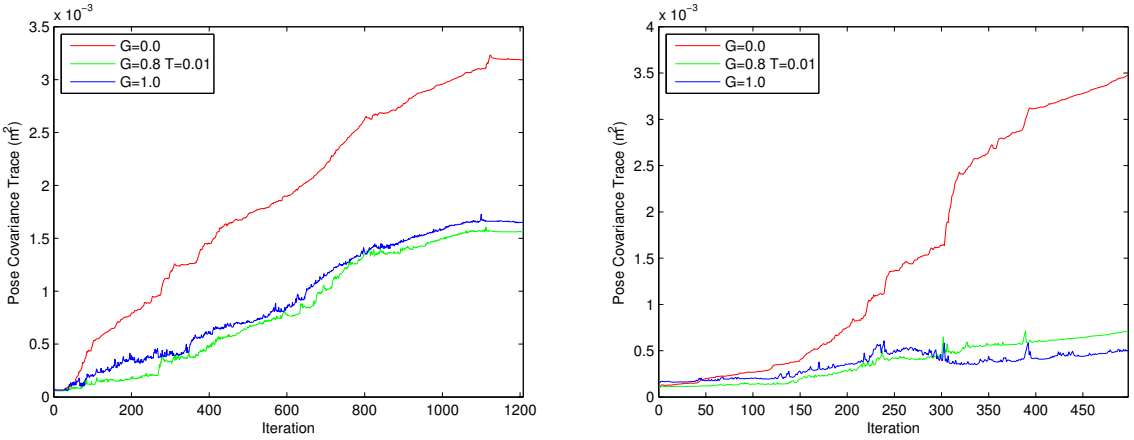


Figure 4.11: SLAM results using DBEKF with different parameters with dataset "B", where $G = 0$ (top) and $G = 1$ (bottom). Refer to figure 4.7 for graphical notation.



(a) (b)

Figure 4.12: Trace of the pose covariance from SLAM using DBEKF with different parameters with dataset "A" (a) and "B" (b).

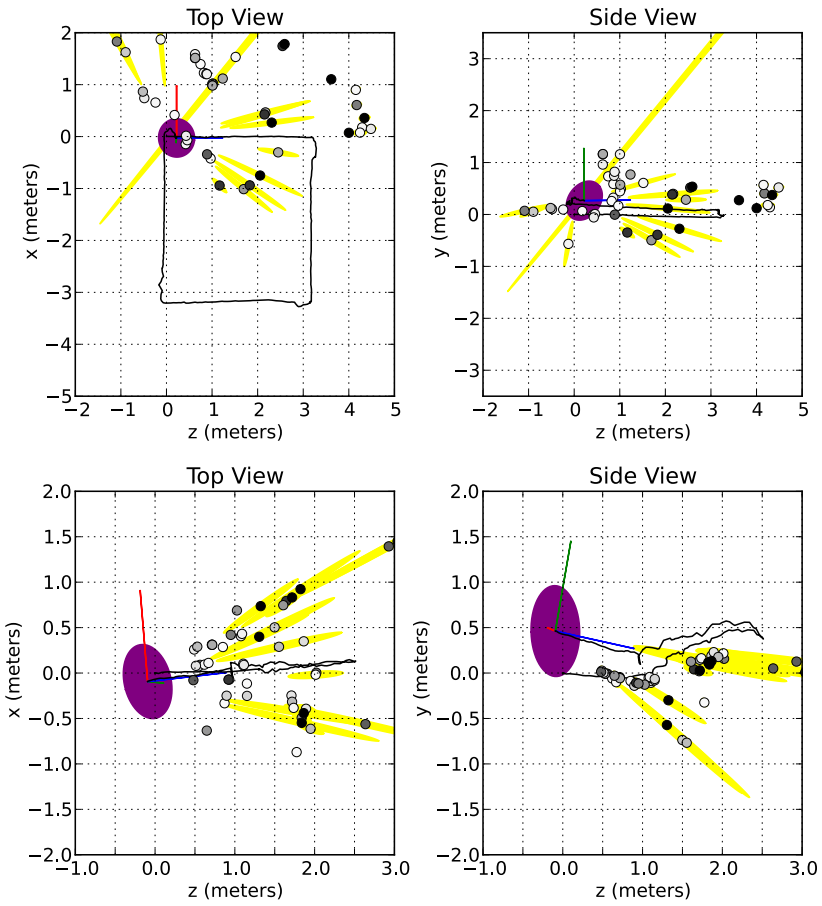


Figure 4.13: SLAM results using DBEKF with *Random Walk* with dataset "A" (top) and "B" (bottom). Refer to figure 4.7 for graphical notation.

Chapter 5

Conclusion and Future Work

The usage of both cameras as a stereo vision decreases the uncertainty from all landmarks and allows a better initialization for the *Simultaneous Localisation and Mapping* (SLAM) algorithm, but the lack of stereo features may offer some problems to the SLAM problem if no other type of observations are used. With this work, both stereo and monocular features are used as observations, and as such one can use monocular information without worrying with the scalability problem as long as stereo observations are available. From the presented results, it is clear that using both monocular and stereo observations in the way introduced by this thesis increases the overall quality of SLAM over monocular only or stereo only observations, covering a bigger space of observations and suffering no lack of scale.

Although the usage of the *Extended Kalman Filter* (EKF) has been extensively used to solve the SLAM problem, its computational complexity grows with the number of landmarks to a point in time that it becomes unusable. This thesis showed that, with DBEKF, one can achieve good estimations with constant complexity when removing landmarks from state according to an evaluation criterion. It also became clear the need to estimate the utility of each landmark in regard with past evaluations.

The presented SLAM has a lower computational effort using the feature detector ORB when compared with SURF, proving to be a good source for pose estimations for a later use with other mapping algorithms that are visually more compelling for the operator.

From the test results, it can be noticed that even without odometry and IMU measurements one can achieve a acceptable estimate of the real trajectory, unless big variations are observed that can compromise the EKF behaviour.

While some particular problems were addressed, other important factors may be of interest for further development, such as:

1. A more sophisticated learning method for landmark classification, with regard to other evolution parameters besides utility, such as landmarks spatial distribution and the innovate covariance.

2. Incorporation of the dimensional-bounded concept to other Kalman Filter variants, such as Unscented Kalman Filter for SLAM [6];

3. Experimentation with new parametrizations for both monocular and stereo cameras besides the inverse depth camera, such as the inverse scale variant [23].

Bibliography

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[2] R. Smith, M. Self, and P. Cheeseman, "Autonomous robot vehicles." New York, NY, USA: Springer-Verlag New York, Inc., 1990, ch. Estimating uncertain spatial relationships in robotics, pp. 167–193.

[3] J. Leonard and H. Durrant-Whyte, "Simultaneous map building and localization for an autonomous mobile robot," in *Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on*, nov 1991, pp. 1442 –1447 vol.3.

[4] C. F. Marques, J. Cristovão, P. U. Lima, J. Frazão, M. I. Ribeiro, and R. M. M. Ventura, "RAPOSA: Semi-autonomous robot for rescue operations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 3988–3993.

[5] S. J. Julier and J. K. Uhlmann, "A new extension of the Kalman Filter to nonlinear systems," in *Proc. of AeroSense: The 11th Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls*, 1997, pp. 182–193.

[6] R. Martinez-Cantin and J. A. Castellanos, "Unscented SLAM for large-scale outdoor environments," 2005.

[7] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," *International Joint Conference on Artificial Intelligence*, vol. 18, no. 1, pp. 1151–1156, 2003.

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

[9] S. Thrun and A. Buecken, "Integrating grid-based and topological maps for mobile robot navigation," in *Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence*, 1996.

[10] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *Int. J. Comput. Vision*, vol. 13, no. 2, pp. 119–152, Oct. 1994.

[11] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera slam," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[12] J. Civera, A. Davison, and J. Montiel, "Inverse depth parametrization for monocular SLAM," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 932–945, October 2008.

[13] P. Pinies, T. Lupton, S. Sukkarieh, and J. Tardos, "Inertial aiding of inverse depth SLAM using a monocular camera," in *Robotics and Automation, 2007 IEEE International Conference on*, 2007, pp. 2797 –2802.

[14] J. Sola, A. Monin, and M. Devy, "BiCamSLAM: Two times mono is more than stereo," in *Robotics and Automation, 2007 IEEE International Conference on*, 2007, pp. 4795 –4800.

[15] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[16] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, June 2008.

[17] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *International Conference on Computer Vision*, Barcelona, 2011.

[18] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[19] S. Särkkä, "Notes on quaternions," *Technical report*, 2007.

[20] O. J. Woodman, "An introduction to inertial navigation," University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-696, Aug. 2007.

[21] D. Oram, "Rectification for any epipolar geometry," in *BMVC'01*, 2001, pp. 653–662.

[22] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[23] D. Marzorati, M. Matteucci, D. Migliore, and D. G. Sorrenti, "Monocular SLAM with inverse scaling parametrization," in *BMVC*, 2008.