Cranfield UNIVERSITY

Diego Rodríguez Bartolomé

Single and Multiple Stereo View Navigation for Planetary Rovers

Cranfield Defence and Security

PhD

Cranfield University

Cranfield Defence and Security

Department of Informatics and Systems Engineering

PhD

2013

Diego Rodríguez Bartolomé

Single and Multiple Stereo View Navigation for Planetary Rovers

Supervisor: Dr. Nabil Aouf

5th March, 2013

This thesis is submitted in partial fulfilment of the
requirements for the Degree of Doctor of Philosophy

Abstract

This thesis deals with the challenge of autonomous navigation of the ExoMars rover. The absence of global positioning systems (GPS) in space, added to the limitations of wheel odometry makes autonomous navigation based on these two techniques - as done in the literature - an inviable solution and necessitates the use of other approaches. That, among other reasons, motivates this work to use solely visual data to solve the robot's Egomotion problem.

The homogeneity of Mars' terrain makes the robustness of the low level image processing technique a critical requirement. In the first part of the thesis, novel solutions are presented to tackle this specific problem. Detection of robust features against illumination changes and unique matching and association of features is a sought after capability. A solution for robustness of features against illumination variation is proposed combining Harris corner detection together with moment image representation. Whereas the first provides a technique for efficient feature detection, the moment images add the necessary brightness invariance. Moreover, a bucketing strategy is used to guarantee that features are homogeneously distributed within the images. Then, the addition of local feature descriptors guarantees the unique identification of image cues.

In the second part, reliable and precise motion estimation for the Mars's robot is studied. A number of successful approaches are thoroughly analysed. Visual Simultaneous Localisation And Mapping (VSLAM) is investigated, proposing enhancements and integrating it with the robust feature methodology. Then, linear and nonlinear optimisation techniques are explored. Alternative photogrammetry reprojection concepts are tested. Lastly, data fusion techniques are proposed to deal with the integration of multiple stereo view data.

Our robust visual scheme allows good feature repeatability. Because of this, dimensionality reduction of the feature data can be used without compromising the overall performance of the proposed solutions for motion estimation. Also, the developed Egomotion techniques have been extensively validated using both simulated and real data collected at ESA-ESTEC facilities. Multiple stereo view solutions for robot motion estimation are introduced, presenting interesting benefits. The obtained results prove the innovative methods presented here to be accurate and reliable approaches capable to solve the Egomotion problem in a Mars environment.

To my parents, Mari Paz and Pablo

Acknowledgements

I would like to express sincere gratitude to my supervisor Dr Nabil Aouf for his continuous guidance, support and patience as well as for giving me the chance to undertake this research in the first place. I am also grateful to my committee members, Dr Mark Richardson and Dr David James, for their supportive approach in discussing and reviewing my progress in addition to their invaluable comments and suggestions.

I would also like to thank the European Space Technology Centre (ESTEC) of the European Space Agency (ESA) and European Aeronautic Defence and Space Company (EADS-ASTRIUM) for their financial support.

I wish to extend special thanks to all my friends in my home town of Madrid, because every time I see you, you make me feel at home as if I have never left. Missing time with you is a high opportunity cost.

Many thanks are reserved to all the new friends and colleagues that I have had the pleasure to meet at the Defence Academy / Cranfield Defence and Security at Shrivenham. Every technical, philosophical or personal discussion with you has been a joy. Also, this adventure would have never been the same without all the great people I met in Oxford along this journey. You have marked these years being the best friends and housemates I could have ever found. Mohd, Karim, Lounis, Steven, Steve, Rodrigo, Mateusz, Pierre, Antonio, Tina, Anna, Becks, Natalia, Pam and Ewa are only a few of the names extracted from an unlimited list.

Javier García Bayón, one cannot find a more reliable and trustful friend and housemate than you have been to me, many thanks for all your support and help along these years. Javier Grande Gil, you are the unfailing source of positiveness and service and the ideal mix between reality and motivation that I could not have gone without, no words can express my gratitude. Saad Ali Imran, your rigorous and professional feedbacks together with your inexhaustible support and kindness deserve nothing less than my most honest admiration and gratefulness.

Last, but never least, I would like to thank my parents Mari Paz and Pablo and my sister Marta. Your endless love and support has made this possible.

Shrivenham, 5th March, 2013 Diego Rodríguez Bartolomé

Contents

List of Figures

List of Tables

Nomenclature

Conventions

a	Lower case letters indicate scalars
$\boldsymbol{b}$	Bold face letters indicate vectors
b_i	i^{th} element of the vector $\boldsymbol{b}$
b_x, b_y, b_z	Cartesian components of the vector $\boldsymbol{b}$
$\boldsymbol{b}^T, \boldsymbol{C}^T$	Superscript T indicates transposition
$\boldsymbol{b}^*, \boldsymbol{C}^*$	Superscript $*$ indicates hermitian transformation
$\boldsymbol{b}_H$	Subscript H indicates homogeneous coordinates of the vector $\boldsymbol{b}$
$\boldsymbol{C}$	Upper case bold letters indicate matrices
$\boldsymbol{c}_i$	i^{th} row of the matrix $\boldsymbol{C}$
c_{ij}	Component of matrix $\boldsymbol{C}$ at row i and column j
a_L, $\boldsymbol{b}_L$, $\boldsymbol{C}_L$	Subscript L indicates left camera of the stereo pair
a_R, $\boldsymbol{b}_R$, $\boldsymbol{C}_R$	Subscript R indicates right camera of the stereo pair
$\hat{a}$, $\hat{\boldsymbol{b}}$, $\hat{\boldsymbol{C}}$	Hat symbol indicates estimation
$\boldsymbol{C}^{-1}$	Superscript -1 indicates inverse of the matrix
$\boldsymbol{C}^{+}$	Superscript $+$ indicates pseudo inverse of the matrix
$\overline{AB}$	Segment from point A to point B
P	Capital letters generally denominate points in $\mathbb{R}^3$

Image and camera notation

O	Center of perspective of the camera
Q	Projection of point P onto the Image plane
π	Image plane
$\boldsymbol{x}$	Pixel position on Pixel frame
u, v	Pixel coordinates on Pixel frame
$\boldsymbol{c}$	Principal point
u_0, v_0	Pixel coordinates of the Principal point $\boldsymbol{c}$
$\tilde{\boldsymbol{x}}$	Pixel position on the Image plane frame
$\tilde{x}$, $\tilde{y}$	Pixel coordinates Image plane
$\mathbf{x}_C^P$	Position of point P on Camera frame
x_C^P, y_C^P, z_C^P	Coordinates for the point P on Camera frame
$\mathbf{x}_C^Q$	Position of point Q on Camera frame
$\mathbf{x}^P$	Position of point P on World frame
x, y, z	Coordinates of the point P on World frame
$\boldsymbol{M}$	Projection matrix
$\boldsymbol{T}$	Homogeneous transformation matrix
$\boldsymbol{K}$	Intrinsic Calibration Matrix
$\widetilde{\boldsymbol{K}}$	Augmented version of the Intrinsic Calibration Matrix
f	Focal length of the camera
α_u, α_v	Focal lengths on the cartesian axis
γ	Skew factor on Camera Intrinsic Camera
s_x, s_y	Scale factors on the cartesian axis
$\tilde{\boldsymbol{x}}_d$	Distorted position of the pixel $\tilde{\boldsymbol{x}}$ on the image plane
$\tilde{x}_d$, $\tilde{y}_d$	Coordinates of $\tilde{\boldsymbol{x}}_d$ on the Image plane

r	Euclidean distance of a pixel with respect to the Principal point $\boldsymbol{c}$
k_i	i^{th} distortion coefficient of the lens
$\boldsymbol{\omega}$	Scaling factor for homogeneous pixel coordinates
$\boldsymbol{R}$	Rotation matrix
$\boldsymbol{t}$	Translation vector
Bl	Baseline of the stereo pair
d	Horizontal disparity on the stereo images
I, J	Images
$\boldsymbol{x}$	Pixel of coordinates u, v
$I(\boldsymbol{x})$	Image I at pixel $\boldsymbol{x}$
$I(u,v)$	Image I at pixel of given coordinates u, v
$c(u,v)$	Autocorrelation function at pixel coordinates u, v
Δu, Δv	Horizontal and Vertical displacement in Pixel coordinates
W	Window around the pixel, size is normally $2N+1$
N	Integer number of pixels
$\sum_W$	Summation along the pixels contained on an image window W
I_Σ	Integral image computed from image I
∇I	Spatial gradient of the Image I
I_u, I_v	Spatial derivatives in u and v directions
$\mathcal{F}^{(i)}$	i^{th} feature
$u^{(i)}$, $v^{(i)}$	Coordinates of the i^{th} feature $\mathcal{F}^{(i)}$
$\zeta(\boldsymbol{x})$	Cornerness function at the pixel $\boldsymbol{x}$, from Harris Corners
k	User-defined parameter used on for the ζ function
$\det(W)$	Determinant of the image window W
$\mathrm{tr}(W)$	Trace of the image window W

$m_{pq}\big(I(u,v)\big)$	Geometrical moment of image pixel $I(u,v)$
p, q	Order of the moments
ε	Cross correlation error
n	Number of detected interest points in the image
tp	True positives, number of correct matches
fp	False positives, number of wrong matches
ρ	Precision of the matching process
σ	Matching rate
$r_{i,j}$	Distance in Pixel frame coordinates from $\mathcal{F}^{(i)}$ to $\mathcal{F}^{(j)}$
ε	Residue to minimise optical flow in KLT
$\boldsymbol{d}$	Displacement vector in Pixel coordinates, optical flow
$w(\boldsymbol{x})$	Weighting function for the Pixel $\boldsymbol{x}$
G	Spatial gradient matrix of the image in the window W
$\boldsymbol{g}^T$	Spatial gradient of the Image I, same as ∇I
λ_1, λ_2	Eigen values of the spatial gradient matrix, G
τ	Time consumption
m	Mismatching ratio
η_ρ	Ratio of true positives w.r.t. the best case
η_τ	Ratio of time consumptions w.r.t. the best case

Kalman Filter symbols

$\mathbf{x}_k$	State vector of the system at time-step k
$\boldsymbol{z}_k$	Measurements vector at time-step k
$\boldsymbol{f}$	Transition function, also process function
$\boldsymbol{F}_k$	Transition matrix, also process matrix, at time-step k

$\boldsymbol{G}_k$	Input matrix, at time-step k
$\boldsymbol{H}_k$	Observation matrix, at time-step k
$\boldsymbol{h}$	Observation function
$\boldsymbol{P}_k$	Covariance of the state matrix, at time-step k
$\boldsymbol{S}_k$	Innovation prediction matrix, at time-step k
$\boldsymbol{Q}_k$	Process noise covariance matrix, at time-step k
$\boldsymbol{K}_k$	Kalman gain matrix, at time-step k
$\boldsymbol{R}_k$	Measurements noise covariance matrix, at time-step k
$\mathbf{u}_k$	Input vector, at time-step k
$\mathbf{w}_k$	Process noise, at time-step k
$\mathbf{v}_k$	Measurements noise, at time-step k
$\boldsymbol{\nu}_k$	Innovation vector, at time-step k
ψ	Yaw angle for a robot moving in the ground plane
γ	Bearing angle on the steering wheel of a three-wheeled robot
$\mathcal{L}^{(i)}$	i^{th} landmark
$\boldsymbol{l}^{(i)}$	Position vector of the i^{th} landmark
p	Number of landmarks
$\rho^{(i)}$	Range of the i^{th} landmark w.r.t the robot, LASER sensors
$\theta^{(i)}$	Bearing angle of the i^{th} landmark w.r.t the robot, LASER sensors
$r^{(i)}$	Distance of the i^{th} landmark w.r.t the robot, 3D LASER sensors
$\alpha^{(i)}$	Azimut of the i^{th} landmark w.r.t the robot, 3D LASER sensors
$\beta^{(i)}$	Elevation of the i^{th} landmark w.r.t the robot, 3D LASER sensors
V	Velocity of the robot
x_M, y_M, z_M	Coordinates of the robot
ΔT	Discrete time increment

$\boldsymbol{\delta}^{(i)}$	Descriptor of the i^{th} landmark
$\eta^{(i)}$	Rate of usability of the i^{th} landmark
$t^{(i)}$	Time-steps passed from i^{th} landmark initialization
$n^{(i)}$	Number of time-steps that i^{th} landmark appeared since initialization
N	Number of features
$\varepsilon_{i,j}$	Euclidean distance from a descriptor i to descriptor j
$\mathcal{F}^{(i)}$	i^{th} Feature
$\boldsymbol{\xi}_i$	Crescent order sorted vector of distances $\varepsilon_{i,j}$ for every $j \neq i$
κ	Matching threshold
m_{ij}	Component of the projection matrix $\boldsymbol{M}$ at row i and column j
$\hat{\mathbf{x}}_k$	Estimate of $\mathbf{x}_k$
$\dot{\mathbf{x}}$	Time derivative of $\mathbf{x}$
$\boldsymbol{I}$, $\boldsymbol{I}_n$	Identity matrix, Identity matrix of order n
$\mathbf{0}$, $\mathbf{0}_n$	Zero vector/matrix, zero matrix of order n

Motion Estimation

$\boldsymbol{\Sigma}$	Singular matrix associated to the system's matrix $\boldsymbol{A}$
$\boldsymbol{U}$, $\boldsymbol{V}$	Unitary matrices used for SVD decomposition of the matrix $\boldsymbol{A}$
$\{\mathbf{x}_k^{(i)}\}$	Cloud or set of 3D points at time k for the different points (i)
$\boldsymbol{R}_{k\to k+1}$	Rotation matrix that describes a transformation from time k to $k+1$
$\boldsymbol{t}_{k\to k+1}$	Displacement vector from time-step k to time-step $k+1$
μ_k	Mean vector of $\mathbf{x}_k^{(i)}$ for each i. Centre of Gravity of the set $\{\mathbf{x}_k^{(i)}\}$
$\boldsymbol{\Sigma}_{k,k+1}$	Cross covariance matrix for the clouds at k and at $k+1$
$\boldsymbol{Q}(\boldsymbol{\Sigma}_{px})$	Quaternion matrix used on the quaternion motion estimation method

$\boldsymbol{q_R}$	Quaternion associated to the rotation on the quaternion method
q_i	For $i = 0,1,2,3$, components of the quaternion $\boldsymbol{q_R}$
$\boldsymbol{p}$, $\hat{\boldsymbol{p}}$	Transformation parameters and its estimate
$\boldsymbol{T}(\boldsymbol{p})$	Homogeneous transformation matrix as a function of the $\boldsymbol{p}$
$\boldsymbol{T}_{xyz}$	Displacement written as a transformation matrix
$\boldsymbol{R}_x(\alpha)$	Pure rotation of α $[rads]$ about the x axis
$\boldsymbol{R}_y(\beta)$	Pure rotation of β $[rads]$ about the y axis
$\boldsymbol{R}_y(\gamma)$	Pure rotation of γ $[rads]$ about the z axis
C_θ, S_θ	Cosine and sine functions of the angle θ
$\boldsymbol{t}$	Displacement vector composed by the components t_x, t_y and t_z
$S(\boldsymbol{p})$	Cost function on $\boldsymbol{p}$ used for motion estimation optimisation
$\boldsymbol{J}$	Jacobian of the cost function $S(\boldsymbol{p})$
$\boldsymbol{r}$	Vector of residuals used in the optimisation process
$\boldsymbol{r}_{fw}$	Vector of residuals, forward reprojection strategy
$\boldsymbol{r}_{bw}$	Vector of residuals, backward reprojection strategy
$\boldsymbol{r}_{dual}$	Vector of residuals, dual reprojection strategy
$r_j\left(\boldsymbol{p}, \mathbf{x}^{(i)}\right)$	j^{th} residual function, evaluated at $\boldsymbol{p}$ and $\mathbf{x}^{(i)}$
$\boldsymbol{f}$, $\boldsymbol{f}^{(k)}$	Projection function, at view k when multiple views
$\boldsymbol{y}$	Reduced vector of projected coordinates
$\tilde{\mathbf{x}}_H$	Coordinates of the feature $\mathbf{x}_H$ after the motion
$\boldsymbol{T}_k(\hat{\boldsymbol{p}})$	Estimation of the transformation a from k to $k+1$
$\boldsymbol{\varepsilon}$	Threshold used for motion estimations inliers classification
$\boldsymbol{\omega}$	Weighting coefficient on Covariance Intersection
$\boldsymbol{v}_t$, $\boldsymbol{a}_t$	Velocity and acceleration of the vector $\boldsymbol{p}$ on Covariance Intersection

Acronyms

AI	Artificial Intelligence
BA	Bundle Adjustment
BRISK	Binary Robust Invariant Scalable Keypoints
CCD	Charge Coupled Device
CMOS	Complementary Metal Oxide Semiconductor
DOF	Degrees Of Freedom
EKF	Extender Kalman Filter
ESTEC	European Space TEchnology Centre
ETSI	Escuela Tecnica Superior de Ingenieros
ExoMaDeR	EXOMArs DEmonstration Rover
FOV	Field Of View
FPS	Frame Per Second
GFTT	Good Features To Track
GN	Gauss-Newton
GPS	Global Positioning System
HMSURF	Harris-Moments-SURF
HSURF	Harris-SURF
ICP	Iterative Closes Point
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IR	Infra Red
KF	Kalman Filter
KLT	Kanade-Lucas-Tomasi
LASER	Light Amplification by Stimulated Emission of Radiation

OpenCV	Open Computer Vision libraries
PANGU	Planet and Asteroid Natural-scene Generation Utility
PTU	Pan and Tilt Unit
RADAR	RAdio Detection And Ranging
RANSAC	RANdom SAmple Consensus
SFM	Structure From Motion
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localisation And Mapping
SONAR	SOund Navigation And Ranging
SSD	Sum of Square Differences
SURF	Speeded Up Robust Features
SVD	Singular Value Decomposition
UKF	Unscented Kalman Filter
U-SURF	Upright-SURF
USB	Universal Standard Bus
VO	Visual Odometry
VSLAM	Visual Simultaneous Localisation And Mapping

Chapter 1

Introduction

Evolution of science has been such that nobody can question an autonomous robot's ability in performing certain challenging tasks with efficiency and reliability. Autonomous robotics is a multidisciplinary science that lays close to many fields of knowledge and which has also empowered the origin of many others. Mechatronics, computer science, machine vision and Artificial Intelligence (AI) are to name a few. The integration of all these disciplines has made possible higher levels of autonomy for robots in the last decade.

Nowadays, autonomous robots are also used to extend the possibilities of manned machines and vehicles. Benefits range from low costs to safe and security of personnel. Among many others, autonomous robots present advantages derived from their ability to overcome spatial or environmental restraints, reaching places which human beings cannot safely access such as mine exploration, deep underwater environments and even outer space. In all of these contexts, the risk reduction with respect to human expeditions makes autonomous robots a viable option.

One of the main challenges related to autonomous robots is the problem of self-localisation. To tackle this issue, Simultaneous Localisation And Mapping (SLAM) is maybe the most extended solution found in the literature. Since its first appearance in 1986, this solution and many beneficial variations have been used by researchers all across the world to address the problem of autonomous navigation [1, 2, 3].

Two types of problems can be formulated in the context of SLAM. When the robot has prior information about its surroundings, normally in the form of a map and the goal is to recognise the environment characteristics to locate itself within this map. This problem is referred in the literature as *the kidnapped robot problem*. The second possibility is for the robot not to have any previous knowledge of the surroundings where it has been placed. This is known as *the wake-up robot problem*. The robot's goal in this case is to analyse its surrounding to make a map, defining reference landmarks, while defining its own position with respect to the map. The process of extending a map of the surroundings is called *mapping*.

In the context of a robot waking up in an unknown environment, the SLAM solution necessitates a set of onboard sensors. These devices, which measure physical proprieties of the surrounding, are responsible for data acquisition used by SLAM. There are a variety of sensors that can be used to allow interaction between the robot and its environment. These include RADAR, SONAR, LASER and CCD cameras which are used as sensing strategies for localisation and mapping in submarine, aerial and land applications [4, 5, 6, 7, 8].

Many SLAM solutions have been developed using the Kalman Filter (KF) and its variations i.e. Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF) [9, 10]. The Kalman Filter is optimal for linear systems which makes it very useful to process information from the onboard sensors under certain noise assumptions.

Other alternative filters like H_∞ or the FastSLAM are proposed to cope with the non-linearities arising in the SLAM problem [11, 12]. One of the advantages of these methods based on filtering is the possibility of data fusion provided by different sensors.

Contributions relevant to improving SLAM solutions have appeared over the years. Some of these contributions are oriented to alleviate the computational expense of the solution [12, 13, 14, 15]. Another important aspect that has motivated the evolution of the algorithms is the need for robustness of the results [16, 17]. Advanced solutions as iSAM have been proposed tackling both of the issues [18, 19].

With regards to being able to sense the environment a number of alternatives to provide robots with visual perception have been studied. Whereas the most common solutions are based on mono camera or stereo camera designs [11, 20] other options as trinocular solutions or custom solutions with multiple cameras have been presented [21, 22]. However, solutions based on stereo cameras present certain advantages with respect to monocular vision, as the complexity of the photogrammetry is reasonably narrowed for this.

There are some reasons that make cameras an appealing sensing solution. The first reason is the cost effectiveness, not only in terms of price but also in terms of power consumption. The passive nature of majority of the cameras makes them a low consuming sensor. This is specially useful for autonomous robot applications, where energy efficiency is desirable. Moreover, images are meaningful for a human end user. In this sense, imagery data can be exploited not only onboard an unmanned robot, but also be used for other scientific purposes. For instance, in case of planetary exploration by means of an autonomous robot, images of the landscape can be of further use, as opposed to inertial measurements which are of limited use.

Another approach to solve the localisation problem is via Visual Odometry. This solution is not based on filtering algorithms like the KF, but on optimisation techniques applied to the 3D reconstructed information of reference points perceived from travelled areas. Surveys of these techniques are found in the literature [23, 24, 25].

Both Visual Simultaneous Localisation and Mapping (VSLAM) and Visual Odometry approaches are highly dependant on the visual information. Although vision solutions are effective they however also present certain limitations and challenges. When perception is carried out through visual systems, making a robot autonomous requires that images should be processed to extract useful information in an automatic fashion. This information is usually extracted at the low level by detecting geometrical elements as feature points on the images. One of the classic methods to achieve this is the Kanade Lucas Tracker (KLT) technique used for feature tracking [26]. The evolution detection techniques from KLT has been remarkable, moving to advanced feature detection and sophisticated description that allow the unique identification of features. The first important step in this evolution is the Scale Invariant Feature Transform (SIFT) developed by David Lowe *et al.*[27]. More recent strategies have looked at making detection and description more cost efficient in terms of computation. For example Speed Up Robust Features (SURF) is easier to implement and faster while retaining the performance [28]. These techniques have found application in different fields such as face recognition [29] and action recognition where a 3D SIFT alternative is proposed [30].

The continuous need for image processing techniques to be ever more reliable is a driving force for new detection and description techniques. The goal for these new techniques is to augment the robustness, repeatability or invariance to illumination changes of the detected features, while providing efficient solutions [31, 32, 33, 34].

Some studies conducted to analyse and evaluate the available approaches used for VSLAM are contained in [35, 36].

Previously mentioned merits of VSLAM and Visual Odometry applied to autonomous robots constitute two suitable solutions for planetary rovers. Stereo solutions have been widely studied in reputed places as NASA JPL [37, 38, 39], whereas some other solutions as omnidirectional vision solutions are also explored [40].

Figure 1.1: ExoMars exploration robot.

High degrees of autonomy are required for planetary exploration robots, partially due to restricted communication between the Earth and other planets. Computer vision is now able to provide that level of autonomy. Nevertheless, robustness, reliability and efficiency of visual processes is still a challenging and fruitful field of investigation.

The homogeneity of an alien terrain - as the one on Mars - makes a planetary scenario a much more challenging environment compared to an urban one, where colour distinctiveness or object recognition can be employed. At the same time, the frame rate is an important parameter in the context of planetary missions, where power supplies have more constraints than in other autonomous robots applications. In this context, large amounts of data are cumbersome and ineffective and have high energy cost implications.

In this context, the use of multiple stereo cameras views for increasing the robot navigation reliability is an appealing way of extending the capabilities of an autonomous robot equipped with an orientable stereo camera. This solution, proposed in this thesis, entails fusing visual information in a new unexplored way.

Real time performance is a requirement for planetary missions, where the degree of autonomy is high. The implementations of the solutions presented here, written in C and C++, have been tested and are capable of running in real time conditions.

1.1 Research Motivation

The main motivation behind this research is the autonomous navigation for planetary rovers. More specifically, it is the utilisation of stereo cameras to provide unmanned mobile robots with accurate and reliable tools to perform self-localisation and navigation tasks within unexplored planetary terrains.

Part of the complexity resides on the homogeneity of the observed scene. Whereas distinctive and characteristic colours, shapes and objects are observed in urban-like environments, other sort of visual traits have to be sought as visual references for planetary imagery, where all the objects or features may look extraordinarily similar.

The described working conditions motivate solutions where the use of raw image data is processed to give meaningful information. In this manner, the visual information perceived and analysed from exploration rovers is used to provide accurate measurements of the environment's dimensions and layout. In subsequent processing stages, those measurements can be utilised to estimate how the robot moves and changes its orientation in space. The ground where the robot lays cannot be assumed to be planar makes the estimation of 6 degrees of freedom (DOFs) a requirement and also a challenge.

Moved by the willingness of exploiting a stereo camera header to solve visual navigation problems, solutions based on multiple stereo images constitute a new approach. As opposed to the mainstream effort of the scientific community to fuse the information gathered from different sensors, the interest here is to benefit from a modest yet reliable system, based mostly or even uniquely on imagery obtained from a single stereo rig. High levels of accuracy on the motion estimation results and versatility of the solutions are priorities of this study.

1.2 Thesis contributions and organisation

This thesis focuses on developing a set of methods for ground robot visual navigation. The goal has been to improve the solutions that allow autonomous vehicles to operate in unknown environments without compromising the efficiency. The investigated topics are related to feature detection and description, photogrammetry, feature matching and tracking, localisation, motion estimation, Map building and management and VSLAM. In addition, data fusion methods are developed to take advantage of the information contained in multiple view stereo sequences.

The contents of this work have been either published or are being prepared for publication at reputed conferences and journals. The following summarises the contributions of the thesis while highlighting the pieces that have been written in a form of a separate manuscript - all of which are listed below.

A brief summary of the contributions presented in this thesis are as follows:

- **Robust visual scheme (HMSURF)** (*paper3, paper4*)

  In Chapter 3 a novel visual processing technique to extract and describe features is presented. The first goal of this technique is to provide robust feature detection for mobile robots imagery, where lighting conditions vary along the acquired sequence. Secondly, the ability for the detected features to be uniquely identified is incorporated by adding Speeded Up Robust Features (SURF). The technique is compared to most relevant solutions in the field and proved to represent a reliable visual module solution.

- **VSLAM map management**

  Chapter 4 presents a solution for the map management growth that takes place on VSLAM approaches. By introducing the concept of *rate of usability* for the

mapped landmarks, the VSLAM is added the capability of recognising what is the likelihood for a landmark to be useful. The derivation of our VSLAM solution is developed in detail and some other minor contributions are also presented.

• **Robust visual scheme on VSLAM** (*paper5*)

In Chapter 5 the concepts presented in the two previous chapters are combined together to provide a novel robust VSLAM approach. An extensive analysis of the most influencing parameters of the VSLAM system is conducted over real data to determine how to make the system provide the best solution. To conclude the chapter, results obtained for a long range trajectory sequence are shown demonstrating the capacities of the system.

• **Linear Robust Egomotion** (*paper1*)

Chapter 6 presents a family of solutions oriented to solve the mobile robot self localisation. The first part of the Chapter analyses the simulation results obtained when the robust visual module is combined with different motion estimation approaches. The second part presents the results obtained when the solution is applied on real data when the quaternion motion estimation method is combined with our visual scheme.

• **Nonlinear Robust Egomotion** (*paper2*)

In Chapter 7 a novel Egomotion solution based on robust features and a dual reprojection scheme is derived. Combining the advantages of nonlinear motion estimation techniques, based on a Gauss Newton optimisation algorithm, different possibilities on the cost function are proposed and analysed. Experimental results obtained from long range trajectories are presented in the end of the Chapter showing the accuracy of this solution on real data urban environments.

• **Multiple Stereo Views Egomotion** (*paper6*)

Chapter 8 presents two innovative imagery data fusion techniques oriented to improve Egomotion for mobile robots. The first part of the Chapter summarises the experimental data collection designed to acquire the datasets employed for the latter validation process. Then, a solution based on Kalman Filter (KF) and Covariance Intersection (CI) and an alternative solution based on simultaneous optimisation of motion estimation for multiple stereo views are presented.

(1) Diego Rodriguez, Nabil Aouf and Mark Richardson **"Moments-based stereo camera egomotion analysis and results for long-range trajectories,"** in The Imaging Science Journal, *published.*

(2) Diego Rodriguez and Nabil Aouf **"Robust EgoMotion for Large-Scale Trajectories,"** in IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), 2012, Sept. 2012.

(3) Diego Rodriguez and Nabil Aouf **"Robust Harris-SURF features for robotic vision based navigation,"** in 13th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2010, pages 1160 1165, Sept. 2010.

(4) Diego Rodriguez and Nabil Aouf **"Robust features detection for autonomous guidance,"** in 8th Electro-Optics & Infrared Conference, Shrivenham, UK Defence Academy, Jul. 2010.

(5) Diego Rodriguez, Nabil Aouf and Abdelkrim Nemra **"Robust Moment Stereo Based Visual SLAM for Rovers: Experiments and Analysis,"** *to be submitted.*

(6) Diego Rodriguez and Nabil Aouf **"Multiple Stereo Views Visual Egomotion for Mobile Robots,"** *to be submitted.*

Chapter 2

Stereo Imaging

2.1 Overview

Human beings, as with animals, perceive their environment through a collection of senses. Perception is then the first stage for a living being to interact with its surroundings. Following a similar reasoning the evolution of machines into today's robots has been motivated by these ideas. Whereas the first machines were capable of helping humans they were also very dependant, given their lack of means to know their states. The wish for machines to reach higher levels of autonomy lead to the appearance of sensors. The objective of sensors is providing machines with the necessary tools to perceive their internal and external states, but the later enable further interaction with the environment.

There are many sensor-based solutions to perceive the information of the environment surrounding a mobile robot. Among a vast number of possible choices there are some devices which stand out because of their accuracy, reliability or cost.

Sound navigation and ranging (SONAR) [41], radio detection and ranging (RADAR) [42], light amplification by stimulated emission of radiation (LASER) [6, 43, 44] and digital cameras are some of the possibilities available for these purposes [45]. Nonetheless, each sensing choice has its associated benefits along with its shortcomings.

While the two former possibilities are considered cheap sensing solutions, they normally present precision and reliability issues. The LASER alternative is the most accurate of the mentioned sensors, but is also the most expensive. Camera based solutions on the other hand represent a good tradeoff between accuracy and price.

The wide range of camera devices available for equipping mobile robots makes this approach a flexible solution. Furthermore, the variety of feasible setups using cameras is enormous. Solutions based on single, omnidirectional, stereo and multiple stereo cameras are found in the literature. This thesis focuses on the later two.

This Chapter introduces the basic concepts of image formation that serve as a foundation for the main contributions of this text. The most commonly used model for digital cameras devices is the so-called pinhole camera model [46, 47]. Using this model the stereo camera fundamentals are derived at the end of the chapter.

2.2 Pinhole Camera Model

In this model every point in 3D space will be projected onto a plane, known as the *image plane*. For this geometrical model there are four important reference frames that have to be taken into account to express the position of any 3D point P in the field of view and its projected 3D point Q contained on the image plane.

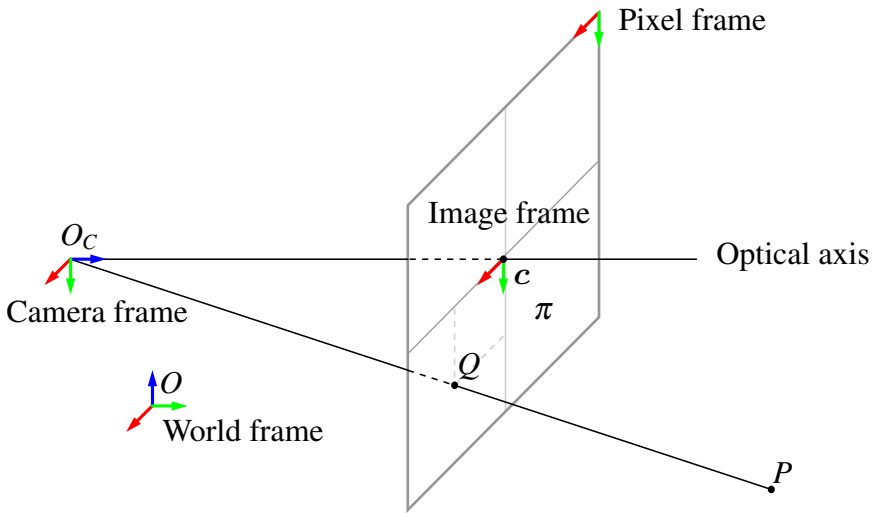


Figure 2.1: Reference frames on the camera.

a - Pixel frame: The origin of this frame is conventionally located on the upper-left corner of the *image plane* π. Coordinates on this frame are measured in pixels and take only integer values $\boldsymbol{x} = (u, v)^T$. It is normal convention to use u as the horizontal coordinate and v as the vertical coordinate. The axis directions are left-right and top-bottom for u and v respectively as in a matrix.

This frame is not used to express 3D coordinates, but only the planar coordinates of the projected point Q on the image plane.

b - Image plane frame: The *principal point*, roughly located on the centre of the image plane, is the origin of this frame. This point, located at pixel frame coordinates $\boldsymbol{c} = (u_0, v_0)^T$, corresponds to the projection of the *centre of perspective projection* of the camera, O_C, on the image plane. It also belongs to the optical axis of the camera. Coordinates on this frame take real values $\tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y})^T$.

As for the previous frame, this one is not used to express the 3D position of a point P, but its projection on the image plane, Q.

c - Camera frame: The origin of the frame is located on the *centre of perspective projection* of the camera. The 3D coordinates of the point P in space are noted as $\mathbf{x}_C^P = (x_C^P, y_C^P, z_C^P)^T$ in this frame. The distance between the origin of this system and the image plane π is the *focal length*, f, of the camera. The depth coordinate, z_C^Q, for any projected point Q is equal to the focal length.

The *optical axis* is the perpendicular line to the image plane that passes through the centre of perspective projection of the camera O_C. The intersection between the optical axis and the image plane corresponds to the principal point, $\boldsymbol{c}$.

d - World frame The world reference frame, or simply global reference frame, is a reference frame external to the camera used for the global positioning. The 3D coordinates of a point P are $\mathbf{x}^P = (x^P, y^P, z^P)^T$ on this frame.

Figure 2.1 depicts the described reference frames with the conventions used here.

It is important emphasising that the two first reference frames, pixel frame and image plane frame, can be considered as 2D reference frames, while the camera and world reference frames express the position of the point in the 3D euclidean space.

The relationship of the coordinates between the different frames are expressed by the *camera parameters* [48, 49]. On a perspective projection model these parameters can be classified as *intrinsic parameters* and *extrinsic parameters*. The point P is projected onto the image plane through a projection matrix $\boldsymbol{M}$, that can be written in terms of the *augmented intrinsic calibration matrix*, $\widetilde{\boldsymbol{K}}$, and the *extrinsic calibration matrix*, $\boldsymbol{T}$, as in (2.1).

$$\boldsymbol{x}_H = \boldsymbol{M} \cdot \mathbf{x}_H = \widetilde{\boldsymbol{K}} \cdot \boldsymbol{T} \cdot \mathbf{x}_H \tag{2.1}$$

where the subscript H indicates that homogeneous coordinates are used.

2.2.1 Intrinsic parameters

The intrinsic camera parameters are necessary to link the pixel coordinates of an image point with the corresponding coordinates in the camera reference frame. These are parameters that characterise the optical, geometric, and digital characteristics of the camera:

- Perspective projection.
- Transformation from the pixel frame to the image frame.
- Geometric distortion due to the optics.

The transformation from the pixel frame to the image frame is expressed as:

$$\begin{Bmatrix} u \\ v \\ 1 \end{Bmatrix} = \begin{bmatrix} 1/s_x & 0 & u_0 \\ 0 & 1/s_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{Bmatrix} \tag{2.2}$$

where $\boldsymbol{c} = (u_0, v_0)$ are the coordinates of the principal point and s_x, s_y are scale factors that correspond to the effective size of the pixels in the horizontal and vertical directions respectively. For image frame coordinates expressed in $[mm]$ the scaling factors dimensions are $\left[\frac{mm}{pixel}\right]$.

Observing coordinates of P and its projection Q expressed in the camera frame, we have (2.3) and (2.4).

$$\mathbf{x}_C^P = \begin{pmatrix} x_C^P & y_C^P & z_C^P \end{pmatrix}^T \tag{2.3}$$

$$\mathbf{x}_C^Q = \begin{pmatrix} x_C^Q & y_C^Q & z_C^Q \end{pmatrix}^T = \begin{pmatrix} \tilde{x} & \tilde{y} & f \end{pmatrix}^T \tag{2.4}$$

Establishing the equations for the similar triangles for both right-angled triangles, the first composed by the segment $\overline{O_C P}$ with the optical axis and the second one composed by the segment $\overline{O_C Q}$ with the optical axis, it is obtained:

$$\tilde{x}^P = f\,\frac{x_C^P}{z_C^P}, \qquad \tilde{y}^P = f\,\frac{y_C^P}{z_C^P} \tag{2.5}$$

Then, from (2.2) and (2.5), the coordinates in the camera frame can relate to the coordinates on the pixel frame as:

$$\begin{Bmatrix} u_H^P \\ v_H^P \\ \omega \end{Bmatrix} = \begin{bmatrix} f/s_x & 0 & u_0 \\ 0 & f/s_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} x_C^P \\ y_C^P \\ z_C^P \end{Bmatrix} \tag{2.6}$$

where the vector $\boldsymbol{x}_H^P = (u_H^P, v_H^P, \omega)^T$ are the homogeneous coordinates of Q in the image frame such that:

$$u^P = \frac{u_H^P}{\omega}, \qquad v^P = \frac{v_H^P}{\omega}, \qquad \omega = z_C^P \tag{2.7}$$

where $\omega \in \mathbb{R}$ is the scaling factor in homogeneous coordinates.

The intrinsic calibration matrix is written in (2.8) to express the transformation that relates the pixel frame coordinates of Q with the camera frame coordinates of P.

$$\boldsymbol{K} = \begin{bmatrix} f/s_x & \gamma & u_0 \\ 0 & f/s_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \alpha_u & \gamma & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

As it is seen on the right-hand side of (2.8), the terms of the intrinsic calibration matrix are grouped into α_u and α_v to reduce the notation.

To alleviate the formulation for the cases when the points are expressed in homogeneous coordinates, the intrinsic calibration matrix, $\boldsymbol{K} \in \mathbb{R}^{3\times3}$, is written using its *augmented* version $\widetilde{\boldsymbol{K}} \in \mathbb{R}^{3\times4}$ by adding a column of zeros to the right to make the dimensions agree.

$$\widetilde{\boldsymbol{K}} = \begin{bmatrix} \alpha_u & \gamma & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.9}$$

The skewing factor, that we will omit hereinafter, should be taken into account for those cases when its influence on the projection model is not negligible. This antisymmetric element of the intrinsic calibration matrix is used to represent the angle between u and v, when these two cannot be considered perpendicular.

Likewise, the optical distortion of the camera lens can be modelled under the assumption of *radial distortion* as:

$$\begin{aligned} \tilde{x}^P &= \tilde{x}_d^P \,(1 + k_1\, r^2 + k_2\, r^4) \\ \tilde{y}^P &= \tilde{y}_d^P \,(1 + k_1\, r^2 + k_2\, r^4) \end{aligned} \tag{2.10}$$

where $(\tilde{x}_d^P, \tilde{y}_d^P)^T$ are the distorted coordinates in the image frame, k_1 and k_2 are radial distortion parameters and $r^2 = \|\tilde{\boldsymbol{x}}_d^P\|$ is the square of the distance to the principal point [47]. Tangential parameters, not shown here for the sake of simplicity, can also be modelled.

2.2.2 Extrinsic parameters

The camera's extrinsic parameters identify uniquely the transformation between the world reference frame and the camera reference frame. This transformation usually consists of:

- Translation vector between the origin of the camera frame and the origin of the established world reference frame.
- Rotation matrix that brings the corresponding axes of the two camera frames into alignment.

The transformation to move from coordinates in world reference frame $\mathbf{x}$ to coordinates in camera reference frame $\mathbf{x}_C$ is mathematically written as follows:

$$\mathbf{x}_C = \boldsymbol{R} \cdot (\mathbf{x} - \boldsymbol{t}) \tag{2.11}$$

where the rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times3}$, and the translation vector $\boldsymbol{t} \in \mathbb{R}^3$.

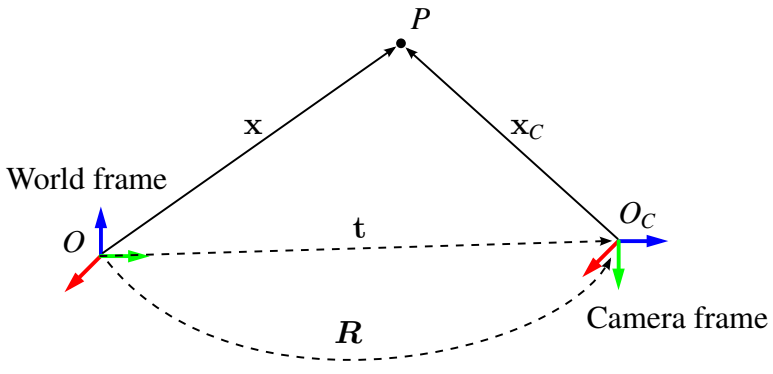


Figure 2.2: Transformation from the camera to the global reference frame.

$$\boldsymbol{t} = \begin{pmatrix} t_x & t_y & t_z \end{pmatrix}^T \tag{2.12}$$

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \\ \boldsymbol{r}_3 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.13}$$

where r_{ij} are the components of the rotation matrix and $\boldsymbol{r}_i \in \mathbb{R}^{1\times 3}$ is the i^{th} row of the matrix.

The extrinsic calibration matrix, $\boldsymbol{T} \in \mathbb{R}^{4\times 4}$ is a homogeneous transformation matrix that represents the translation and the rotation as in (2.14).

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{r}_1 & -\boldsymbol{r}_1\,\boldsymbol{t} \\ \boldsymbol{r}_2 & -\boldsymbol{r}_2\,\boldsymbol{t} \\ \boldsymbol{r}_3 & -\boldsymbol{r}_3\,\boldsymbol{t} \\ \boldsymbol{0}_3^T & 1 \end{bmatrix} \tag{2.14}$$

Multiplying the intrinsic calibration matrix and the extrinsic calibration matrix, the projection matrix is obtained to represent the transformation from the pixel frame to the world frame [46]. Using homogeneous coordinates we obtain:

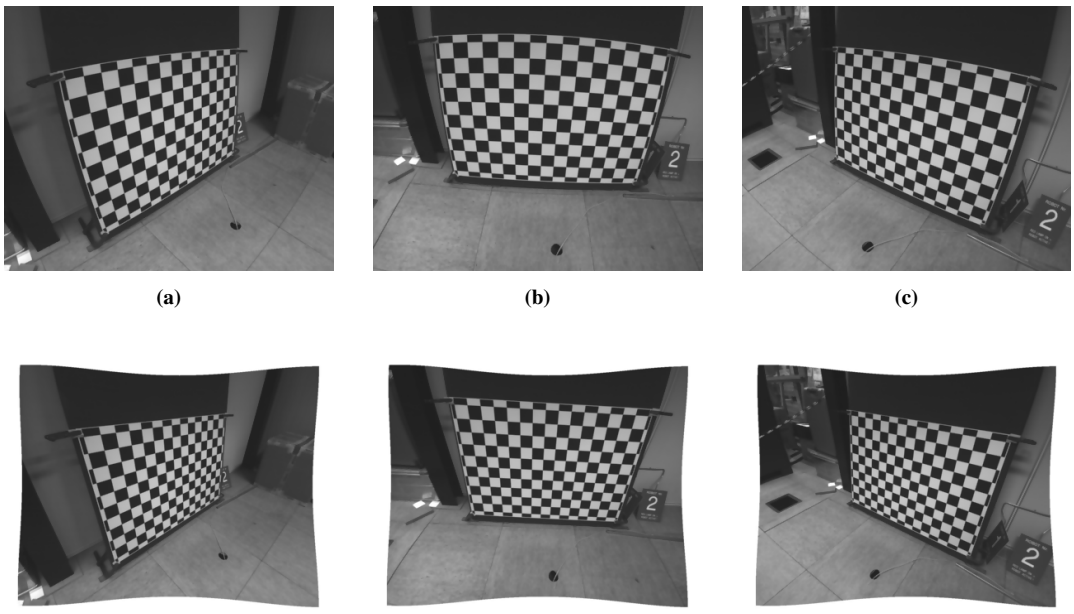
$$\begin{Bmatrix} u_H \\ v_H \\ \omega \end{Bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \cdot \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix} \tag{2.15}$$

Equations like (4.32) and (4.33) result from computing the previous product, that can be achieved through the move from the homogeneous coordinates to the non-homogeneous coordinates.

2.3 Stereo Cameras

The previous section explains the required notions to understand the projective camera model. However, the solutions presented here are based on stereo cameras devices.

The stereo systems discussed in this section are composed of two single cameras mounted on a common frame and separated by horizontal distance from each other. This horizontal distance is known as the *baseline*, *Bl*, of the stereo rig.



(a) **(b)** **(c)**

(d) **(e)** **(f)**

Figure 2.3: Example of unrectified/distorted images from calibration sequence (a), (b), (c), and their rectified/undistorted resulting images prior to being cropped (d), (e), (f).

Both the cameras composing the stereo pair are ideally identical. However, we have to take into account that when looking for accurate solutions, we have to consider that the cameras are not exactly the same.

Table 2.1: Stereo Camera calibration used for the uEye cameras composing the stereo rig on the ExoMader rover.

	Name	Parameter	Value	Std. Deviation	Units
	Resolution	width	1280		px
		height	1024		px
Intrinsic	Focal length	$\alpha_{u,L}$	830.44	$132.50 \cdot 10^{-3}$	px
		$\alpha_{v,L}$	831.12	$129.80 \cdot 10^{-3}$	px
		$\alpha_{u,R}$	829.85	$149.50 \cdot 10^{-3}$	px
		$\alpha_{v,R}$	830.44	$146.10 \cdot 10^{-3}$	px
	Principal point	$u_{0,L}$	674.25	$163.00 \cdot 10^{-3}$	px
		$v_{0,L}$	508.21	$122.10 \cdot 10^{-3}$	px
		$u_{0,R}$	643.52	$185.70 \cdot 10^{-3}$	px
		$v_{0,R}$	504.53	$136.80 \cdot 10^{-3}$	px
	Distortion coefficients	$k_{1,L}$	$-248.81 \cdot 10^{-3}$	$178.40 \cdot 10^{-6}$	-
		$k_{2,L}$	$85.64 \cdot 10^{-3}$	$237.90 \cdot 10^{-6}$	-
		$k_{3,L}$	$-20.00 \cdot 10^{-6}$	$17.30 \cdot 10^{-6}$	-
		$k_{4,L}$	$-120.00 \cdot 10^{-6}$	$17.30 \cdot 10^{-6}$	-
		$k_{1,R}$	$-245.20 \cdot 10^{-3}$	$187.20 \cdot 10^{-6}$	-
		$k_{2,R}$	$81.40 \cdot 10^{-3}$	$231.40 \cdot 10^{-6}$	-
		$k_{3,R}$	$100.00 \cdot 10^{-6}$	$19.00 \cdot 10^{-6}$	-
		$k_{4,R}$	0.00	$20.50 \cdot 10^{-6}$	-
Extrinsic	Translation	t_x	-99.60	$24.10 \cdot 10^{-3}$	mm
		t_y	$651.40 \cdot 10^{-3}$	$23.50 \cdot 10^{-3}$	mm
		t_z	$156.40 \cdot 10^{-3}$	$72.90 \cdot 10^{-3}$	mm
	Rotation	r_x	$-7.30 \cdot 10^{-3}$	$78.70 \cdot 10^{-6}$	rad
		r_y	$1.60 \cdot 10^{-3}$	$139.10 \cdot 10^{-6}$	rad
		r_z	$-3.10 \cdot 10^{-3}$	$8.80 \cdot 10^{-6}$	rad

The calibration is an accurate estimation process to determine the camera's intrinsic and extrinsic parameters. It is accomplished by taking pictures of standard calibration targets of well-known dimensions from different viewpoints. Then with this collected data, the estimation of the camera parameters is numerically computed.

Figure 2.3 shows a sample of three images extracted from the calibration set and the rectified and undistorted resulting images. The calibration and rectification processes have been conducted using the Matlab toolbox developed by Bouguet *et al.* and OpenCV Libraries [50, 51].

If the two cameras of the stereo pair are assumed to be equal, the estimation of the parameters for only one camera can be considered as a calibration approach. However, this will not allow the accurate estimation of the extrinsic parameters, responsible for representing the transformation from one imager to the other.

Table 2.1 shows the calibration results for a real stereo camera system used in our experiments, composed of the two USB 2.0 uEye cameras that constitute the stereo set mounted on the Exomader rover at European Space Technology Centre (ESTEC). The extrinsic parameters shown in the table, represent the position of the left camera centre of projection with respect to the the right camera centre of projection, so that:

$$\mathbf{x}_R = \boldsymbol{R} \cdot \mathbf{x}_L + \boldsymbol{t} \tag{2.16}$$

The rotation matrix $\boldsymbol{R}$ is the responsible for aligning the image planes of the two cameras. This alignment process, called *rectification*, guaranties the collinearity of the conjugate epipolar lines and their parallelism to the horizontal image axis.

It is important emphasising that the calibration parameters are assumed to be constant throughout the thesis. Hence, deformations on the stereo camera frame due to landing and/or significant changes in temperature will be neglected.

2.3.1 Extracting information of a 3D world

The approach that we use for our stereo odometry solution is based on the depth estimation by horizontal *binocular disparity*.

A fact in stereo vision is that the majority of the information appearing in the left image will appear for the same time-step in the right image. Nevertheless, not every point in space will be seen in both the images at the same time, due to the effect of occlusions and non-overlapping areas on the field of view. However, there will be a set of 3D points visible from both the cameras that will let us estimate the depth by triangulation.

For an ideal pair of stereo cameras perfectly aligned or more generally a pair of rectified cameras, where the camera planes $O_L X_L Y_L$ and $O_R X_R Y_R$ are coincident, we can draw the projection in the plane OXZ as shown in the Figure 2.4b.

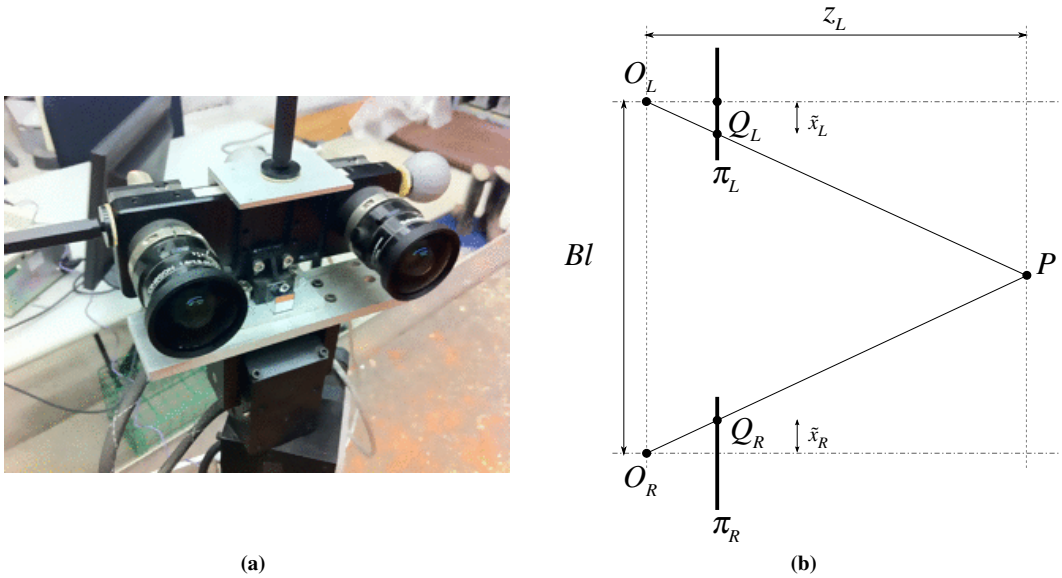


Figure 2.4: Exomader's Stereo Camera (a) and geometrical explanation for the disparity (b).

Considering similar triangles, the horizontal disparity can be easily derived as follows:

$$d(Q_L, Q_R) = \tilde{x}_L - \tilde{x}_R = f\,\frac{Bl}{z_C} \tag{2.17}$$

where Bl is the *baseline* of the stereo system, which is the segment between both camera frames origins. This equation gives the depth of corresponding points within the pairs of stereo images.

We want briefly discuss the problem of ambiguous depth estimation due to the pixel resolution, Figure 2.5. The finite resolution of digital images is the cause of depth estimation errors that grow quadratically with respect to the distance between the point and stereo camera baseline. It is seen from the top-bottom view in Figure 2.5, how all the points contained within big areas, comprising wide ranges of depth values, can only be represented by the same pair of pixels $\boldsymbol{x}_L$ and $\boldsymbol{x}_R$ on the respective stereo images.

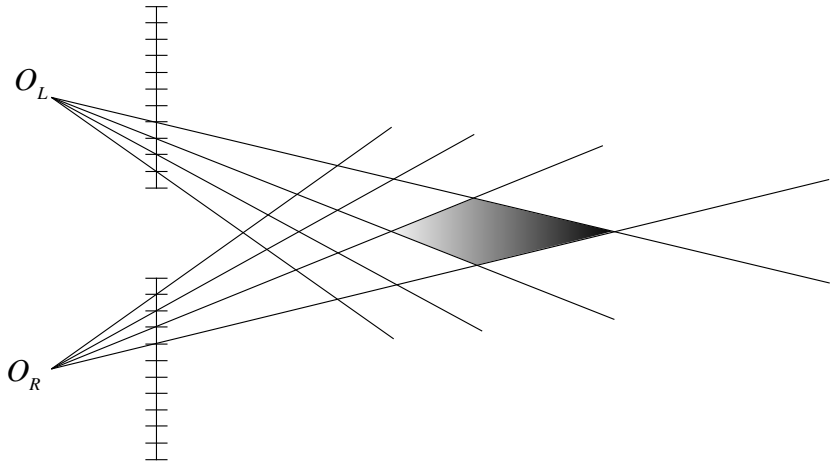


Figure 2.5: Ambiguity in the depth estimation using stereo cameras.

Effects as blurring or smearing do commonly appear when rolling shutter camera devices are employed to acquire images while in motion. To avoid these and similar effects, images are assumed to be acquired at stationary poses of the robot. Likewise, stereo images are assumed to be perfectly synchronous.

Chapter 3

Visual detection and image illumination

3.1 Overview

One may not realise the complexity of distinguishing different objects perceived using sight. This is by no means a simple task to be conducted by unsupervised machines and a very active research field.

Once raw data is acquired, either from a charge coupled device (CCD) or from a complementary metal oxide semiconductor (CMOS) image sensor, it has to be processed in order to extract meaningful information. A possible way to analyse the image information consists in *recognising* different objects appearing on the scenario. To achieve this, points within the images can be specially chosen so that a reduced number of data can contain the most important information. These points are normally referred to as *features*, *keypoints* or *interest points*.

There are several ways to select interest points from images. The most common and efficient of which are corner detection techniques [52, 53].

Finding discrete points contained within images is not sufficient to understand them. Extracted features would be useless themselves. Nonetheless, if the features are grouped or labelled they can be used for processes like matching, registration, tracking or recognition among others [54, 55, 56].

Over the last decades numerous techniques have been developed towards achieving and improving these processes. An important pioneering method precursor to later feature tracking techniques is the well known Lucas-Kanade iterative registration algorithm [57]. It would inspire the conceptualisation of current tracking techniques specially the Kanade-Lucas-Tomasi (KLT) tracker [26]. However, tracking and registration processes are only the tip of the iceberg of what can be achieved through image processing techniques for which KLT cannot be employed.

Some of the most advanced techniques that can be used for tracking and other different tasks are SIFT (Scale Invariant Feature Transform) and SURF (Speeded Up Robust Features) [28, 58]. These are robust methods that can be used for interest point extraction and identification by the introduction of local histograms called *descriptors*. Among the advantages of these later techniques is the reduction of constraints derived from the descriptors invariance to scale, rotation, contrast and brightness.

This Chapter explains how some of the most important methods found in the literature are used to detect, track and match image features, section 3.2 to section 3.6. The Chapter ends on an experimental section, section 3.9, where a new proposed technique, Harris-Moments-SURF subsection 3.8, is presented and examined justifying the need for it to be considered [59].

3.2 Harris Corners

This popular interest point detector can distinguish if the points of an image are corners, parts of an edge or are just part of a flat area. This classification is done by examining the intensity variation in the region surrounding each point in the image.

3.2.1 Method

Harris corner detector algorithm defines an auto-correlation function [60] for each point within an image (3.1).

$$c(u,v) = \sum_{W} \left[I(u^{(i)}, v^{(i)}) - I(u^{(i)} + \Delta v, v^{(i)} + \Delta v) \right]^2 \tag{3.1}$$

where $(\Delta u, \Delta v)$ is a given shift and $(u^{(i)}, v^{(i)})$ are the points in the window W. Approximating by a first order Taylor series expansion of the shifted image:

$$I(u^{(i)} + \Delta u, v^{(i)} + \Delta v) \approx I(u^{(i)}, v^{(i)}) + \begin{bmatrix} I_u(u^{(i)}, v^{(i)}) & I_v(u^{(i)}, v^{(i)}) \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} \tag{3.2}$$

where $\nabla I = [I_u \; I_v]^T$ is the spatial gradient of the image. Substituting this approximation we can derive that

$$c(u,v) = \begin{bmatrix} \Delta u & \Delta v \end{bmatrix} \begin{bmatrix} \sum_W I_u^2 & \sum_W I_u I_v \\ \sum_W I_u I_v & \sum_W I_v^2 \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} \tag{3.3}$$

This matrix representation gives an idea of the local structure surrounding each point in the image, which can be classified depending on the matrix eigenvalues:

- If both eigenvalues are small the area is flat.
- If one of the eigenvalues has a low value while the another one has a high value the area corresponds to an edge.
- If both eigenvalues are high the area is a corner.

One of the main innovation proposed by Harris to complement Moravec's previous studies [61], is the introduction of the cornerness function, ζ, used to classify the image points at lower computational expenses (3.4).

$$\zeta(\boldsymbol{x}) = \det(W) - k \cdot \mathrm{tr}^2(W) \tag{3.4}$$

where the $\boldsymbol{x}$ is and image pixel, W is the window surrounding the pixel $\boldsymbol{x}$ and k is a user-defined parameter, recommended to be chosen on the interval $[0.04 - 0.15]$ [62].

3.2.2 Advantages and Disadvantages

Repeatability Harris corner detector is defined as a robust method, in terms of invariant detection, for rotation, translation and image noise [60]. In spite of this, its robustness to scale changes may require additional enhancements.

Time Harris corner is light enough to be implemented on low resources systems.

Descriptors This corner extractor is earlier to the introduction of descriptors, thus raw intensity is used for tracking and matching purposes. This entails decrease on the computational and the memory expenses but also on the robustness.

Matching The absence of descriptors makes it difficult to achieve good matchings in order to establish correspondence between points in the image sequence.

Colour The colour information is not used in this method.

Figure 3.1, Figure 3.2, Figure 3.3 and Figure 3.4, show the interest points detected using this method over 4 sequences where there are rotation in the boat sequence, perspective changes in graffiti sequence, blurred images in the bikes sequence and illumination changes in the Leuven sequence.

Two important conclusions can be withdrawn from the detection images, where the feature detection takes place using a k parameter value of 0.04 and the pixels with a cornerness η value higher than the values specified on the figures. The first conclusion is that recurrent detected corners appear independently of the transformation or effect that takes place from one image to another. This confirms the robustness of the detector, that is able to detect the same point even when important image changes takes place.

Secondly, it is seen how the number of detected corners varies noticeably from one image to another when fixed tuning parameters are used. It is seen that the number of corners detected decreases dramatically when blurring effects are intense and when illumination - darkened images from the Leuven sequence in the case presented here - appear on the images.

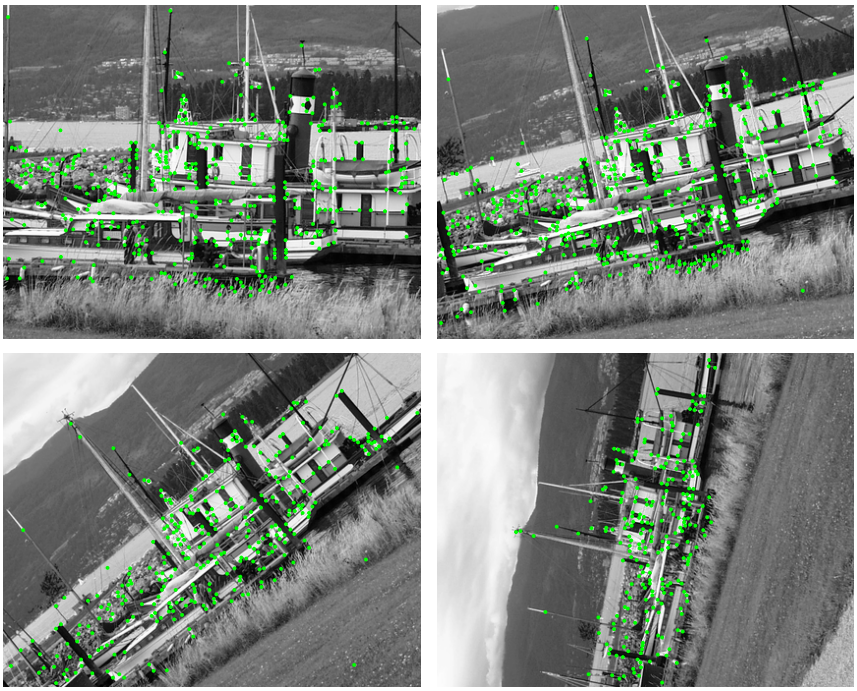


Figure 3.1: Harris Corners detection, Boat sequence. Sequence to test invariance against rotation. Green dots represent the detected corners, namely, the pixels with a cornerness value higher than 10^9, for $k = 0.04$. A total of 466, 538, 336 and 245 corners are respectively detected.

Figure 3.2: Harris Corners detection, Bikes sequence. Sequence to test invariance against blurring. Green dots represent the detected corners, namely, the pixels with a cornerness value higher than 10^7, for $k = 0.04$. A total of 846, 254, 147 and 35 corners are respectively detected.

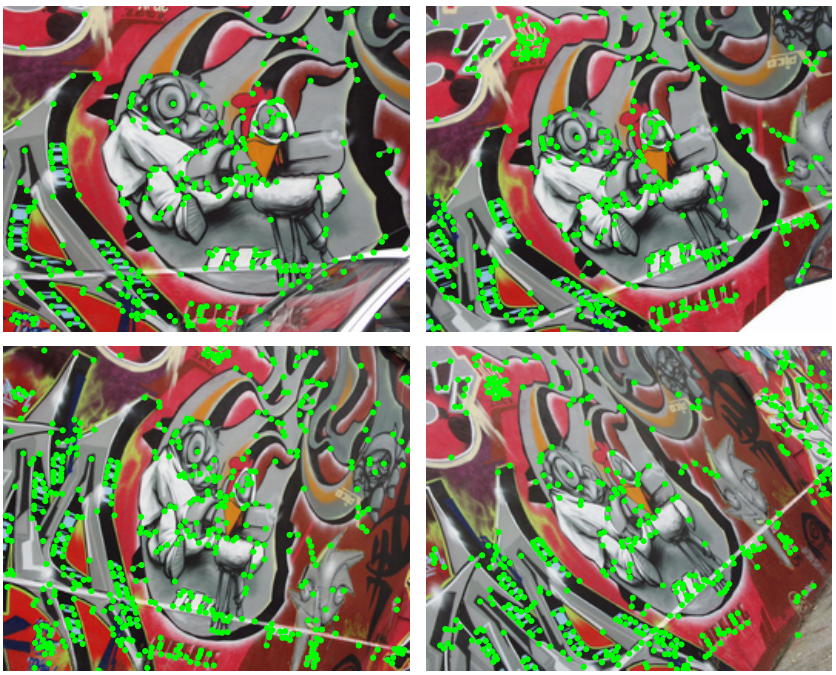


Figure 3.3: Harris Corners detection, Graffiti sequence. Sequence to test invariance against 3D projective viewpoint changes. Green dots represent the detected corners, namely, the pixels with a cornerness value higher than 10^8, for $k = 0.04$. A total of 338, 442, 511 and 456 corners are respectively detected.

Figure 3.4: Harris Corners detection, Leuven sequence.Sequence to test invariance against illumination changes. Green dots represent the detected corners, namely, the pixels with a cornerness value higher than 10^8, for $k = 0.04$.

3.2.3 Feature matching

Mathematical feature detection methods like the Harris corner detector are very useful in finding image features but do not provide information about their identity. This means that keypoints do not have any associated labels to be recognised along different images in a sequence.

Corresponding features are identified by the process of *matching*. The simplest way to do the matching of the features from image I to the features from image J is identifying each feature from I to its closest feature in the image J.

For spatial matching, the Euclidean distance between two features is computed as follows:

$$r_{i,j} = \sqrt{\left(u_1^{(i)} - u_2^{(j)}\right)^2 + \left(v_1^{(i)} - v_2^{(j)}\right)^2} \tag{3.5}$$

Then, the i^{th} feature from I, $(u_1^{(i)}, v_1^{(i)})$, is matched to the j^{th} feature from J, $(u_2^{(j)}, v_2^{(j)})$, if $r_{i,j} < r_{TH}$, where r_{TH} is a user-defined threshold.

The shortcoming of this technique is that features do not always remain on similar positions from an image to the next. This implies that the association of a feature to its closest one in the next image is simply not enough.

Other ways to do the matching is by comparing the respective neighbourhoods of the detected features. A suitable method developed to conduct this neighbourhood comparison is based on cross correlation scores (3.15) explained in section 3.7.1. Although this is a better matching strategy to the one mentioned before, there are certain situations where this technique will not be robust enough [27]. Among the possible reasons scale changes and rotations can lead to robustness related issues. Figure 3.5 displays the behaviour of cross correlation matching for the detected corners extracted using the Harris corner detection on the boat sequence, where the rotation is the main

transformation from one image to another. A total of only 38 corners is matched from one image to another. The computation of the homography transformation that defines the transformation from one image to another using the 38 corners lets us determine that only 5 of them are valid matches, showing the limitations of this matching technique.

A suitable solution for this matching problem are then the region based local descriptors. These descriptor vectors containing information of the feature's neighbourhood, not necessarily intensity information only, can be used to obtain high and reliable matching rates. SIFT and SURF are the principal methods using this description strategy.

Within the present research project we have developed a possible solution for Harris Corner matching, using the SURF region based descriptors as the providers of a robust environment information for each interest point. The results of the experiments performed can be seen in section 3.9.2

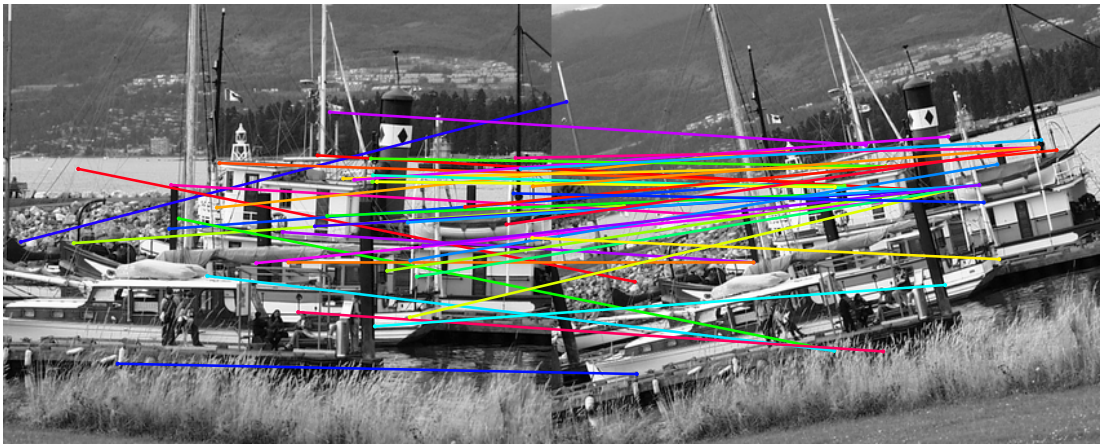


Figure 3.5: Harris Corners matching using cross correlation for the boat sequence. Only 5 out of a total of 38 matched corners are identified as correct matches when a homography transformation is computed using a Ransac algorithm. A total of 466 and 538 features are originally detected from left and right images respectively. Detection settings as in Figure 3.1

3.3 Kanade-Lucas-Tomasi Tracker (KLT)

This method was initially dedicated to tracking applications in successively aligned images from the same sequence [26, 63]. The procedure's objective is to find the displacement $\boldsymbol{d}$, usually called optical flow vector, that minimises the difference in apparent motion between two consecutive images taken from the same sequence.

3.3.1 Feature detection

As in the Harris corner detector method, KLT selects interest points as a function of the eigenvalues of a matrix related to each point [26].

$$G = \int_W \mathrm{g}\,\mathrm{g}^T w\,\mathrm{d}A \tag{3.6}$$

where $\boldsymbol{g}$ is the spatial gradient of the image I and W is the window around the point of the image. Then the selected points are just the ones, which corresponding to lower eigenvalue of G is higher than a specified threshold:

$$\min(\lambda_1, \lambda_2) > \lambda_{TH} \tag{3.7}$$

Both eigenvalues must be large to represent a corner area. Expression (3.7) is sufficient condition to select pairs of large eigenvalues when the appropriated threshold is set.

3.3.2 Alignment

The method is looking for the optical flow $\boldsymbol{d}$ such that consecutive images $I(\boldsymbol{x})$ and $J(\boldsymbol{x})$ satisfy the equation:

$$J(\boldsymbol{x}) = I(\boldsymbol{x} - \boldsymbol{d}) + n(\boldsymbol{x}) \tag{3.8}$$

where $n(\boldsymbol{x})$ is noise. The iterative algorithm that calculates the optimal displacement $\boldsymbol{d}$ to satisfy (3.9) minimises the residue error:

$$\varepsilon = \iint_W \left[I(\boldsymbol{x} - \boldsymbol{d}) - J(\boldsymbol{x})\right]^2 w(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \tag{3.9}$$

where $w(\boldsymbol{x})$ is a function to weight the image points.

The vector $\boldsymbol{d}$ is calculated by a steepest gradient descent iterative method. It is necessary to clarify that $\boldsymbol{d}$ does not have to be a unique vector for every point of the image, but for a region of it. For example, if a scene includes several objects, each object in the image can have its own independent movement, so $\boldsymbol{d}$ will be different depending on the point, or to be more precise on the point and its neighbourhood, which varies the size of the window W.

3.3.3 Advantages and disadvantages

Assumption The method is considering that the difference from an image to the next one in the sequence is small. Otherwise, it does not guarantee a successful tracking. This is due to the fact that the displacement vector is calculated in an iterative process which could not converge if the variations are big.

Iterative The optical flow $\boldsymbol{d}$ is obtained by performing an iterative process. This iterative nature means that divergence or non convergence.

Brightness The method is not robust to illumination variations as it is just considering the values of the intensities of nearby points.

Descriptors There are no defined descriptors for the tracked points on the images. This is a complete method, that calculates the alignment between the points consecutive images, so it is not necessary to use matching algorithms. This entails a lack of robustness with respect to methods as SIFT or SURF, but leads to a better time performance.

Scale The method is not robust to scale changes.

Colour The colour information is not used in this method.

Figure 3.6 show the keypoints that are detected applying the KLT over a set of images generated by the Planet and Asteroid Natural-scene Generation Utility (PANGU), which was implemented to produce representative imagery of Mars-like surfaces suitable for testing image based navigation approaches [64, 65, 66]. No further information about the generation of these images is available, however, PANGU allows tuning a number of settings to reproduce different orography and lighting conditions.

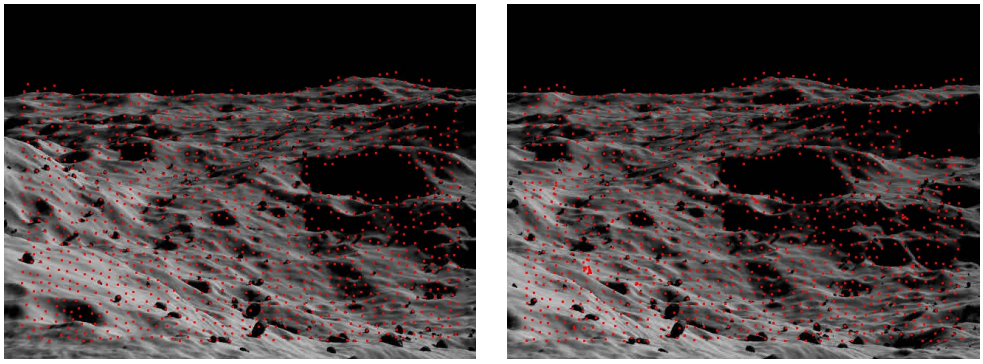


Figure 3.6: Detected interest points (red dots) over a PANGU set of images using the default parameters of Stan Birchfield's KLT implementation.

3.4 Local descriptor matching

The *matching* process is the task responsible of associating the features detected in one image with their equivalents features present in the same or, more generally, in any other image. Although some advanced techniques can be found in the literature, a rather reduced matching technique is presented here for the sake of clarity [67]. Among the possible uses of matching processes two of them are specially relevant here: matching for pairs of stereo images and matching for tracking.

In order to formulate a general feature matching technique based on local descriptors, let I and J be two images. Let us consider two sets of detected features $\{\mathcal{F}_I^{(i)}\}$ and $\{\mathcal{F}_J^{(j)}\}$ extracted from the images I and J respectively, where $i = 1, 2, \ldots, n_I$ for the features in I and $j = 1, 2, \ldots, n_J$ for the features in J.

Let us now consider that for every feature (i) in $\{\mathcal{F}_I^{(i)}\}$ and for every other feature (j) in $\{\mathcal{F}_J^{(j)}\}$ it exists a descriptor vector $\boldsymbol{\delta}_I^{(i)} \in \mathbb{R}^m$ and $\boldsymbol{\delta}_J^{(j)} \in \mathbb{R}^m$. Under these assumptions, the following matching scheme can be used to match any feature $\mathcal{F}_I^{(P)}$:

- Compute the Euclidean distance between descriptors $\boldsymbol{\delta}_I^{(P)}$ and $\boldsymbol{\delta}_J^{(j)}$:

$$\boldsymbol{\varepsilon}^P = (\varepsilon_1^P,\ \varepsilon_2^P, \ldots, \varepsilon_{n_J}^P) \tag{3.10}$$

$$\varepsilon_j^P = \|\boldsymbol{\delta}_I^{(P)} - \boldsymbol{\delta}_J^{(j)}\| \quad \text{for} \quad j = 1, 2, \ldots, n_J \tag{3.11}$$

- Ascending arrangement of the distances vector $\boldsymbol{\varepsilon}^P$ into the vector $\boldsymbol{\xi}$:

$$\boldsymbol{\xi} = (\xi_1,\ \xi_2, \ldots, \xi_{n_J}) \tag{3.12}$$

  Such that $\xi_1 = \min(\boldsymbol{\varepsilon}^P)$ and $\xi_{n_J} = \max(\boldsymbol{\varepsilon}^P)$.

The feature $\mathcal{F}_I^{(P)}$ is matched to the feature $\mathcal{F}_J^{(Q)}$ if $\xi_1 = \kappa \cdot \xi_2$, being $\xi_1 = \min(\boldsymbol{\varepsilon}^P) = \delta_Q$. The factor κ is known as the *matching threshold*.

3.5 Scale Invariant Feature Transform (SIFT)

SIFT is a complete method to locate interest points within regions in an image and to describe them on a distinctive manner. SIFT interest point descriptors are invariant enough to scale changes, to partial illumination changes, to affine changes and some 3D projective changes and to additive noises.

3.5.1 Process summary

Scale-space extrema detection The first set of keypoint candidates are calculated taking the local space extrema points found through several scales. For the scaling process the images are blurred using a Gaussian filter, such that two consecutive Gaussian images are subtracted to find the extrema. A pixel is considered as local scale-space extrema if its value is the highest or the lowest compared to its neighbour pixel values. Keypoints are found in different scales.

Keypoint localisation From the set of points obtained in the previous step, the ones smaller than a threshold value are discarded in order to keep only stable keypoints. The locations of these keypoints are recomputed more precisely and the edge responses are eliminated by analysing the curvature of the area surrounding each keypoint.

Orientation assignment An orientation is assigned to each detected keypoint based on the orientation histogram formed from the gradient orientations of a Gaussian-weighted circular window that depends on the scale where the keypoint is detected. This step is the most important in order to guarantee the rotation invariance of each keypoint.

Keypoint descriptors SIFT descriptor is a vector $\boldsymbol{\delta} \in \mathbb{R}^{128}$, that contains information about the orientation and gradient magnitude of the keypoint neighbourhood. These keypoint descriptors are used to do the matching between two image features using a matching strategy as the one explained in section 3.4.

3.5.2 Advantages and disadvantages

Scale The method is robust to scale changes as the points are detected as extrema through several scales.

Rotation The method is robust to rotation changes as each key-point has an associated distinct orientation.

Brightness The method is only partially robust to illumination variations because keypoints correspondence between two images is not based on the brightness of the keypoints only but based on the structure of the image gradient around the interest points.

Descriptors The descriptors are generated to summarise the distinctive information that let a keypoint of an image be identified.

Time Detection and description represent a significant computational effort, mostly due to scale-space feature search and descriptor matching of 128 elements vectors. Hence SIFT is expensive in a computational time sense and cannot be used for real time systems with low resources.

Colour The colour information is not used in this method.

Figure 3.7 and Figure 3.8, show the detected keypoints over 2 sequences where there are rotation and scale changes in boat sequence, perspective and illumination

changes in the graffiti sequence. Squares in green and blue indicate the curvature of the neighbourhood of a keypoint whereas red lines indicate edge-like features. Detection is performed with the default parameters of Scott Ettinger's Matlab implementation.

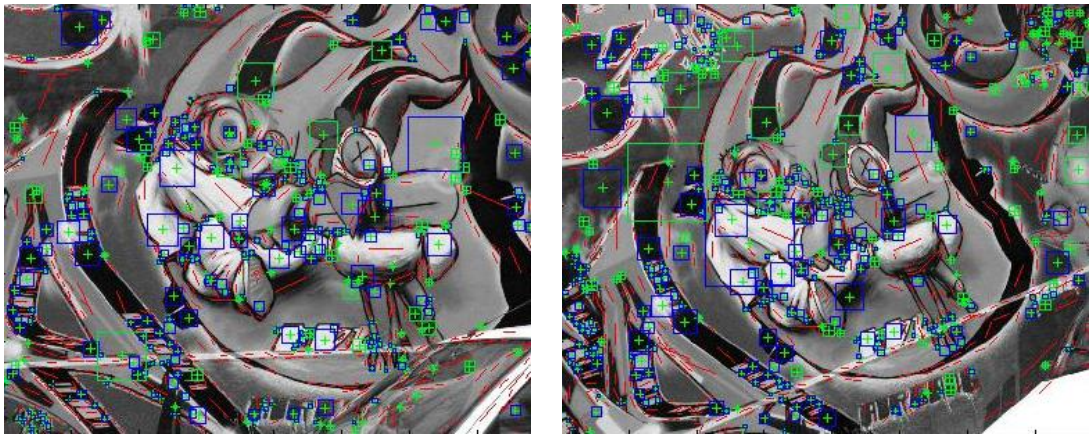


Figure 3.7: SIFT keypoints detected in Graffiti sequence images. Squares in green and blue indicate the curvature of the neighbourhood of a keypoint whereas red lines indicate edge-like features. Detection is performed with the default parameters of Scott Ettinger's Matlab implementation.

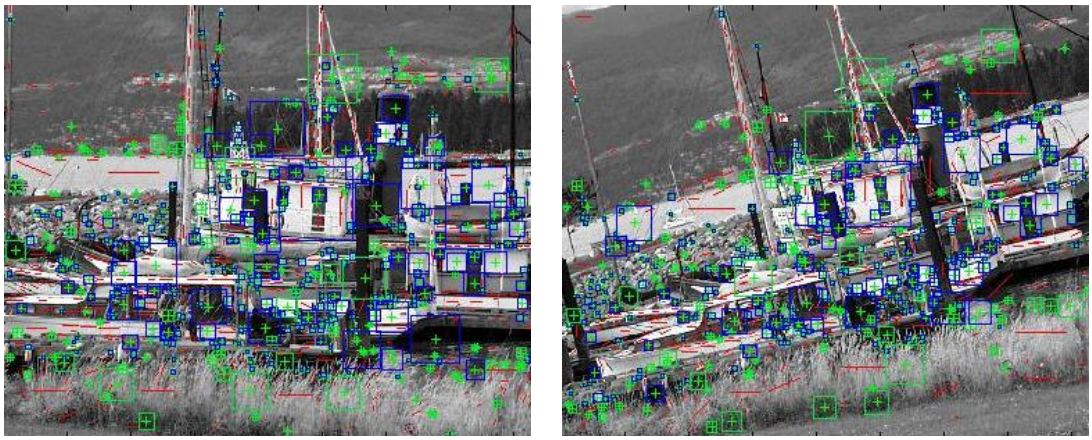


Figure 3.8: SIFT keypoints detected in Boat sequence images. Squares in green and blue indicate the curvature of the neighbourhood of a keypoint whereas red lines indicate edge-like features. Detection is performed with the default parameters of Scott Ettinger's Matlab implementation.

3.6 Speeded Up Robust Features (SURF)

SURF like SIFT is a technique to locate features within an image and to describe them. This technique was proposed after SIFT as an alternative and fast feature extraction and description method.

Detection SURF Keypoints are detected using the fast-hessian detector, which uses integral images to find local scale-space extrema locations. An Integral image I_Σ of an image I is calculated as $I_\Sigma(\boldsymbol{x}) = \sum_{i=0}^{i<u}\sum_{j=0}^{j<v} I(u,v)$, where $\boldsymbol{x} = (u,v)^T$ are the pixel coordinates [28].

Orientation assignment Haar wavelet responses are used to estimate the dominant orientation of keypoints over a circular Gaussian-weighted window.

Description Keypoint neighbourhood is split up in four subregions structure. Then its contained information is collected in a vector composed of the vertical and horizontal Haar wavelet responses of the closest points to the keypoint.

3.6.1 Differences with respect to SIFT

Descriptors SURF descriptor is a vector $\boldsymbol{\delta} \in \mathbb{R}^{64}$, in for standard SURF. The dimensionality reduction with respect to SIFT make SURF matching noticeably faster than the former.

Scale Instead of computing different Gaussian images as in SIFT, SURF uses integral images and their proprieties, which increases the speed.

Rotation Instead of performing extensive gradient orientation histograms, rotation robustness on SURF is due to Haar wavelet responses.

Figure 3.9 shows the detected keypoints with their descriptors obtained using SURF over a sample image. Green segments indicate the main orientation of the keypoints. Red circles indicate light blobs on dark backgrounds and blue circles indicate dark blobs on light backgrounds. The OpenSURF implementation from Chris Evans was used to generate this results.

In Figure 3.10 the matching between two images extracted from the same sequence is displayed. Matched features are shown drawing the segment that joins the matching pairs, whereas for detected points leading to wrong matching the segment is omitted.

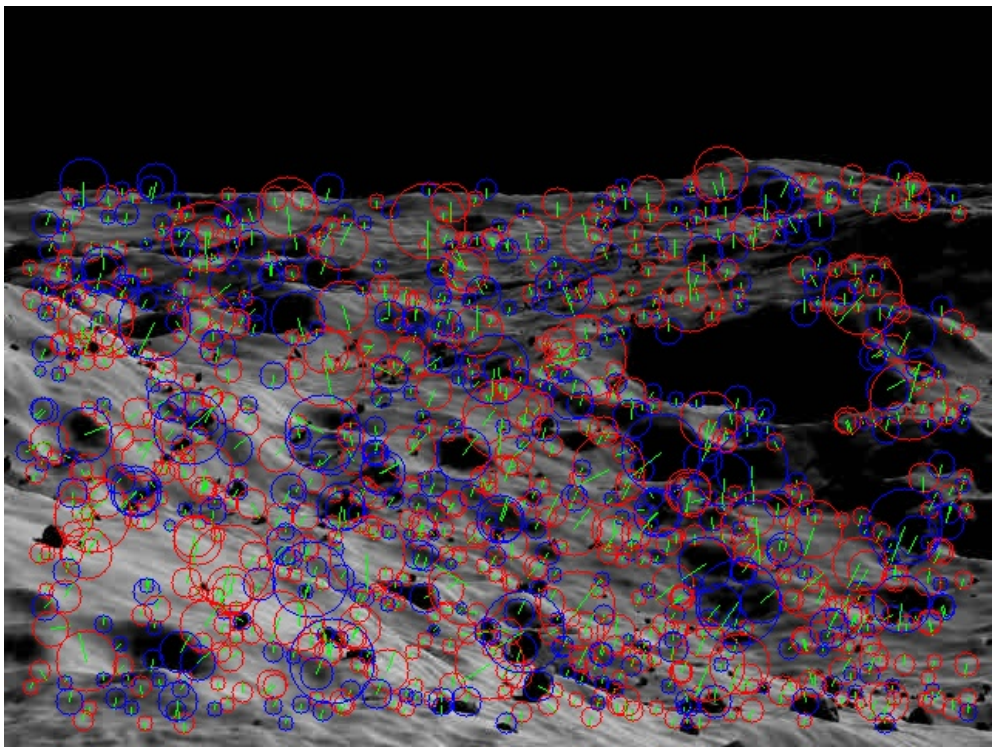


Figure 3.9: SURF keypoints detected in PANGU images. Green segments indicate the main orientation of the keypoints. Red circles indicate light blobs on dark backgrounds and blue circles indicate dark blobs on light backgrounds.

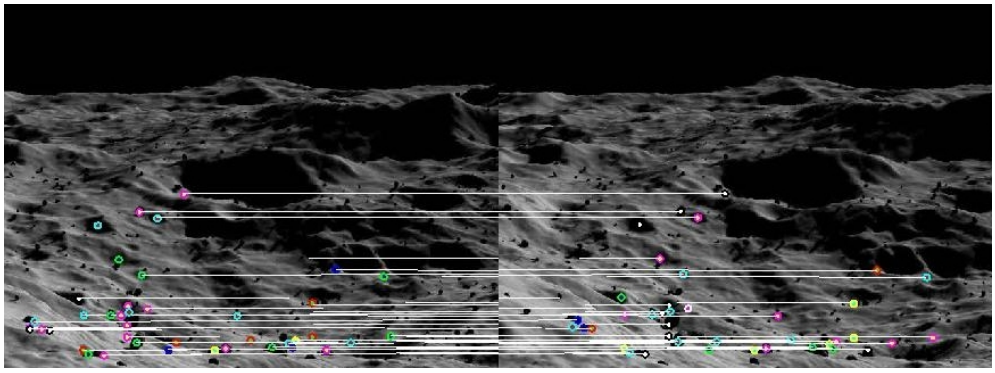


Figure 3.10: SURF keypoints matching, PANGU images. A horizontal motion of the camera takes place from the image in the left to the image in the right.

3.7 Moment image representation

On-board camera systems are exposed to illumination changes along time and space. As a robot equipped with a stereo camera rig moves, the relative position of the cameras with respect to the light sources will change. This induces differences not only between the left and right images of the stereo pair, but also between pairs of images acquired at different times.

To address this, certain measures are taken to diminish the effect of brightness changes on the imagery. One way to mitigate the effects of illumination changes is based on image moment representation.

The geometric moment of the image pixel $I(u,v)$ with respect to a square neighbourhood of size $(2N+1)$, is computed in (3.13), as shown in [68].

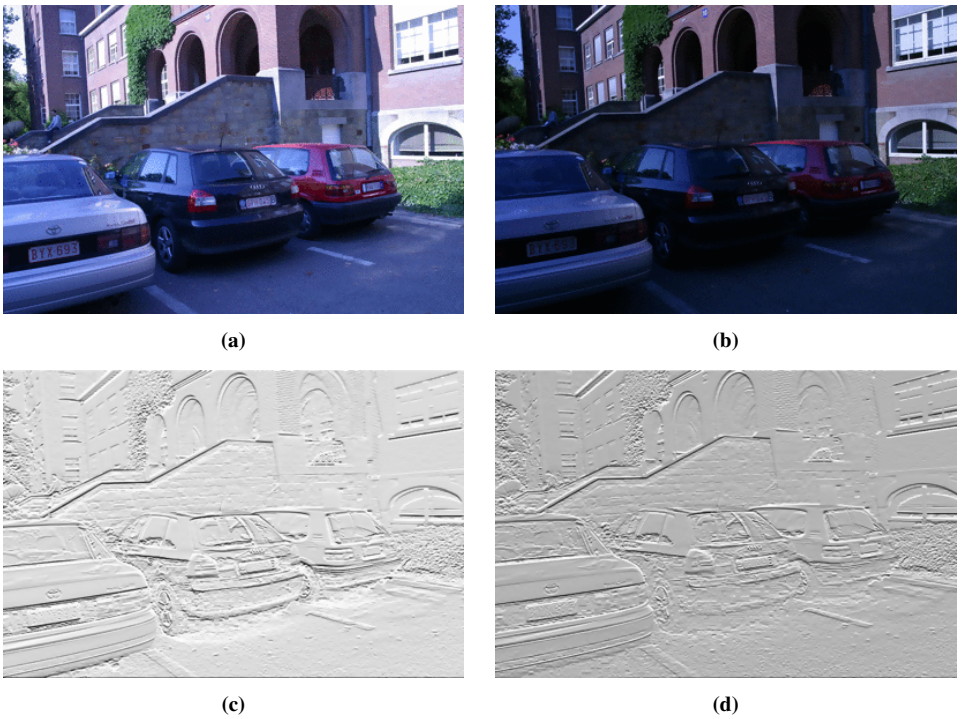
$$m_{pq}\big(I(u,v)\big) = \sum_{i=-N}^{N} \sum_{j=-N}^{N} i^p \, j^q \, I(u-i, v-j) \tag{3.13}$$

where p and q are the orders of the moments in the corresponding axis. Let us point to the reader that a square window of size $(2N+1) \times (2N+1)$ around the pixel of coordinates (u,v) is used on this computation.

Due to the additional invariance to illumination provided by image moments [59, 69, 70], an alternative representation of the scene images is constructed, namely *moment images*. A moment image $J(u,v)$ is expressed as a combination of different image moments of the original image $I(u,v)$. Equation (3.14) shows the function chosen to compute the moment images.

$$J(u,v) = \frac{m_{01}\big(I(u,v)\big)}{m_{20}\big(I(u,v)\big)} \tag{3.14}$$

The parameters p and q used in (3.14) for the computation of moments in numerator and denominator have been empirically selected for our purposes here. However, these values could be further investigated in future research.



(a) **(b)** **(c)** **(d)**

Figure 3.11: Original images (a), (b) and moment images (c), (d), on the dataset "*Leuven*", where illumination conditions dramatically change. Moment images are computed for a window of side 5.

One of the benefits of the moment images is the reduced computational cost obtained through composition of simple image filtering. Given the above definitions, it is possible to prove computational complexity is $O(mn)$, where n is the number of image pixels and m is the number of elements on the convolution kernel used on the filtering stage, equal to $(2N+1)^2$ according to the definition given in (3.13).

The expectation is to increase the feature matching rates when illumination effects are non-negligible on a dataset, as it often happens on robotic navigation applications.

Figure 3.11 presents an example of moment images calculated for two of the images from *"Leuven"* series , used by Mikolajczyk *et al.* [69] to test robustness of features against light changes.

3.7.1 Harris corners and moment images

Experiments have been conducted to compare what happens when feature detection is done over images of a sequence and what changes if detection is done over moment images of the same sequence. This section tries to objectively answer to the questions *"is it better to detect the features over moment images rather than real images when there are illumination changes?", "is it better when there is also motion added to the illumination changes?"*

The maximum number of good matches that can be obtained using Harris corner detector over the images and over the moment images is a good criterion to compare both type of detections. A cross correlation model is used (3.15) to calculate the matching error between every detected corner neighbourhood in the first image with respect to every corner neighbourhood in the subsequent image.

$$\varepsilon = \sqrt{\sum_{i=-N}^{N}\sum_{j=-N}^{N}\big(I(u+i,v+j)-J(u+i,v+j)\big)^2} \tag{3.15}$$

where $(2N+1)\times(2N+1)$ is the size of the neighbourhood compared, and $I(i,j)$ and $J(i,j)$ are the value of its contained pixels.

For the self–matching tests, Test 1 and Test 2, image pairs with no motion between them were used as pair of images. These either formed by an image and the same one again, or an image and its illuminated image version. Therefore, the position of every feature $\mathcal{F}^{(i)}$, found in the first image I, whose coordinates are $\boldsymbol{x}^{(i)} = (u^{(i)}, v^{(i)})^T$, has to be matched with the feature j, found in the second image, J, whose coordinates are, $\boldsymbol{x}^{(j)} = (u^{(j)}, v^{(j)})^T$, if and only if $\boldsymbol{x}^{(j)} = \boldsymbol{x}^{(i)}$. The rest of matches will be considered as false matches.

For cases when there is motion, Test 3, every feature i, found in the first image I, whose coordinates are $\boldsymbol{x}^{(i)} = (u^{(i)}, v^{(i)})^T$, is considered matching with the feature j, found in the second image, J, whose coordinates are, $\boldsymbol{x}^{(j)} = (u^{(j)}, v^{(j)})^T$, if and only if $\boldsymbol{x}^{(j)} = \boldsymbol{x}^{(i)} + (\Delta u, \Delta v)^T$, where $(\Delta u, \Delta v)^T$ is the vector used to represent the motion of the image points from image I to image J. The rest of matches will be discarded.

Table 3.1: Detection comparison results, by matching.

	Units	**Test 1**		**Test 2**		**Test 3**	
		Original	Moments	Original	Moments	Original	Moments
n	[#]	163	161.5	163	161.5	165	162
tp	[#]	161	160.25	66.5	132	4	6
ρ	[−]	1.00	1.00	0.83	0.99	0.02	0.04
σ	[−]	0.99	0.99	0.41	0.82	0.02	0.04

Table 3.1 shows the results obtained for this experiment, where n is the *number of detected corners*, tp is the *number of good matches* or *true positives*, ρ is the *precision* of the matching process, given as the rate $\rho = tp/(tp+fp)$, being fp the *false positives*. The *matching rate* σ has been computed as: $\sigma = tp/n$, dividing the number of total good matches over the number of features that were found. All the displayed values are averages obtained for the different tests. The values on the left side are for the detection over the images while the values on the right side have been obtained for the detection over the moment images.

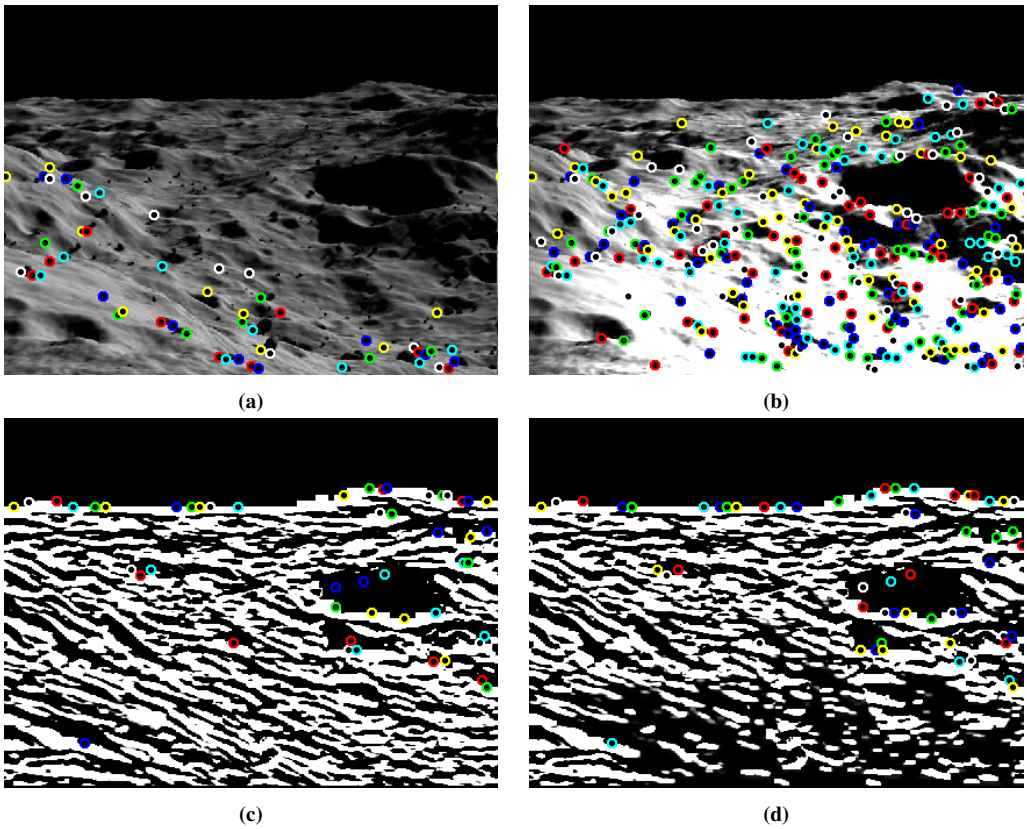


Figure 3.12: Constant threshold features detection. Original images (a) and (b), 53 and 329 features are respectively found. Moment images (c) and (b), 50 and 57 features are respectively detected.

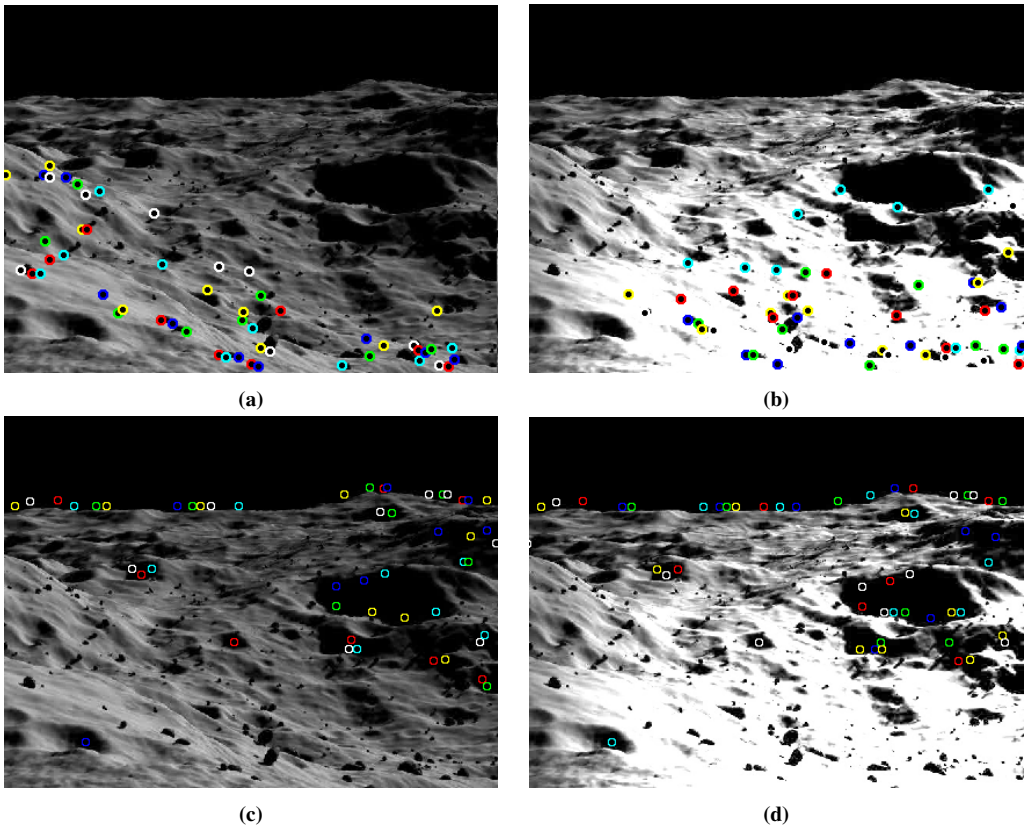


Figure 3.13: Detection of 50 features, over original images, (a) and (b), and over moment images, (c) and (d). Detection results are only represented over the original images here.

By analysing these results, we can confirm that matching using Sum of Square Differences (SSD) technique of features found over the moment images is better for all experimental Tests since the rates ρ and σ are always higher. Indeed, for Test 2, which includes illumination effects, the matching rate increases to twice as much as when the detection is performed over moment images and the rate of good matches increases with around 15%. When there is motion in addition to illumination changes as for Test 3, the results are slightly better as well.

The images in Figure 3.12 show, how the number, and location, of features found over original images changes when there is an illumination effect altering the image (a)

and (b), while the number, and location, of detected of features found over moments images remains similar (a) and (b) . I addition, the number of features detected in illumination altered real images gets lower in comparison with the original non altered real image. This number keeps similar when we process detection on moment images.

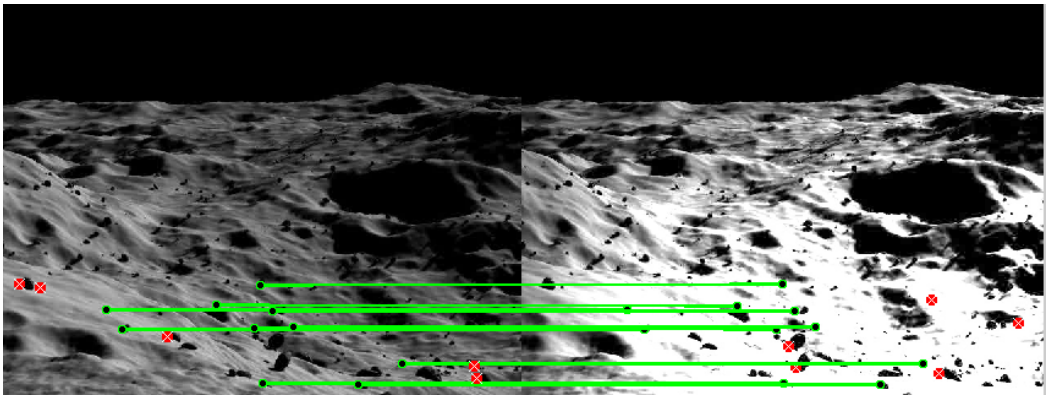
Figure 3.13 shows the detected features obtained using each type of images, when only the 50 features with higher cornerness function response are kept. Note that the keypoints are located in similar areas when detection takes place over the moment images, but on the contrary it changes considerably because of the illumination effect when detection is performed over the original images.

The images in Figure 3.14 display the matches obtained for a self-matching pair with illumination changes. Matches are obtained using a cross correlation matching strategy for the features obtained with a Harris corner detector over original images (a), and over moment images (b). The red coloured features are discarded false matches.

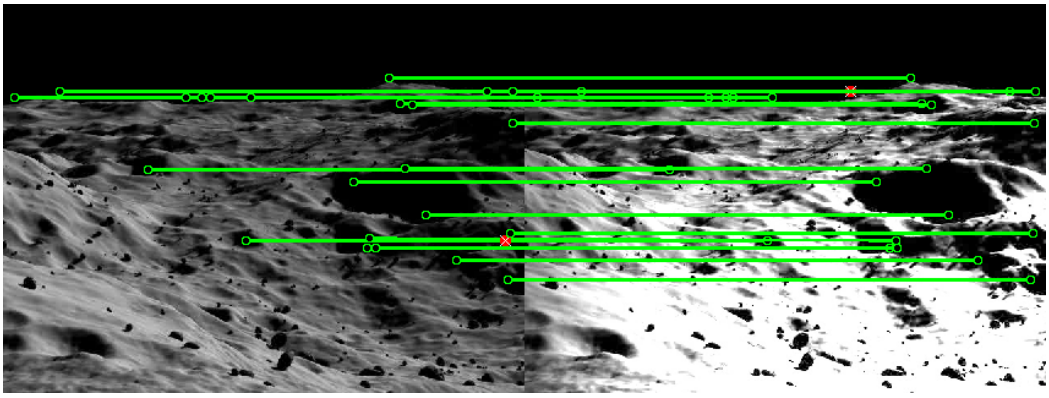
These illuminated images are artificially generated adding 15 units of brightness and a brightness increment proportional to the pixel distance to the top left corner times 300. These setting make the images to be illuminated as if a light focus was added on the right bottom corner.

3.8 Harris over Moment images with SURF (HMSURF)

Harris corner detector presents many advantages for the feature detection task [60], it is easy to implement and a fast technique. However, it is necessary to use additional tools to carry on with the matching and identification tasks as it has not got any descriptor associated to it. The lack of descriptors makes Harris corner detector be less complete than SIFT or SURF and also less robust against affine and scale changes in the scene.



(a)



(b)

Figure 3.14: Cross correlation matching results of features detected over original images (a) and features detected over moment images (a). Features in red colour indicate rejected false matches. Detection results are only represented over the original images here.

We present an alternative to augment Harris detector with a robust descriptor in order to achieve better matching than matching Harris corners using correlation based on Sum of Squared Differences (SSD). Thus based on our initial feature experimental analysis, subsection 3.9.1, we have decided to integrate the SURF description and matching process with the detection process carried out by Harris detector. SURF has been selected because of the similarly performance achieved in less time, in comparison to SIFT. In case of pseudo-planar motion the reduced version of SURF description, U-SURF, where the main orientation does not have to be computed, is used.

The alternative representation based on the moment images, presented in section 3.7, is used in the Harris detection step in order to improve invariance to illumination changes, as it is a way to emphasise corners. The description and matching processes are run over the original images containing the corners detected in the previous step.

3.9 Experimental results

This section presents some results comparing the methods exposed in this chapter, in order to justify the decisions taken for the implementation of HMSURF. The later experiments are meant to analyse the proficiency of the new presented technique and to compare it with SURF.

3.9.1 Comparison of existing methods

In order to compare the different methods for tracking and feature detection, it is necessary to establish some criteria to perform a realistic comparison. In this way, for comparison between KLT, SIFT and SURF we have used codes programmed in C/ C++ languages, all of them implemented using OpenCV libraries for image processing in order to have comparable time scale measurements [51, 71, 72, 73]. Harris corner detector results are not shown in this section because of the low quality results obtained for this method. This is a consequence of the reliability limitations of cross correlation matching using raw intensity vectors.

We have considered for the experiment a set of 6 images generated through PANGU [64, 65, 66]. This set of images contain different problems that can be studied in the tracking problems:

- Translation
- 3D projective changes
- Out-of-plane changes
- Brightness variations

According to the obtained results for this set of comparisons, we can present some conclusions, section 3.9.1.1, based on the study of the next tables of results.

Table 3.2 shows the number of tracked features obtained for consecutive images and time expenses involved in the computation. The system is set to detect 2500 features to track. Table 3.3 and Table 3.5 show the number of detected features using SIFT and SURF respectively, while Table 3.4 and Table 3.6 show time consumptions for these methods. These time consumptions were obtained on a laptop with an Intel Centrino CPU (1.6GHz), single core, on Ubuntu linux (9.10).

Table 3.2: KLT time consumption on 1 core CPU 1.6GHz.

Tracked Features [#]	**Total time** [10^6 clocks]
280	1.66
340	1.64
283	1.70
260	1.67
257	1.65

Table 3.3: Keypoints detection using SIFT.

Image [#]	Features [#]
1	2596
2	2553
3	2672
4	2672
5	2559
6	2665

Table 3.4: Time consumptions using SIFT on 1 core CPU 1.6GHz.

Matches [#]	Detection [10^6 clocks]	Description [10^6 clocks]	Matching [10^6 clocks]	Total time [10^6 clocks]
650	3.44	3.38	0.64	7.46
767	3.37	3.48	0.63	7.52
704	3.49	3.48	0.67	7.64
766	3.50	3.44	0.68	7.62
771	3.45	3.44	0.68	7.57

Table 3.5: Keypoints detection using SURF.

Image [#]	Features [#]
1	2315
2	2271
3	2342
4	2441
5	2405
6	2555

Table 3.6: Time consumptions using SURF on 1 core CPU 1.6GHz.

Matches [#]	Detection [10^6 clocks]	Description [10^6 clocks]	Matching [10^6 clocks]	Total time [10^6 clocks]
1109	1.80	1.77	1.26	4.83
1262	1.83	1.83	1.29	4.95
1129	1.82	1.85	1.34	5.01
1282	1.87	1.84	1.38	5.09
1309	1.92	1.96	1.47	5.35

3.9.1.1 Analysis

Similar to experiments used by Mikolajczyk *et al.* [74], our results prove that SURF is more robust than SIFT or previous methods as mentioned in [28].

Next subsections are written to set the performance differences in terms of time and proficiency of SIFT, SURF and KLT methods, using the results, subsection 3.9.1.

3.9.1.2 Time performance

Looking at the tables of the times obtained from the experiments done with each implementation, it can be concluded that:

Best time Because of the lack of descriptors, and according to the results exposed in the previous tables, KLT method could be thought to be faster than the other two method. Nonetheless, note that the number of tracked features in KLT is two to four times lower than the number of matched interest points obtained using SIFT and SURF respectively. Considering that relationship between the number of tracked features behave on a linear fashion for the three of the methods, it is easy to see that SURF is the fastest method, followed by KLT and finally by SIFT.

Detection Looking at the average of the feature detection, which is actually added to the descriptors generation time, we can conclude that the detection time of SURF is almost a half of the time for SIFT execution.

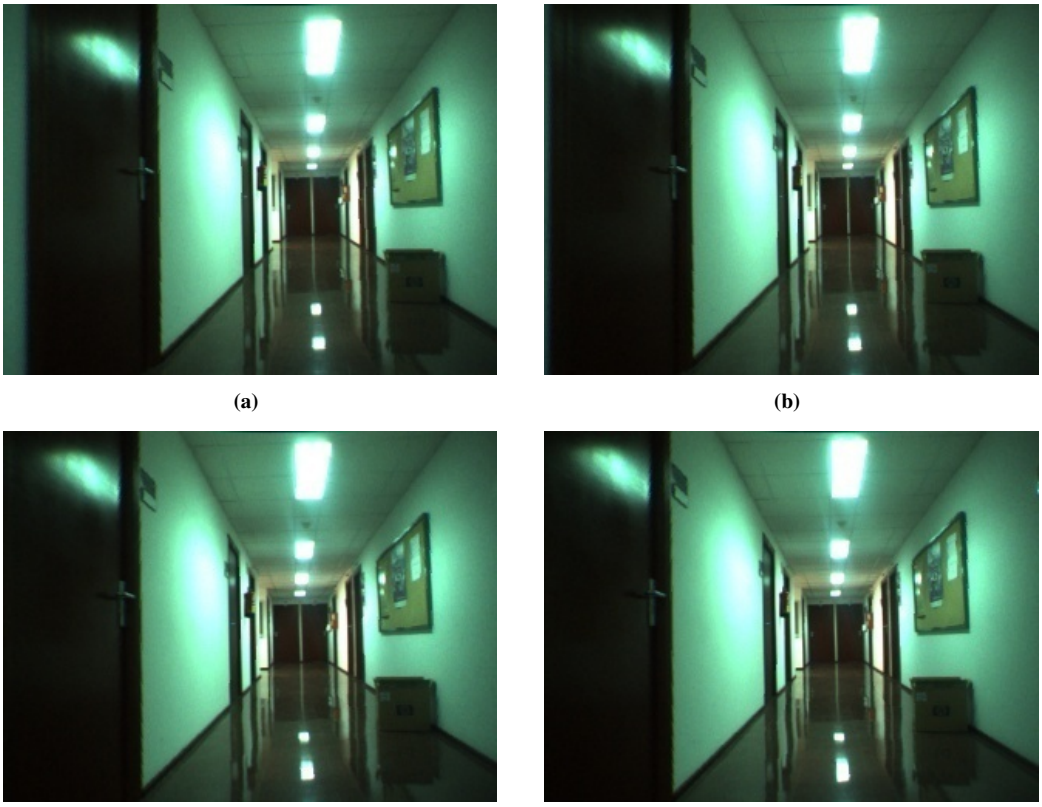
Matching SIFT Matching is around a half of the time needed for the SURF matching. This is because the matching methods are different in the used implementations and the number of matches obtained are different (more than double for SURF). We also know that SURF descriptor are half size of SIFT descriptor, 64 elements with respect to the 128 for SIFT.

Total time The calculations done in our experiments show that SURF is at least a 30% faster than SIFT, using a very similar number of Keypoints for the matching. In the literature [28], we find that SURF just takes the 30% of the whole time used by SIFT.

A second experiment was conducted using the three methods for a sequence of images acquired from an on-board camera mounted on a mobile robot in the University of Malaga, Moreno *et al.* [75].

Table 3.7: Average time consumptions for the different techniques on 1 core CPU 1.6GHz. The number of detected point is similar for all the techniques.

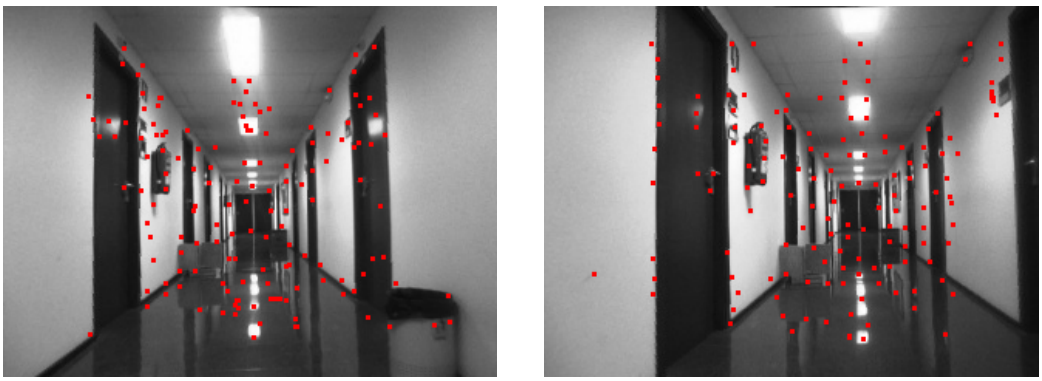
Method	**Detection** [10^6 clocks]	**Description** [10^6 clocks]	**Matching** [10^6 clocks]	**Total time** [10^6 clocks]
KLT	-	-	-	2.69
SIFT	7.70	7.57	0.51	15.77
SURF	2.03	2.03	0.01	4.01



(a) (b)

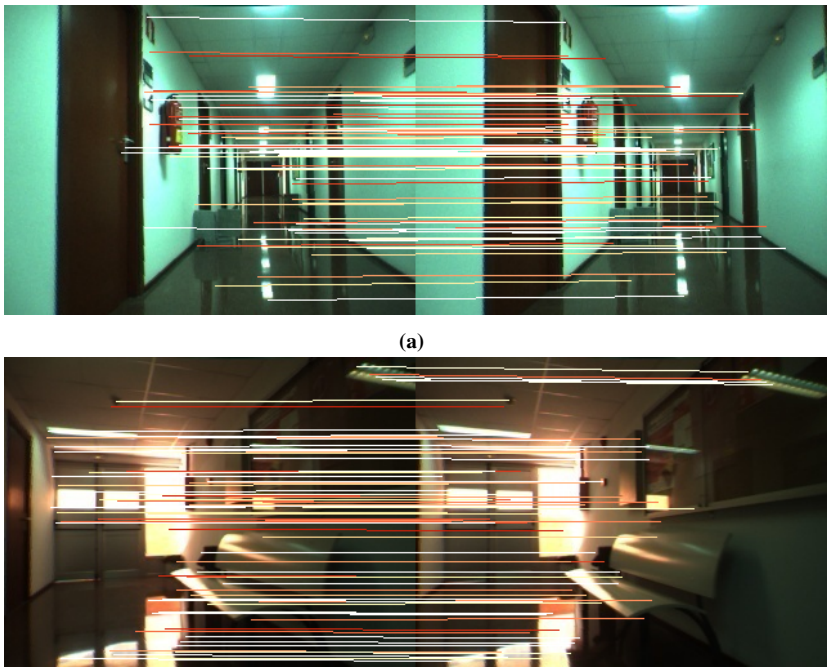
(c) (d)

Figure 3.15: Sample from Corridor 2.2 dataset, University of Málaga, ETSI Telecomunicación.



(a) (b)

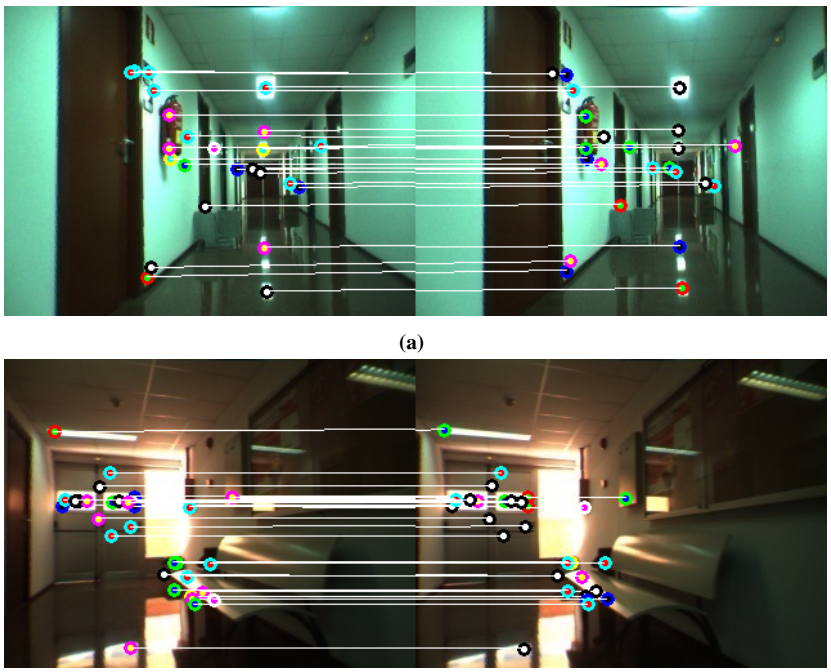
Figure 3.16: KLT tracked points, Corridor 2.2 dataset, University of Málaga, ETSI Telecomunicación.



(a)

(b)

Figure 3.17: SIFT matching, Corridor 2.2 dataset, University of Málaga, ETSI Telecomunicación.



(a)

(b)

Figure 3.18: SURF matching, Corridor 2.2 dataset, University of Málaga, ETSI Telecomunicación.

The frame rate on this later sequence guarantees small differences between subsequent images, which makes the sequence conditions more favourable for KLT to perform correctly. The figure Figure 3.15 shows 4 consecutive images extracted from the sequence.

Figure 3.17 and Figure 3.18, show the matching using SIFT and SURF methods respectively, while Figure 3.16 shows just the detected keypoints using KLT.

As in the previous experiment Table 3.7 shows that KLT and SURF are faster techniques than SIFT for a similar number of detected features. This is normally attributable to the size of the descriptor used by SIFT, that entails higher time consumptions, specially at description and matching stages.

3.9.1.3 Matching performance

Looking at Table 3.2, Table 3.4 and Table 3.6 obtained as result of the performed experiments with each studied technique, it can be concluded that:

KLT The number of tracked features using this method is very low, reaching values just around the 10% of the initial interest points detected, which was fixed to be about 2500 for every technique. The violation of the assumption of small differences between consecutive images necessary for KLT to perform rightly is a suitable explanation for the small rate of tracked features presented here. Indeed, in the original experiments performed by Tomasi *et al.* translations of only one pixel from an image to the subsequent image in the sequence were considered [26]. It is visible to the eye that the translation applied to the images on the set of tested images is always greater than one pixel.

SIFT This method has a matching rate of about a 25% with respect to the interest points detected for each picture. This result is not realistic without taking into account that some of those matches are false.

The usage of descriptors means performing a better tracking from one image to the next one in the sequence.

SURF The matching rate for SURF was around a 50%, with the lowest rate of false matches. This is due to the nature of the images and SURF which was developed to outperform SIFT using descriptors based on Haar wavelets responses.

It is shown by the results obtained that even considering a lower number of keypoints, SURF matches from one image to the subsequent one presents high numbers.

3.9.2 Comparison between HMSURF and SURF

This experiment that has been carried out to study the performance of the proposed robust HMSURF technique concentrates in comparing it with the original SURF technique.

For this experiment, several tests have been conducted in order to study the time consumption required for the HMSURF and for the original SURF. As important as time consumption, the matching and reliability performances between a pair of images using HMSURF and SURF need to be evaluated. As explained above, in this work we focused more on the robustness an invariance against illumination changes. This matching performance is compared by looking at the maximum number of features that can be matched and by looking at other relevant matching rates as well.

Table 3.8: HMSURF vs SURF.

	Units	Test 1		Test 2		Test 3	
		SURF	HMSURF	SURF	HMSURF	SURF	HMSURF
n	[#]	351	488	366	488	327	305
tp	[#]	351	488	110	349	67	79
m	[−]	0.00	0.00	0.30	0.16	0.81	0.75
η_ρ	[−]	0.72	1.00	0.32	1.00	0.85	1.00
τ	[10^5 clocks]	3.72	4.51	3.73	4.51	3.72	3.94
η_τ	[−]	1.00	1.21	1.00	1.21	1.00	1.06

Table 3.8 presents the results obtained running both SURF and HMSURF techniques. As in Table 3.1, tp is the number of the *good matches* found from the of n *detected features*, $m = (1-\rho)$ is the *mismatching ratio* computed as $m = (1-\rho) = fp/(tp+fp)$, the rate η_ρ is calculated as: $\eta_\rho = tp/\max(tp)$, τ is the total time required for the whole process, detection, description and matching, in computer clocks multiplied by 10^{-5} and the rate η_τ is computed as: $\eta_\tau = \tau/\min(\tau)$. It seems important to the author emphasising on the importance of the mismatching ratio m. This rate evaluates the reliability of matches. From Table 3.8 it can be seen that for the cases without motion, Test 1 and 2, the proposed technique gives the same or even better results than the achieved by the original SURF method. For Test 3 case a slight enhancement can be seen, as well.

Time requirements seem to be a slightly higher to similar for the proposed technique on the image test used. Nevertheless, if results in Table 3.8 are meticulously studied we will see that the number of good matches obtained using the proposed is always bigger. Assuming the relationship between the time expenses and the number good matches to be linear, both for SURF and HMSURF, it is easy to empirically demonstrate that SURF and HMSURF computational complexities are comparable.

3.10 Conclusions

In this chapter, a new solution to cope with illumination changes and feature matching problem related to Harris corners has been presented. The proposed robust Harris-Moments-SURF detector/descriptor provides an alternative that fuses the advantages of Harris corner detector over moment images and SURF descriptor. It has been shown that this alternative performs better than some other state-of-the-art methods. Certain conditions that adversely affect the visual guided navigation, such as illumination changes, can be dealt with using this proposed solution.

Chapter 4

Enhancing Visual SLAM techniques

4.1 Overview

For missions similar to the ExoMars mission, the ability to autonomously navigate in an environment about which the rover has no prior knowledge is a desired capability. This permits to the robots to accurately determine their own position and orientation and to plan paths towards their goals.

As Global Positioning System (GPS) data is unavailable on the planet Mars and because robot odometry data is not fully reliable for long term navigation, other exteroceptive solutions are required [76, 77]. A possible approach is the Simultaneous Localisation and Mapping (SLAM) technique. SLAM is the process by which a mobile robot can incrementally build a consistent map of its environment and at the same time use this map to compute its location. A variety of sensors such as laser scanning systems have been used in SLAM for robotic platforms [6, 43]. SLAM has also been used in both indoor and outdoor for ground, underwater and airborne applications

[4, 78, 79].

As vision is one of the main means of exteroceptive sensing, VSLAM based on monocular camera approaches have been investigated previously [45, 80, 81, 82]. On the other hand VSLAM solutions based on stereo cameras have also been looked at [8, 83]. As the rover is meant to be equipped with a stereo head, our interest was naturally oriented to investigate stereo based VSLAM approaches.

Different filtering based approaches have been adopted in SLAM [14, 84] and VSLAM solutions [75, 85]. Most of the suggested estimation filters used are nonlinear. This is necessary mainly because image variations along a sequence cannot be considered as linear variations, making VSLAM observation models highly nonlinear. The question of which filtering technique is recommended for VSLAM has been addressed in the past [11, 20]. However, only little care was given to the visual processing and the detection of visual natural landmarks to build the VSLAM and more specifically stereo based VSLAM solutions [86, 87].

In this chapter, we propose an EKF-VSLAM solution relying on a tight stereo observation model and an innovative feature extraction technique based on Harris Moment Speed Up Robust Features (HMSURF), explained in the previous chapter. Section 4.2 and section 4.3 introduce the filtering techniques used. All the visual processing including feature detection, feature matching and landmark management is addressed in section 4.4. At the end of this section, in subsection 4.4.5 and subsection 4.4.6 contributions to the VSLAM visual module are proposed. Experiments shown in section 4.5 demonstrates the advantages of the proposed enhancements.

4.2 Kalman Filter

The Kalman Filter (KF) is a recursive algorithm used to estimate the state of a timely-controlled dynamic system. It is proved an optimal filter for linear systems [9], where sensor measurements are corrupted by white noises. The discrete version of this famous filter has been the subject of extensive research and applications in the area of autonomous systems.

One of the main advantages of the Kalman filter is that the noisy data acquired from different sensors can be joined taking into account the reliability and uncertainty of each.

The system's dynamic process is governed by the stochastic difference equation:

$$\mathbf{x}_{k+1} = \boldsymbol{F}_k \cdot \mathbf{x}_k + \boldsymbol{G}_k \cdot \mathbf{u}_k + \mathbf{w}_k \tag{4.1}$$

where:

- $\mathbf{x}_k \in \mathbb{R}^n$ is the state vector that represents the system at the discrete time-step k
- $\mathbf{u}_k \in \mathbb{R}^m$ is the input vector
- $\boldsymbol{F}_k \in \mathbb{R}^{n \times n}$ is the state transition or process matrix of the system
- $\boldsymbol{G}_k \in \mathbb{R}^{n \times m}$ is the input matrix
- $\mathbf{w}_k \in \mathbb{R}^n$ is a random variable $\mathbf{w} \sim N(\mathbf{0}, \boldsymbol{Q}_k)$ that represents the process noise.

Furthermore the evolution of the system is observed through the measurements vector $\boldsymbol{z}$.

$$\boldsymbol{z}_{k+1} = \boldsymbol{H}_{k+1} \cdot \mathbf{x}_k + \mathbf{v}_{k+1} \tag{4.2}$$

where:

- $\boldsymbol{z}_k \in \mathbb{R}^m$ is the measurements vector at time-step k
- $\boldsymbol{H}_{k+1} \in \mathbb{R}^{m \times n}$ is the observation matrix
- $\mathbf{v}_k \in \mathbb{R}^m$ is the measurement noise $\mathbf{v} \sim N(\mathbf{0}, \boldsymbol{R}_k)$.

The covariance matrices $\boldsymbol{Q}_k \in \mathbb{R}^{n \times n}$ and $\boldsymbol{R}_k \in \mathbb{R}^{m \times m}$ represent the uncertainties of the process model and the observation model respectively.

The measurement noise $\mathbf{v}_{k+1}$ and the process noise $\mathbf{w}_k$ are independent. The matrix $\boldsymbol{P}_{k+1} \in \mathbb{R}^{n \times n}$ is the covariance of the state and $\boldsymbol{S}_{k+1} \in \mathbb{R}^{m \times m}$ is known as the innovation predictions matrix. These matrices propagate as follow:

$$\boldsymbol{P}_{k+1|k} = \boldsymbol{F}_k \cdot \boldsymbol{P}_{k|k} \cdot \boldsymbol{F}_k^T + \boldsymbol{Q}_k^T \tag{4.3}$$

$$\boldsymbol{S}_{k+1} = \boldsymbol{H}_{k+1} \cdot \boldsymbol{P}_{k+1|k} \cdot \boldsymbol{H}_{k+1}^T + \boldsymbol{R}_{k+1} \tag{4.4}$$

Since $\mathbf{v}_{k+1}$ and $\mathbf{w}_k$ are zero-mean noises, the state $\mathbf{x}_{k+1}$ and the measurements $\boldsymbol{z}_{k+1}$ vectors are predicted by the *prediction equations*:

$$\hat{\mathbf{x}}_{k+1|k} = \boldsymbol{F}_k \cdot \hat{\mathbf{x}}_{k|k} + \boldsymbol{G}_k \cdot \mathbf{u}_k \tag{4.5}$$

$$\hat{\boldsymbol{z}}_{k+1|k} = \boldsymbol{H}_k \cdot \hat{\mathbf{x}}_{k+1|k} \tag{4.6}$$

Then $\boldsymbol{\nu}_k = \boldsymbol{z}_{k+1|k} - \hat{\boldsymbol{z}}_{k+1|k}$ and it is known as the innovation of the process. The state vector $\hat{\mathbf{x}}_{k|k+1}$ is call the *a posteriori* estimation of the state, as it contains the information of the measurements till the time-step $k+1$.

$$\hat{\mathbf{x}}_{k|k+1} = \hat{\mathbf{x}}_{k+1|k} + \boldsymbol{K}_{k+1} \cdot \mathbf{v}_{k+1} \tag{4.7}$$

where $\boldsymbol{K}_{k+1}$ is the Kalman gain:

$$\boldsymbol{K}_{k+1} = \boldsymbol{P}_{k+1|k} \cdot \boldsymbol{H}_{k+1}^{T} \cdot \boldsymbol{S}_{k+1}^{-1} \tag{4.8}$$

The covariance of the system is updated at each time-step according to the equation:

$$\boldsymbol{P}_{k|k+1} = (\boldsymbol{I} - \boldsymbol{K}_{k+1} \cdot \boldsymbol{H}_{k+1}) \cdot \boldsymbol{P}_{k+1|k} \tag{4.9}$$

where $\boldsymbol{I}$ is the identity matrix.

Under the assumption that all the noises entering into the system are white noises, the Kalman filter minimises the square error of the estimates of the system states.

4.3 Extended Kalman Filter

The Extended Kalman Filter (EKF) is the nonlinear version of the Kalman Filter. The SLAM problem is usually formulated in the literature as a nonlinear system, expressed as the differentiable equations as (4.10) and (4.11), where EKF is a suitable method to obtain a solution [9]. Other nonlinear filtering solutions based on the Unscented Kalman Filter (UKF) and the Particle Filter (PF) are discarded here due to their computational complexities base on multiple space variation hypothesis. This makes them less suitable than EKF even with its linearisation step.

$$\mathbf{x}_{k+1} = \boldsymbol{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \tag{4.10}$$

$$\boldsymbol{z}_{k+1} = \boldsymbol{h}(\mathbf{x}_k, \mathbf{v}_{k+1}) \tag{4.11}$$

where $\mathbf{x}_k$ is the state vector of the system containing the position of the robot and the position of the mapped landmarks, and $\boldsymbol{z}_k$ are the visual measurements that will be used to estimate the state evolution through on the EKF.

The *process function*, also known as the *transition function*, $\boldsymbol{f}$ depends on the state and on the input vector $\mathbf{u}_k$ and represents the robot's kinematic behaviour. The variables $\mathbf{w}_k$ and $\mathbf{v}_k$ are random variables used to model the process and the measurement noises as in the previous section. The EKF as the Kalman Filter, assumes that the probability distributions of the noises are modelled as Gaussian functions.

Therefore, to find a solution for the system given in (4.10) and (4.11) using the Kalman Filter methodology it is necessary to linearise it first. A suitable possibility is using the first order Taylor series expansion around the current estimate at each time-step.

$$\boldsymbol{F}_{k+1} = \left(\frac{\partial \boldsymbol{f}}{\partial \mathbf{x}}\right)_{\hat{\mathbf{x}}_{k|k+1}} \quad \text{Jacobian of the transition model function} \tag{4.12}$$

$$\boldsymbol{H}_{k+1} = \left(\frac{\partial \boldsymbol{h}}{\partial \mathbf{x}}\right)_{\hat{\mathbf{x}}_{k+1|k}} \quad \text{Jacobian of the observation function} \tag{4.13}$$

The prediction equations for EKF are rewritten as follow:

$$\hat{\mathbf{x}}_{k+1|k} = \boldsymbol{f}(\mathbf{x}_{k|k}, \mathbf{u}_k) \tag{4.14}$$

$$\hat{\boldsymbol{z}}_{k+1} = \boldsymbol{h}(\mathbf{x}_{k+1|k}) \tag{4.15}$$

EKF is not an optimal nonlinear estimator, but can cope with nonlinear systems and with non-Gaussian distributed noises giving a reasonable performance. Badly modelled systems or wrong initial estimates of the state can lead to divergence issues. It is also important to state that the estimated covariance matrix for EKF tends to underestimate the real covariance matrix.

4.4 VSLAM

As a contribution to the thesis, we have developed a VSLAM solution based on an observation model later explained in this section, (4.32). Through tests conducted using a Matlab implementation we show a comparison between the performance of VSLAM versus the robot odometry data.

The visual extraction and matching of keypoints is based on the concepts explained earlier in this text, chapter 3, where we justify the use of Speeded Up Robust Features (SURF).

4.4.1 State vector

When a robot is placed in an unknown location, it should use its surrounding selecting landmarks to localise itself with respect to them. For a three wheeled mobile robot, Figure 4.1, the kinematic model is shown in (4.16) (4.17) (4.18), where subscript M refers to the mobile robot.

$$\dot{x}_M = V \cdot \cos\psi \tag{4.16}$$

$$\dot{y}_M = V \cdot \sin\psi \tag{4.17}$$

$$\dot{\psi}_M = \frac{V}{D} \cdot \sin\gamma \tag{4.18}$$

The transition matrix for the discrete system then is:

$$\begin{Bmatrix} x_{M,k+1} \\ y_{M,k+1} \\ \psi_{M,k+1} \end{Bmatrix} = \begin{Bmatrix} x_{M,k} + \Delta T \cdot V_k \cdot \cos\psi_{M,k} \\ y_{M,k} + \Delta T \cdot V_k \cdot \sin\psi_{M,k} \\ \psi_{M,k} + \Delta T \cdot \frac{V_k}{D} \cdot \sin\gamma_k \end{Bmatrix} + \mathbf{w}_k \tag{4.19}$$

where:

- $x_{M,k}$ and $y_{M,k}$ are cartesian coordinates of the robot
- $\psi_{M,k}$ is the bearing angle of the robot
- V_k is the velocity of the robot
- γ_k is the bearing angle of the front robot wheel
- D is the distance from the rear axis to steering wheel.

Since the transition model function contains trigonometric functions, the KF cannot be applied to solve the problem. Thus, the EKF is used instead. Here, the mapping problem is dealt with by selecting landmarks that will be assumed to be stationary. These landmarks are physical reference points that will be detected using on-board sensors.

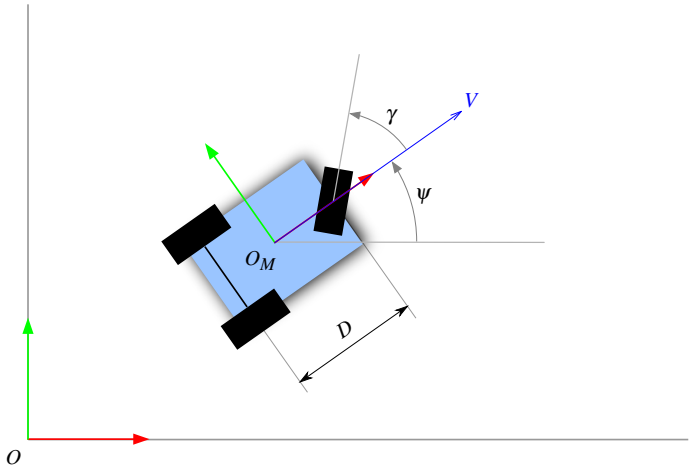


Figure 4.1: Three wheeled robot model

The equations of the landmarks in the state vector are written as follows:

$$\begin{aligned} \boldsymbol{l}_{k+1}^{(i)} &= \boldsymbol{l}_{k}^{(i)} = \boldsymbol{l}^{(i)} \\ x_{k+1}^{(i)} &= x_{k}^{(i)} = x^{(i)} \\ y_{k+1}^{(i)} &= y_{k}^{(i)} = y^{(i)} \\ z_{k+1}^{(i)} &= z_{k}^{(i)} = z^{(i)} \end{aligned} \tag{4.20}$$

where:

- $\boldsymbol{l}_{k}^{(i)} \in \mathbb{R}^3$ is the position vector of the i^{th} landmark at timestep k
- $x^{(i)}, y^{(i)}, z^{(i)} \in \mathbb{R}$ are cartesian coordinates of the landmark $\boldsymbol{l}^{(i)}$.

The ground robot used here is a two wheeled robot and is represented by a simple tricycle kinematics model, where the state vector is composed of two coordinates that define the 2D position on the plane and an additional state for the bearing angle of the robot. These states complete the definition of the robot pose. The transition model function $\boldsymbol{f}$ of the robot correspond to the one illustrated in (4.19).

For any time-step, the first three elements of the state vector of the VSLAM are the ones corresponding to the pose of the robot, while the rest of them contain the information concerning to the position of the p landmarks expressed in world reference frame coordinates (4.21).

Based on the state vector in (4.21) and on the definition of the Jacobian of the transition model function given in (4.12) the linearised version of the transition matrix is derived as in (4.22). The input matrix is analogously obtained (4.23).

$$\begin{Bmatrix} x_{M,k} \\ y_{M,k} \\ \psi_{M,k} \\ \boldsymbol{l}_k^{(1)} \\ \vdots \\ \boldsymbol{l}_k^{(p)} \end{Bmatrix} = \begin{Bmatrix} x\ position \\ y\ position \\ bearing\ angle \\ 3D\ position\ of\ 1^{st}\ landmark \\ \vdots \\ 3D\ position\ of\ last\ landmark \end{Bmatrix} \tag{4.21}$$

$$\boldsymbol{F}_k = \begin{bmatrix} \boldsymbol{F}_{M,k} & 0 \\ 0 & \boldsymbol{I}_{3p} \end{bmatrix} \longrightarrow \boldsymbol{F}_{M,k} = \begin{bmatrix} 1 & 0 & -\Delta T \cdot V_k \cdot \cos\psi_{M,k} \\ 0 & 1 & -\Delta T \cdot V_k \cdot \sin\psi_{M,k} \\ 0 & 0 & 1 \end{bmatrix} \tag{4.22}$$

$$\boldsymbol{G}_k = \begin{bmatrix} \boldsymbol{G}_{M,k} & 0 \\ 0 & \boldsymbol{0}_{3p} \end{bmatrix} \longrightarrow \boldsymbol{G}_{M,k} = \begin{bmatrix} \Delta T \cdot \cos\psi_{M,k} & -\Delta T \cdot V_k \cdot \sin\psi_{M,k} \\ \Delta T \cdot \sin\psi_{M,k} & \Delta T \cdot V_k \cdot \cos\psi_{M,k} \end{bmatrix} \tag{4.23}$$

The landmarks $\{L^{(i)}\}$ for $i = 1, 2, \ldots, p$ are considered stationary, hence invariant with respect to the world reference frame. The 3D position vector of the i^{th} landmark is expressed in (4.24) and (4.25) in world and mobile reference frames respectively.

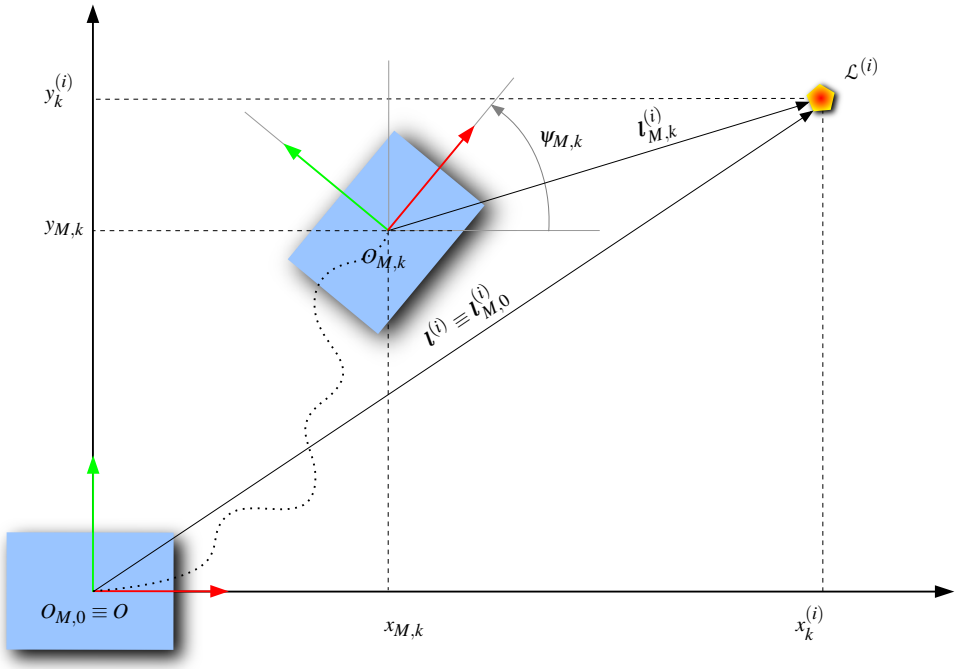


Figure 4.2: World and robot reference frames for VSLAM.

$$\boldsymbol{l}^{(i)} = \boldsymbol{l}_k^{(i)} = \begin{pmatrix} x_k^{(i)} & y_k^{(i)} & z_k^{(i)} \end{pmatrix}^T \tag{4.24}$$

$$\boldsymbol{l}_{M,k}^{(i)} = \begin{pmatrix} x_{M,k}^{(i)} & y_{M,k}^{(i)} & z_{M,k}^{(i)} \end{pmatrix}^T \tag{4.25}$$

Our implementation performs stereo matching and feature tracking using SURF local descriptors of the detected landmarks, as discussed in section 3.4. Thus, in order to make possible the landmark identification from one frame to another one the local descriptors of the features are stored.

Figure 4.2, represents the world and the robot reference frames, as the coordinates of an observed landmark $L^{(i)}$ from these reference frames. The stationary position vector of the landmark with respect to the world reference frame appears as $\boldsymbol{l}^{(i)}$, whereas the relative position with respect to the robot appears as $\boldsymbol{l}_{M,k}^{(i)}$ for two robot positions $k = 1, 2$.

4.4.2 Observation model

The EKF, an more generally the Kalman filter, take into consideration the measurements acquired from the robot through the system *observations*. These are later used to conduct the system update.

The observation model, presented in (4.11), and repeated here (4.26) for the reader's convenience, represents the measurements that are shaped differently depending on the type of sensor employed.

$$\boldsymbol{z}_{k+1} = \boldsymbol{h}(\mathbf{x}_k, \mathbf{v}_{k+1}) \tag{4.26}$$

where $\boldsymbol{z}_{k+1} \in \mathbb{R}^m$ is the measurement vector $\mathbf{x}_k$ is the state vector, and $\mathbf{v}_{k+1} \in \mathbb{R}^m$ is a random variable that represents the error due to the measurement process.

The inverse observation model, (4.27), is used to initialise the landmarks added into the state vector. Given that features are considered stationary with a fixed position in space, their position has to be estimated from the measurements only once.

$$\boldsymbol{l}_{k^{(i)}} = \boldsymbol{h}^{-1}(\boldsymbol{z}_k, \mathbf{v}_k) \tag{4.27}$$

This subsection shows three observation models normally used for the most common perception alternatives used on VSLAM. Firstly, an approach written in polar coordinates specially useful when range and bearing sensors are used is presented. Secondly, an observation model in radial coordinates and suitable for 3D perceptive sensors is shown. Finally, two common approaches for the observation model used on imaging sensors are introduced.

4.4.2.1 Polar measurements: range and bearing

For a range and bearing laser equipped robot, the measurements can be written in polar coordinates as the following observation model:

$$z_k^{(i)} = \begin{Bmatrix} \rho_k^{(i)} \\ \theta_k^{(i)} \end{Bmatrix} + \mathbf{v}_k = \begin{Bmatrix} \sqrt{\left(x_M - x^{(i)}\right)^2 + \left(y_M - y^{(i)}\right)^2} + \mathrm{v}_{\rho,k} \\ \arctan\left(\dfrac{y^{(i)} - y_{M,k}}{x^{(i)} - x_{M,k}}\right) - \psi_{M,k} + \mathrm{v}_{\theta,k} \end{Bmatrix} \tag{4.28}$$

where:

- $\theta_k^{(i)}$ is the angle between the robot and the i^{th} landmark i
- $\rho_k^{(i)}$ is the radius from the robot to the landmark in polar coordinates.

4.4.2.2 Radial measurements

Radial coordinates are more convenient in certain circumstances, for instance when 3D LASER scanners are used. Some solutions represent the landmark measurements in radial coordinates, in terms of the distance $r^{(i)}$, azimuth $\alpha^{(i)}$ and elevation $\beta^{(i)}$, instead of representing them in polar coordinates or as image frame pixel locations [88].

$$z^{(i)} = \begin{Bmatrix} r^{(i)} \\ \alpha^{(i)} \\ \beta^{(i)} \end{Bmatrix} + \mathbf{v}_k = \begin{Bmatrix} \sqrt{\left(x_M - x^{(i)}\right)^2 + \left(y_M - y^{(i)}\right)^2 + \left(z_M - z^{(i)}\right)^2} + \mathrm{v}_{r,k} \\ \arctan\left(\dfrac{y_M - y^{(i)}}{x_M - x^{(i)}}\right) + \mathrm{v}_{\alpha,k} \\ -\arctan\left(\dfrac{\left(z_M - z^{(i)}\right)}{\sqrt{\left(x_M - x^{(i)}\right)^2 + \left(y_M - y^{(i)}\right)^2}}\right) + \mathrm{v}_{\beta,k} \end{Bmatrix} \tag{4.29}$$

4.4.2.3 Stereo image obsevation

For a robot equipped with on-board cameras, the observation model relates the 2D position of the features in the images to their 3D equivalent landmarks. In this manner, features expressed in terms of (u_L, v_L) and (u_R, v_R) will compose the observations of the stereo camera system. This is mathematicaly written in (4.30) and (4.31).

$$\boldsymbol{z}_k = \begin{pmatrix} \boldsymbol{z}_k^{(1)^T} & \boldsymbol{z}_k^{(2)^T} & \dots & \boldsymbol{z}_k^{(p)^T} \end{pmatrix}^T \tag{4.30}$$

$$\boldsymbol{z}_k^{(i)} = \begin{pmatrix} u_{L,k}^{(i)} & v_{L,k}^{(i)} & u_{R,k}^{(i)} & v_{R,k}^{(i)} \end{pmatrix}^T \tag{4.31}$$

where:

- $\boldsymbol{z}_k \in \mathbb{R}^{4p}$ is the vector of measurements at time-step k
- $\boldsymbol{z}_k^{(i)} \in \mathbb{R}^4$ is the measurement of the i^{th} visual landmark.

For systems based on stereo vision setups consisting of two on-board cameras, mathematical solutions to the VSLAM problem can be found as described in [11], where the observation model is derived as follows and is the approach used here.

$$\begin{Bmatrix} u_L^{(i)} \\ v_L^{(i)} \\ u_R^{(i)} \\ v_R^{(i)} \end{Bmatrix} = \begin{Bmatrix} \dfrac{m_{11,L} \cdot x^{(i)} + m_{12,L} \cdot y^{(i)} + m_{13,L} \cdot z^{(i)} + m_{14,L}}{m_{31,L} \cdot x^{(i)} + m_{32,L} \cdot y^{(i)} + m_{33,L} \cdot z^{(i)} + m_{34,L}} \\ \dfrac{m_{21,L} \cdot x^{(i)} + m_{22,L} \cdot y^{(i)} + m_{23,L} \cdot z^{(i)} + m_{24,L}}{m_{31,L} \cdot x^{(i)} + m_{32,L} \cdot y^{(i)} + m_{33,L} \cdot z^{(i)} + m_{34,L}} \\ \dfrac{m_{11,R} \cdot x^{(i)} + m_{12,R} \cdot y^{(i)} + m_{13,R} \cdot z^{(i)} + m_{14,R}}{m_{31,R} \cdot x^{(i)} + m_{32,R} \cdot y^{(i)} + m_{33,R} \cdot z^{(i)} + m_{34,R}} \\ \dfrac{m_{21,R} \cdot x^{(i)} + m_{22,R} \cdot y^{(i)} + m_{23,R} \cdot z^{(i)} + m_{24,R}}{m_{31,R} \cdot x^{(i)} + m_{32,R} \cdot y^{(i)} + m_{33,R} \cdot z^{(i)} + m_{34,R}} \end{Bmatrix} \tag{4.32}$$

where:

- u, v are the pixel coordinates of a feature
- $m_{ij,C}$ is the component of the projection matrix of the camera C, for $C = \{L, R\}$ at i^{th} row and j^{th} column. Cameras left and right are labeled using L and R
- $\left(x^{(i)}, y^{(i)}, z^{(i)}\right)^T$ is the position vector of the i^{th} landmark.

Another observation model, similar to the one above, is the stereo alike visual concept based on a pan-tilt monocular camera proposed in [81]. The way this works is using a couple of images acquired in consecutive time-steps to emulate a stereo capture. The equation for the observation is written as:

$$\begin{Bmatrix} u_k^{(i)} \\ v_k^{(i)} \\ u_{k+1}^{(i)} \\ v_{k+1}^{(i)} \end{Bmatrix} = \begin{Bmatrix} \alpha_u \cdot \left(\dfrac{m_{11,k} \cdot x_k^{(i)} + m_{12,k} \cdot y_k^{(i)} + m_{13,k} \cdot z_k^{(i)} + m_{14,k}}{m_{31,k} \cdot x_k^{(i)} + m_{32,k} \cdot y_k^{(i)} + m_{33,k} \cdot z_k^{(i)} + m_{34,k}} \right) + u_0 \\ \alpha_v \cdot \left(\dfrac{m_{21,k} \cdot x_k^{(i)} + m_{22,k} \cdot y_k^{(i)} + m_{23,k} \cdot z_k^{(i)} + m_{24,k}}{m_{31,k} \cdot x_k^{(i)} + m_{32,k} \cdot y_k^{(i)} + m_{33,k} \cdot z_k^{(i)} + m_{34,k}} \right) + v_0 \\ \alpha_u \cdot \left(\dfrac{m_{11,k+1} \cdot x_{k+1}^{(i)} + m_{12,k+1} \cdot y_{k+1}^{(i)} + m_{13,k+1} \cdot z_{k+1}^{(i)} + m_{14,k+1}}{m_{31,k+1} \cdot x_{k+1}^{(i)} + m_{32,k+1} \cdot y_{k+1}^{(i)} + m_{33,k+1} \cdot z_{k+1}^{(i)} + m_{34,k+1}} \right) + u_0 \\ \alpha_v \cdot \left(\dfrac{m_{21,k+1} \cdot x_{k+1}^{(i)} + m_{22,k+1} \cdot y_{k+1}^{(i)} + m_{23,k+1} \cdot z_{k+1}^{(i)} + m_{24,k+1}}{m_{31,k+1} \cdot x_{k+1}^{(i)} + m_{32,k+1} \cdot y_{k+1}^{(i)} + m_{33,k+1} \cdot z_{k+1}^{(i)} + m_{34,k+1}} \right) + v_0 \end{Bmatrix} \tag{4.33}$$

where:

- u_k, v_k are the pixel coordinates, of a feature at time k
- $m_{ij,k}$ is the component of the projection matrix of the camera at i^{th} row and j^{th} column at time k .
- $\left(x_k^{(i)}, y_k^{(i)}, z_k^{(i)}\right)^T$ is the position vector of the i^{th} landmark.

Note that as this is a stereo-like example based on a unique camera, we can omit the subscript k that indicates time-step, under the assumption that the parameters are time independent so that:

$$\alpha_u = \alpha_{u,k} = \alpha_{u,k+1} \tag{4.34}$$

$$\alpha_v = \alpha_{v,k} = \alpha_{v,k+1} \tag{4.35}$$

$$u_0 = u_{0,k} = u_{0,k+1} \tag{4.36}$$

$$v_0 = v_{0,k} = v_{0,k+1} \tag{4.37}$$

Figure 4.3 shows how landmark positions are reconstructed from pairs of stereo matches, in order to add a new landmark position into the state vector of the system.

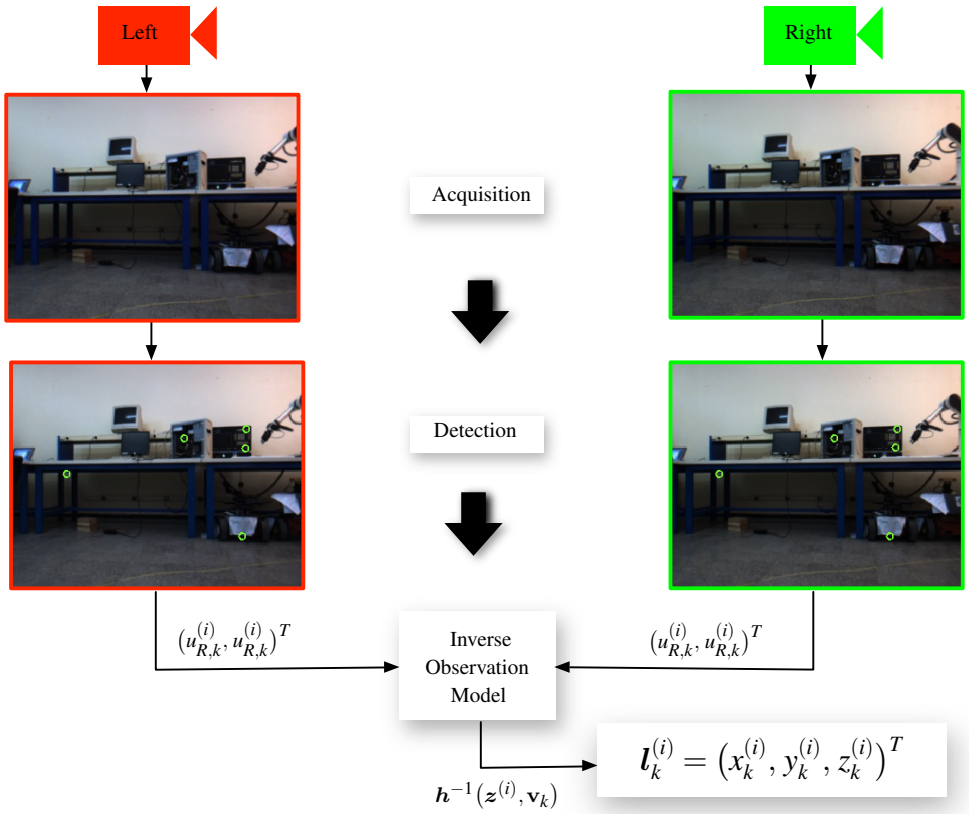


Figure 4.3: Inverse Model and new features addition

4.4.3 Association process

In order for an EKF-VSLAM solution to update the system, it is necessary to identify which landmark does correspond to each measurement. This step is part of the observation process and, although it may look trivial, it is a challenging stage to be tackled.

Between the suitable alternatives available to solve this problem three of them can be highlighted: registration , tracking and association techniques. The first is the simplest. What registration algorithms do is essentially fitting sets of points. Hence, the way of applying a registration technique to VSLAM consists on fitting sets of 3D points so that the best closest point once the sets have fitted is considered to be the corresponding landmark. Iterative Closes Point must be mentioned here as a representative technique for this family of solutions. The second alternative consists of employing a visual tracker as KLT or any alternative optical flow solution. In this manner, the features are identified along time, so that the tracks correspond to the same landmark. Finally, we want to emphasise on the possibility selected here, that consists on a landmark association process based on local descriptors matching, section 3.4.

One of the benefits of the selected association strategy used here is that the same visual descriptors that are used for stereo matching, are also used to identify the landmarks along time. For every detected feature $\mathcal{F}_{C,k}^{(i)}$ a descriptor $\boldsymbol{\delta}_{C,k}^{(i)}$ is computed and stored for left and right cameras, $C = \{L, R\}$. We user SURF local descriptors, because of the advantages explained in chapter 3,

The association process is responsible for identifying which of the detected landmarks are new landmarks to be initialise and which are known landmarks already seen in prior time-steps.

At every time-step, the set of stored descriptors corresponding to previously associated landmarks is matched against the descriptors of the currently detected features. If a descriptor does not correspond to anyone of the existing landmarks it corresponds to a new landmark hence the inverse observation model is used to add it into the state vector Figure 4.3. On the contrary, when a match is found for the descriptor the state vector is updated using the visual observation, as explained in subsection 4.4.4. Only landmarks that have been already stereo matched are considered at this stage.

4.4.4 Updates

Associated landmarks are used to update the VSLAM state vector. Within the prediction step of the EKF, the positions in the current time-step of previously associated landmarks are computed. Using the *a priori* state vector through the observation model (4.26) the difference between the predicted positions and the positions of the associated features in the image frames yield to the *innovation vector* $\boldsymbol{\nu}_k$, .

$$\hat{\boldsymbol{z}}_{k+1} = \boldsymbol{h}(\mathbf{x}_{k+1|k}) \tag{4.38}$$

$$\boldsymbol{\nu}_k = \boldsymbol{z}_{k+1|k} - \hat{\boldsymbol{z}}_{k+1|k} \tag{4.39}$$

Then, the Kalman gains are computed for each associated landmark in the set $\{\mathcal{L}^{(i)}\}$, for $i = 1, 2, \ldots, p$, as in (4.41), using the Jacobian matrix of the observation model with respect to the landmark, $\boldsymbol{H}_{\boldsymbol{l}^{(i)}}$, (4.40). The Kalman gain $\boldsymbol{K}_{\boldsymbol{l}^{(i)}}$, is multiplied by the innovation vector, (4.42), to update the state vector.

$$\boldsymbol{H}_{\boldsymbol{l}^{(i)}} = \left(\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{l}^{(i)}} \right)_{\hat{\mathbf{x}}_{k+1|k}} \tag{4.40}$$

$$\boldsymbol{K}_{\boldsymbol{l}^{(i)}} = \boldsymbol{P}_{k+1|k} \cdot \boldsymbol{H}_{\boldsymbol{l}^{(i)}}^{T} \cdot \left(\boldsymbol{H}_{\boldsymbol{l}^{(i)}} \cdot \boldsymbol{P}_{k+1|k} \cdot \boldsymbol{H}_{\boldsymbol{l}^{(i)}}^{T} + \boldsymbol{R} \right)^{-1} \tag{4.41}$$

Note that the Jacobian of the observation model with respect to the landmark position is not time invariant, but it has not been noted with the subindex $(k+1|k)$ just to reduce the notation and make (4.41) clear to the reader.

$$\hat{\mathbf{x}}_{k+1|\boldsymbol{l}^{(j)}} = \hat{\mathbf{x}}_{k+1|\boldsymbol{l}^{(j-1)}} + \boldsymbol{K}_{\boldsymbol{l}^{(j)}} \cdot \left(\boldsymbol{z}_{k+1|k} - \hat{\boldsymbol{z}}_{k+1|k} \right) \tag{4.42}$$

where the state vector $\hat{\mathbf{x}}_{k+1|\boldsymbol{l}^{(j)}}$ is the *a posteriori* estimation obtained after using the first j landmark observations.

For the first landmark the equation takes the form shown in (4.43).

$$\hat{\mathbf{x}}_{k+1|\boldsymbol{l}^{(1)}} = \hat{\mathbf{x}}_{k+1|k} + \boldsymbol{K}_{\boldsymbol{l}^{(1)}} \cdot \left(\boldsymbol{z}_{k+1|k} - \hat{\boldsymbol{z}}_{k+1|k} \right) \tag{4.43}$$

where $\hat{\mathbf{x}}_{k+1|k}$ is the *a priori* estimate of the state vector, obtained from the evolution of the system through the transition model, (4.44).

$$\hat{\mathbf{x}}_{k+1|k} = \boldsymbol{f}(\mathbf{x}_{k|k}, \mathbf{w}_k) \tag{4.44}$$

The covariance of the state is updated as shown in (4.45), (4.46) and (4.47).

$$\boldsymbol{P}_{k+1|\boldsymbol{l}^{(i)}} = \left(I - \boldsymbol{K}_{\boldsymbol{l}^{(i)}} \cdot \boldsymbol{H}_{\boldsymbol{l}^{(i)}}\right) \cdot \boldsymbol{P}_{k+1|m_{(i-1)}} \tag{4.45}$$

$$\boldsymbol{P}_{k+1|\boldsymbol{l}^{(1)}} = \boldsymbol{P}_{k+1|k} \tag{4.46}$$

$$\boldsymbol{P}_{k|k+1} = \boldsymbol{P}_{k+1|\boldsymbol{l}^{(p)}} \tag{4.47}$$

The flowchart in Figure 4.4 summarises how the state vector is sequentially updated using the information obtained from the visual measurements. The flowchart in Figure 4.5 schematises the whole VSLAM process for the developed implementation.

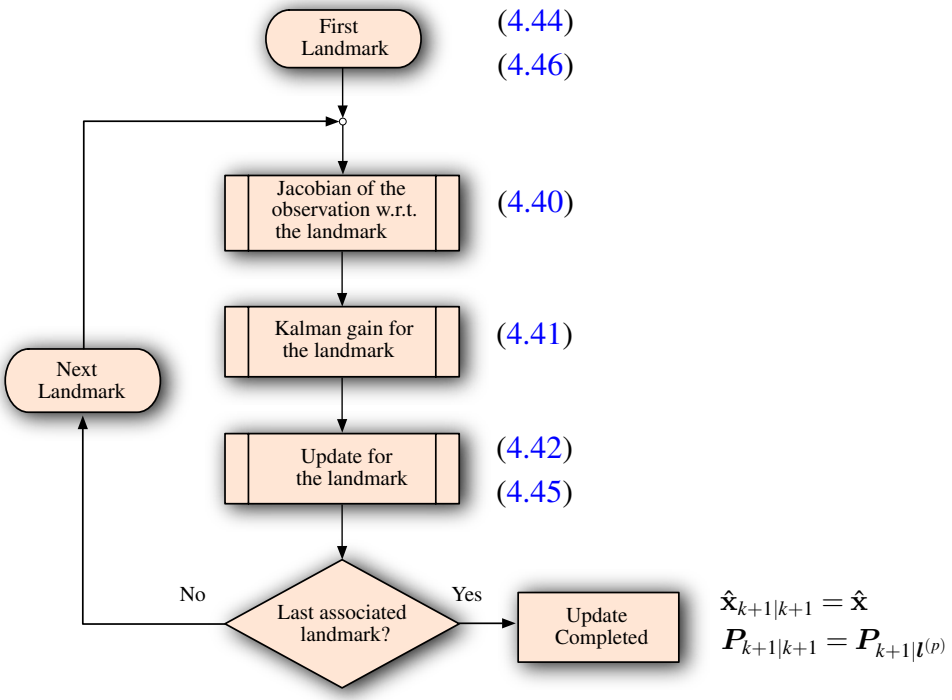


Figure 4.4: VSLAM update process

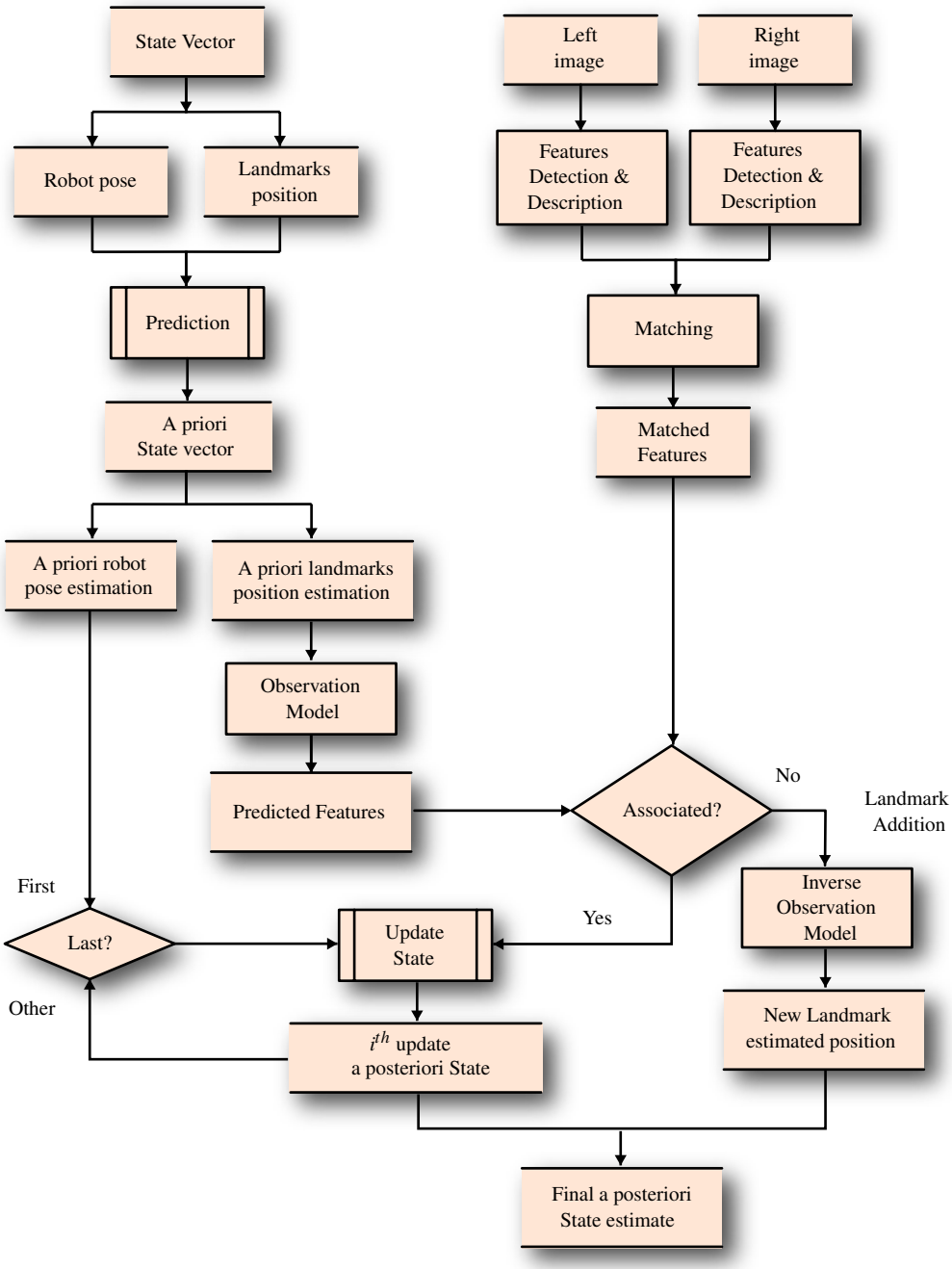


Figure 4.5: Flowchart of the implemented VSLAM

4.4.5 Map management

One of the important aspects of VSLAM is the *map management*. The observed landmarks that a mobile robot perceives through the on-board sensors correspond to physical references on the environment. In this manner the landmarks compose the map that the robot uses to self-locate, checking what is its relative position with respect those landmarks in the map.

Hence, map management is the VSLAM process responsible for either adding or removing landmarks from the map. Thus, this process involves modifying both the state vector, where the position of the landmarks is stored, and the covariance of the system, that contains the information relative to the uncertainty of the landmarks location.

The landmark *addition process* is conducted every time a new landmark is observed. Because of the matching and association strategy selected in our VSLAM is based on local SURF descriptors (section 4.4.3) features that have a stereo pair and are not associated with any landmark contained in the state vector will be considered as new landmarks and added into the system. Notwithstanding, landmark addition can lead to an uncontrolled growth of the state vector and the covariance of the system when no further actions are considered.

For the case study here, only visual features are considered as system observations. Therefore, it has to be pointed that not every visual feature has the same usability. This is due to different factors. Two sources of landmarks usability reduction are as follows:

- Features detection and description limitations
- Perspective conditions.

Firstly, because of repeatability of the features through which landmarks are observed, it is noted that not all the features observed at certain time-step will necessarily be detected on later images. This problem is intimately related to the detection technique used, although every detection techniques will present its own flaws under unfavourable circumstances that will lead to feature detection instabilities. Furthermore, even in cases of features that reappear along the sequence it happens that the description process does not guarantee successful stereo matching or association.

Secondly, it happens that landmarks observed at certain time-step might end up lying out of the camera's field of view (FOV) as the robot moves. But this is not the only consequence of perspective changes between the camera and the landmarks. The second issue due to perspective changes is the *occlusion* problem that occurs when mapped landmarks are hidden from the camera due to intermediate objects. Whereas the consequences derived from the second of the effects cannot be tackled, as they just depend on the physical layout the robot's environment, some improvements alleviate the former. For instance, reducing the detection threshold is a straightforward possibility that can be employed to reduce any excess of detected features. Nevertheless, it would not only affect the system growth, by dismissing features with low repeatability, it would also induce a reduction on the number of detected and associated landmarks.

For the reasons explained above, we introduce the landmark *rate of usability* $\eta^{(i)}$ as the quotient of the number of times that the landmark has been observed with respect to the number of times that the landmark could have been observed.

$$\eta^{(i)} = \frac{n^{(i)}}{t^{(i)}} \tag{4.48}$$

where:

- $\eta^{(i)}$ rate of usability of a landmark $\mathcal{L}^{(i)}$
- $n^{(i)}$ number of appearances of the landmark $\mathcal{L}^{(i)}$
- $t^{(i)}$ time-steps since the landmark's first appearance.

Using the proposed rate, we can avoid the uncontrolled growth of the system due to the limitations on visual processes robustness, by classifying the landmarks as usable and non-usable. In this way, those features whose rate of usability is under a minimum value are classified as non-usable and made inactive. We have empirically set the minimum of usability rate to 0.2.

To illustrate how the rate of usability is used let us consider two landmarks: $\mathcal{L}^A$ and $\mathcal{L}^B$ observed for the first time at time-step k. Now let us consider that at time-step $k+5$ the landmark $\mathcal{L}^A$ has been observed five times, whereas the landmark $\mathcal{L}^B$ has only been observed once. Then $n^A = 5$ and $n^B = 1$ while $t^A = t^B = 6$ according to their definitions. In this case the rates of usability will take values $\eta^A = 0.83$ and $\eta^B = 0.17$, thus $\mathcal{L}^A$ will remain active feature whereas $\mathcal{L}^B$ will become inactive.

The elements of the state vector corresponding to the position of non-usable landmarks are removed. Likewise, rows and columns corresponding to non-usable landmarks are removed from the covariance matrix . But one may think of a possible issue derived from this solution: it could lead to an excessive shrinking of the system. In order to prevent this and guarantee that a minimum of landmarks are kept, the proposed removal scheme is disabled when there are less that 20 or 30 landmarks.

4.4.6 Observation model enhancements

In this subsection we present two post-processing procedures oriented to improve VSLAM solutions. These enhancements are specially focused to improve:

- Association outliers removal
- Computational expenses reduction.

Although ambiguous matches are detected and dismissed, both at stereo matching and at association stage using the matching procedure explained in section 3.4, additional post-processing is required to improve the system's performance. In order to handle this common situation, we propose a statistical outlier rejection scheme based on the histograms of length and orientation of the matching segments. Considering that there are no mobile objects in the scene in front of the robot and based on the fact that the projective visual change will be small enough between two consecutive time-steps, it seems reasonable to think that the matching segments that join associated landmarks will be similar in terms of length and orientation. The similarity between association segments is checked building histograms of length and orientation, so that, only landmarks whose segments length and orientation are among the p closes elements with respect to the corresponding mode are kept as inliers.

The second issue we deal with is the influence on the time consumption of the number of managed features. This problem is partially tackled by the enhancement explained above and by the map management strategy explained in the previous subsection. However, further actions are proposed to alleviate the computational burden. The first measure to alleviate the time consumption of the VSLAM strategy consists of approximating the computation of the Euclidean distance calculated to check the distance between pair of descriptors, as explained in subsection 3.4, to the arccosine

of the dot product of the descriptors. This approach is valid for unit vectors as the distance is approximated as the angle between the vectors.

$$\text{dist}(\boldsymbol{\delta}^{(i)}, \boldsymbol{\delta}^{(j)}) = \sqrt{\sum_k \left(\delta_k^{(i)} - \delta_k^{(j)}\right)^2} \approx \arccos\left(\langle \boldsymbol{\delta}^{(i)}, \boldsymbol{\delta}^{(j)} \rangle\right) \tag{4.49}$$

where $\langle \boldsymbol{\delta}^{(i)}, \boldsymbol{\delta}^{(j)} \rangle$ is the dot product between the descriptors $\boldsymbol{\delta}^{(i)}$ and $\boldsymbol{\delta}^{(j)}$.

As an alternative to the statistical selection solution proposed above, a second measure has been studied to reduce the dimensionality of the set of associated landmarks. For this, a randomised selection of elements extracted from the set associated landmarks is considered to update the VSLAM system.

4.5 Experimental results

This section shows and analyses results for pose estimation obtained using our SURF-VSLAM implementation with and without the enhancements proposed in the previous section. Three strategies are considered here:

- **Strategy A:** Original SURF-VSLAM algorithm
- **Strategy B:** Outlier rejector based on matching lengths
- **Strategy C:** Reduced number of associated landmarks.

For all of these strategies, the map management enhancement that reduces the number of landmarks based on the rate of usability has been considered.

Figure 4.6 shows the association results obtained when no other enhancement than the rate of usability map management scheme is used. These sample pictures extracted from a sequence of 100 images, show the predicted (red) and detected (green)

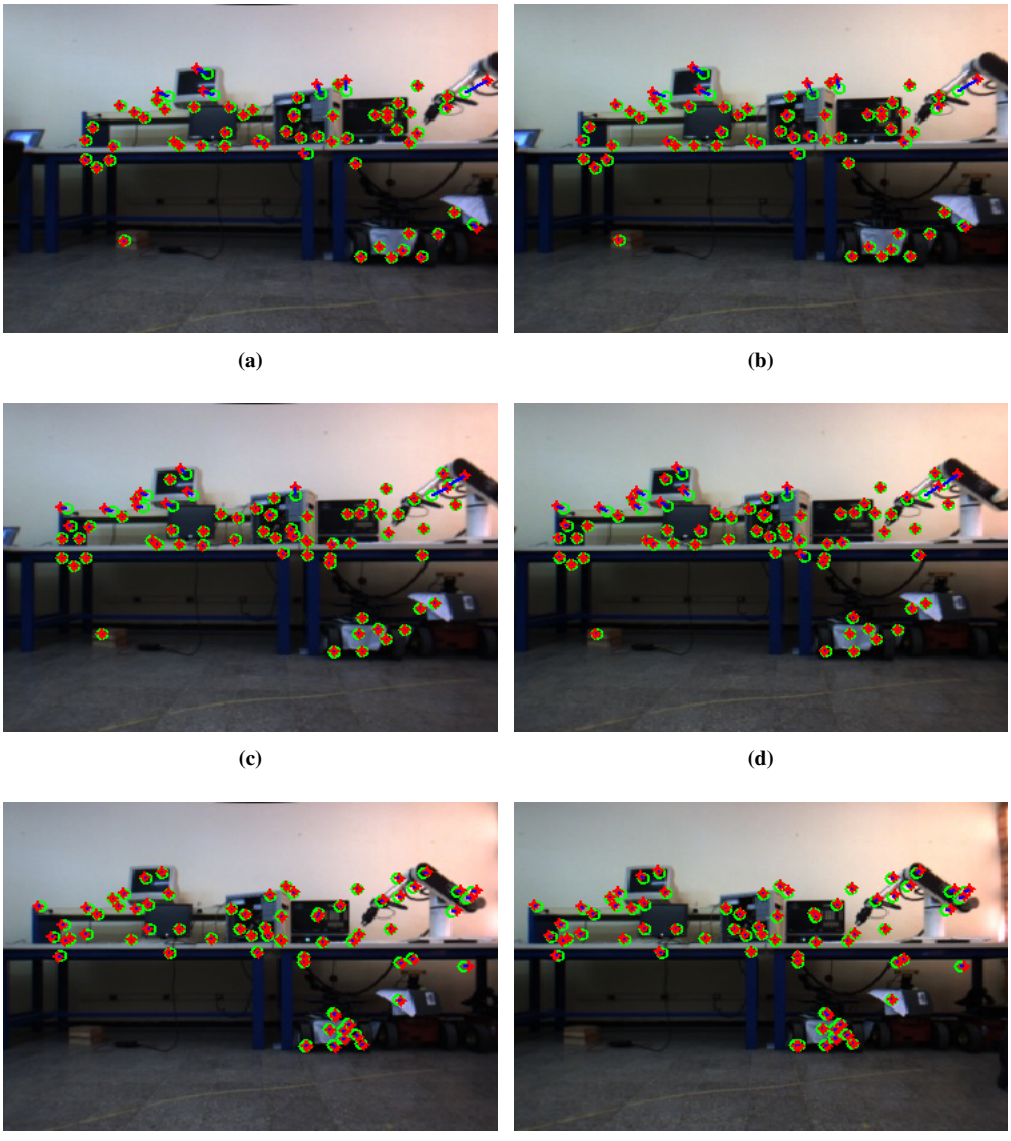
pixel positions corresponding to the features used to extract the associated landmarks at three subsequent time-steps. The blue segments are the *association segments* that represent associated pairs. In this figure, it is easy to reliase that there are a number of segments corresponding to association outliers. This is the case of the association segments where the association segment is clearly visible.

The fact is that most of the association segments are hidden by the features. This is because when there are no convergence issues the predicted position and the observed position of the landmark lie sufficiently close and the represented features are superimposed on each other.

Figure 4.7 shows the association results obtained when our statistical outlier rejector based on the length and orientation of the association segments is used. It is noted from the analysis of the figure how a number of associated landmarks are reduced. Moreover, it is seen how the long blue segments, corresponding to association outliers, do no appear here which show the suitability of the technique.

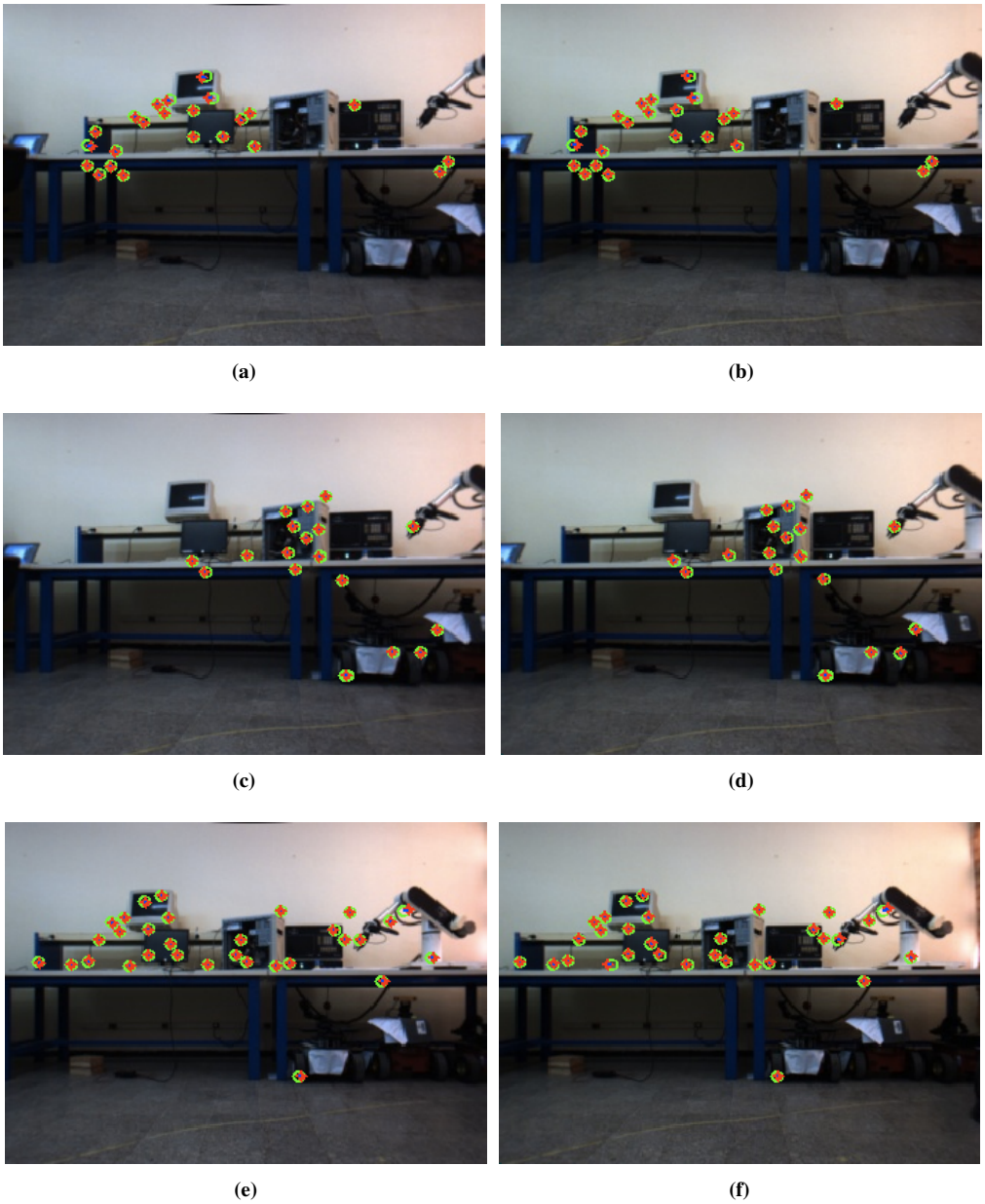
Figure 4.8 shows the landmark association results for the third strategy, where only a reduced randomised subset of associations is considered for system updates. Here, the reader can see how the number of associated landmarks decreases considerably with respect to the two previous cases.

Figure 4.9, shows the robot pose estimations obtained for the original SURF-VSLAM implementation and the two other enhanced strategies analysed here. Results for odometry estimation (red) and SURF-VSLAM (cyan) are superimposed together with the ground truth data (blue). Figure 4.10 shows the top-bottom view of the trajectory estimation (first row) and the location and orientation absolute errors from the different strategies (second and third rows). Odometry results (red) are displayed together with the proposed strategies (cyan/black) and the ground truth (blue).



(a) (b) (c) (d) (e) (f)

Figure 4.6: Association results for Strategy A.

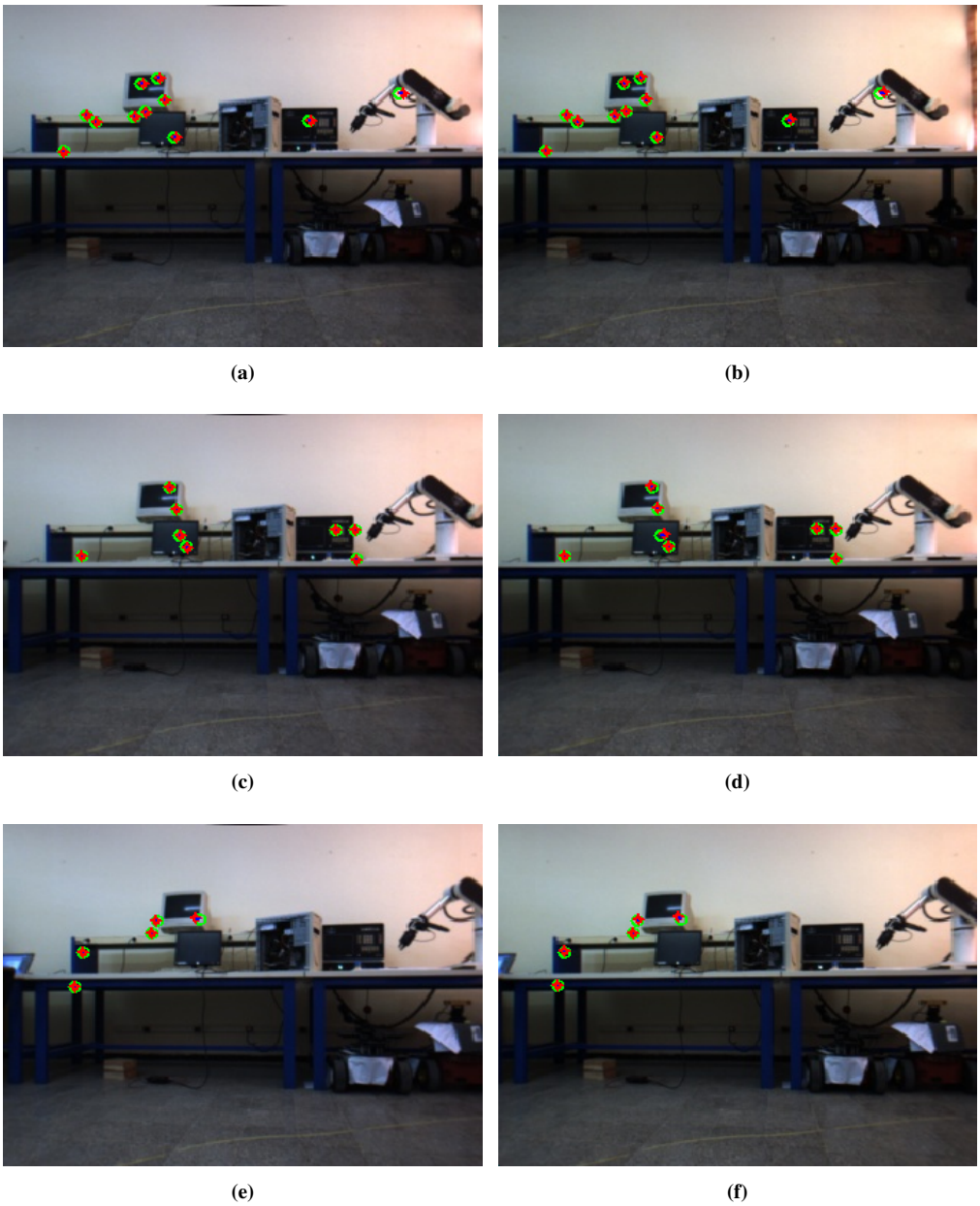


(a) (b)

(c) (d)

(e) (f)

Figure 4.7: Association results for Strategy B.



(a) (b) (c) (d) (e) (f)

Figure 4.8: Association results for Strategy C.

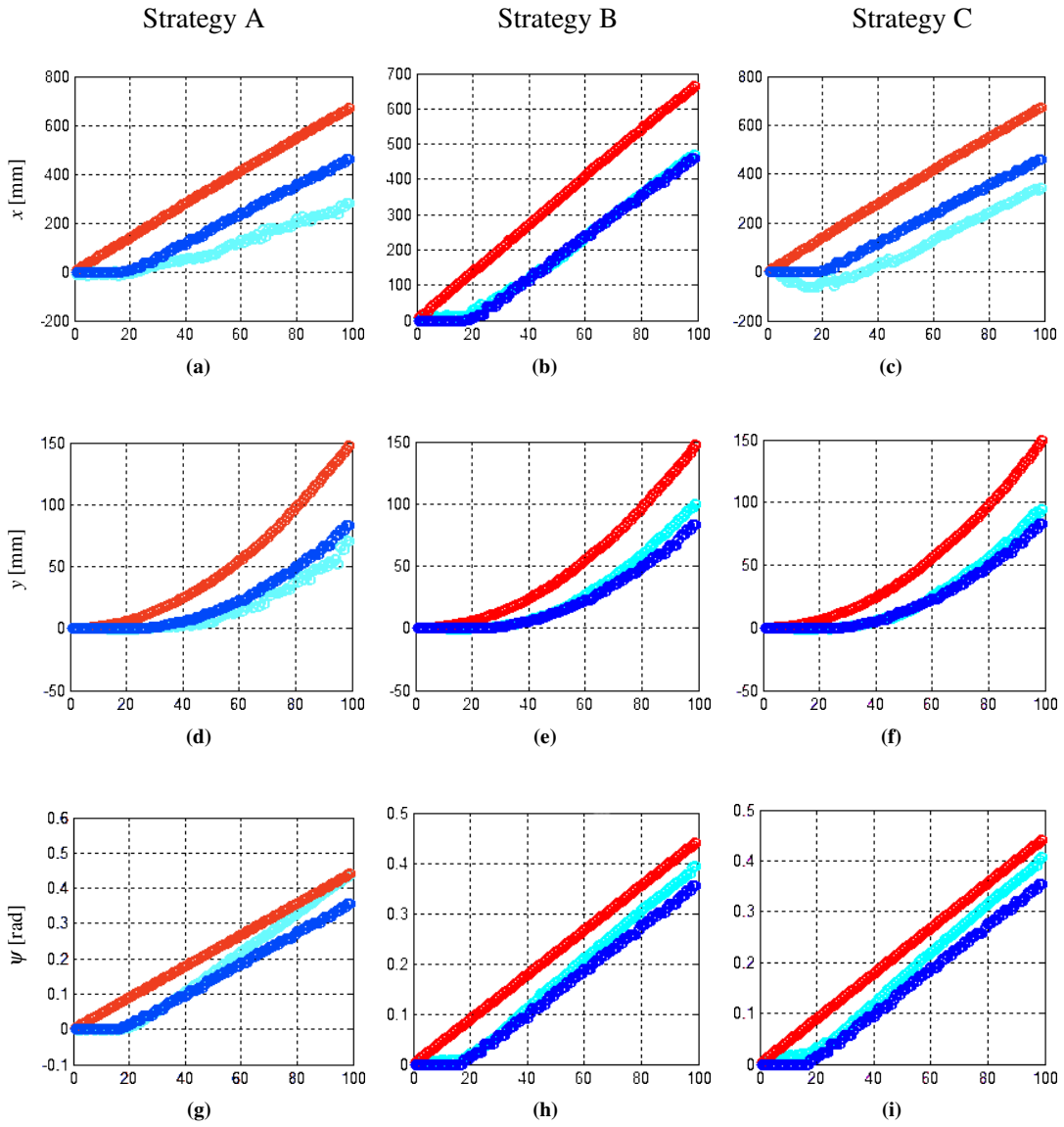


Figure 4.9: SURF-VSLAM estimated coordinates of the robot (cyan), reference data (blue) and odometry results (red). Original solution results are shown in (a), (b) and (c), results for the solution enhanced with the outlier rejection strategy are shown in (d), (e) and (f), results for the solution where the number of associated landmarks is reduced are shown in (g), (h) and (i).

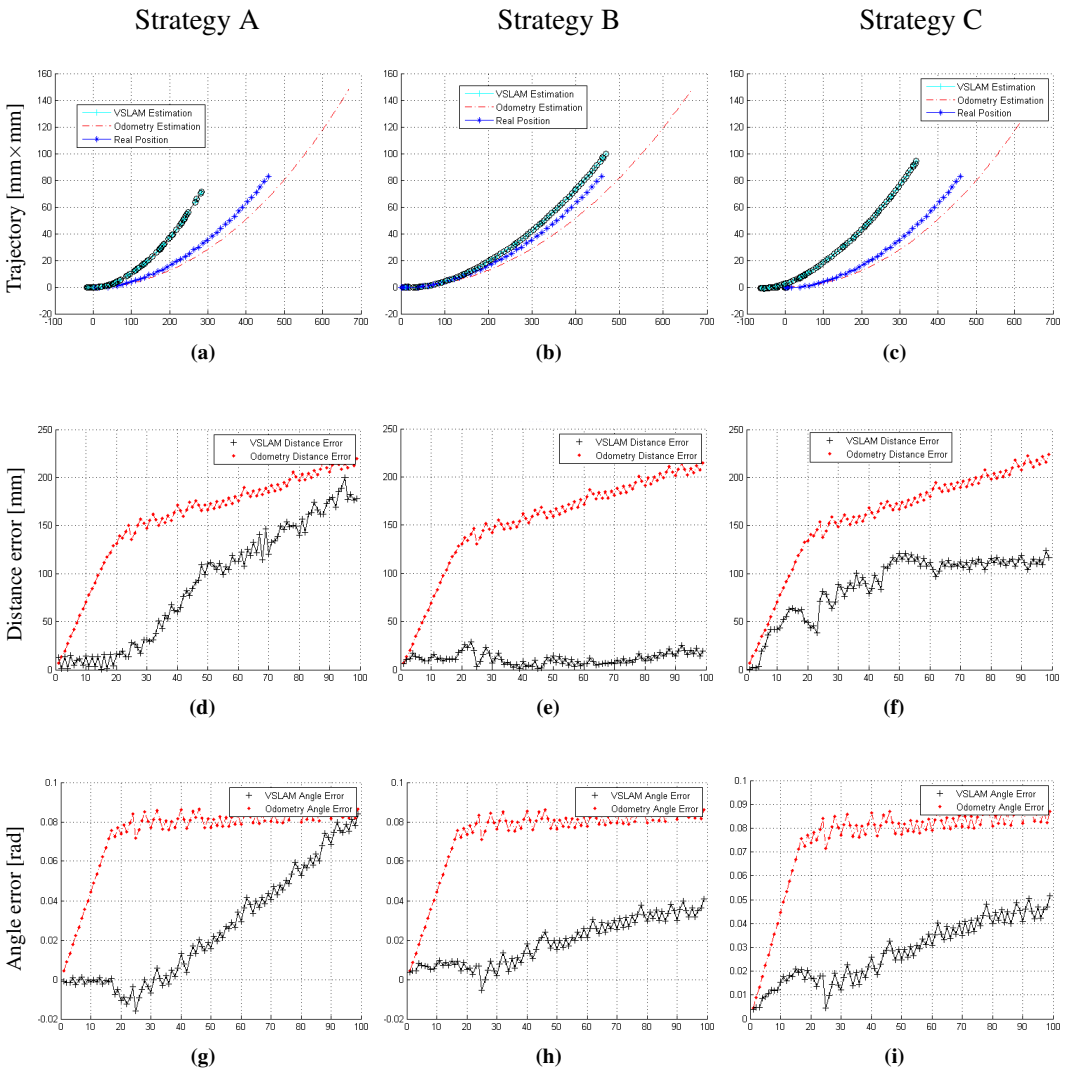


Figure 4.10: Estimation results of enhanced SURF-VSLAM strategies. Strategy A (Original implementation), Strategy B (False matches rejection) and Strategy C (Limited number of associations per time-step).

From the analysis of the figures, it is seen that the estimation using Strategy B entails an appreciable error decrease. The trajectory obtained using Strategy B lies closer to the ground truth compared to the other two strategies.

4.6 Conclusions

In this section an EKF-VSLAM solution based on SURF local descriptors for stereo matching and landmark association has been derived. System limitations have been identified and overcome by enhancing the VSLAM system using three specific strategies.

Firstly, an uncontrolled growth of the system due to the limitations of current feature detection techniques has been solved enhancing the map management procedure. This classifies the landmarks in two groups: usable and non-usable. The later are considered unreliable or unstable landmarks, hence removed from the EKF structures.

Secondly, further enhancements have been proposed to reduce the computational expenses and improve the performance of the VSLAM system. In order to tackle the first of these two, a randomised reduction on the number of observations is considered. For the later an statistical outlier rejection scheme based on the association segments joining tracked landmarks has been presented.

The proposed solutions have shown to reduce the system limitations and improve the results based on experimental results obtained from applying the techniques on a real sequence acquired from a mobile robot. We have seen that a strategy based on our statistical outlier rejector comes out on top.

Chapter 5

HMSURF as a VSLAM improvement

5.1 Overview

A VSLAM technique was presented in the previous Chapter as a suitable solution for the autonomous mobile robots navigation problem. The presented solution accomplishes the visual processing tasks by means of SURF local descriptors.

This Chapter presents an innovative VSLAM solution based on our Harris Moments Speed Up Robust Feature (HMSURF) visual scheme. Furthermore another visual module alternative Harris Speeded Up Robust Features (HSURF) is also analysed. HSURF and HMSURF are studied as visual module candidate alternatives, where we call visual module in the subsystem of the VSLAM solution responsible for all the visual processing tasks. The main goal of this study is finding a visual solution capable of overcoming the limitations of SURF description due to illumination changes.

Extensive experiments were conducted to provide a deep analysis on the visual processing stages involved in the proposed VSLAM solution. Thus, feature detec-

tion and feature matching and also landmark management are topics in the scope of this chapter. The best tradeoff between accuracy of VSLAM and execution time is achieved.

The analysis in this Chapter has two distinct parts. The first emphasises on studying the influence on VSLAM of the most important visual parameters, section 5.2. The second part consists of a comparison of the navigation results obtained using tested techniques both for indoor and outdoor sequences, section 5.3 and section 5.4. It is showed that HMSURF based VSLAM is the best alternative in comparison to the other tested possibilities. Section 5.5 concludes with final discussions and remarks.

5.2 HMSURF characterisation

This section presents a thorough characterisation of our HMSURF-VSLAM implementation. Using an indoor dataset composed of 100 pairs of stereo images and the ground truth robot odometry corresponding to the travelled distance, the system is tested for different test conditions. The trajectory of the studied sequence corresponds to an arc that takes place on a flat indoor surface. These conditions allow us to consider the estimation of planar motion only, where two degrees x, y of freedom (DOF) are used to determine the position in the plane and a third ψ is used for the attitude. The data was collected from a Pioneer3 AT equipped with a Bumblebee stereo camera [89]. Figure 5.1 shows a few images extracted from the sequence.

A black-box testing technique has been chosen to analyse the system. Using the same input sequence of images one visual module parameter is changed at a time, while the rest of the systems parameters remain fixed, for the results generated here.

In addition, the robustness of the system has been studied by corrupting the

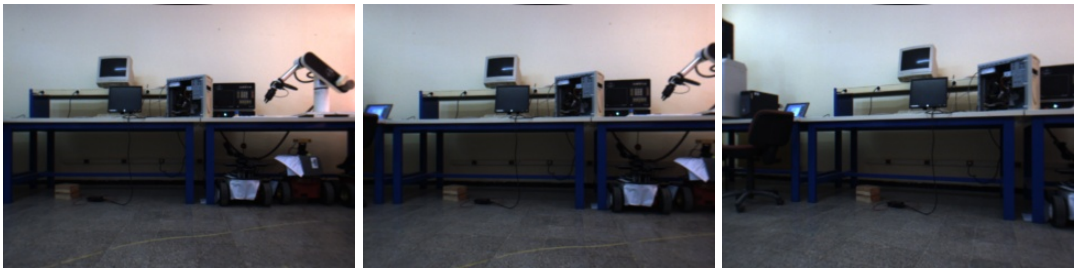


Figure 5.1: Sample images extracted from the experimental sequence.

system's input with noise of different intensity and nature. The noises used correspond to zero mean Gaussians, whose standard deviations are generated as percentages of the nominal input values. These percentages that determine the levels of noise are specified on the tables that summarise the experiment conditions for each of the experiment.

The three following parameters have been considered to be most influential:

- **Matching threshold:** Is the parameter responsible for the stereo matching of features detected in left and right images.
- **Number of detected features:** Is the parameter that determines the amount of information from the images that will be used.
- **Covariance matrix initialisation for new landmarks:** Is how the system to model the uncertainty of the new landmarks added to the system.

The rest of the influencing parameters have been tuned to optimise the visual module used in our VSLAM solution. However, we have decided that the parameters mentioned above are the most relevant and influential. This means excluding tuning the association threshold due to the large number of tuneable parameters.

5.2.1 Influence of the matching threshold

The importance of the matching threshold resides in the fact that it is responsible for rejecting the ambiguous matches caused by the similarity of more than one descriptor of the features in the same image. When two or more descriptors of the features in the right image are not different enough, their distances with respect to the descriptor of the feature to be matched in the left image will be similar. This is one of the causes of false matching that can be avoided by tuning the value of the matching threshold.

Here we test different values for the matching threshold for VSLAM based on proposed HMSURF, and show computational costs and estimation performance.

For this test 7 different threshold values $\{0.1,\ 0.2,\ 0.3,\ 0.4,\ 0.5,\ 0.6,\ 0.7\}$ have been considered. Note that the value of the matching threshold has to be contained between $[0, 1]$. Although at first sight the reader could consider the possibility of testing a bigger set of values, the fact that the optimal value for this threshold can differ depending on the nature of the images sequence, makes the sharpening of this dependant on the training sequences. Table 5.1 summarises other relevant experiment conditions.

The extreme values of the matching threshold have important implications.

Table 5.1: Matching threshold experimental conditions.

Experiment condition	Value	Units
Features	80	[#]
max. landmarks	20	[#]
Association threshold	0.15	[-]
Matching threshold	$\{0.1,\ 0.2,\ 0.3,\ 0.4,\ 0.5,\ 0.6,\ 0.7\}$	[-]
Velocity	120	[mm/s]
Angular rate	5	[deg/s]
Velocity noise std.	10	[%]
Angular rate noise std.	5	[%]
Error type	unbiased	[-]

Table 5.2: Summary of results varying the matching threshold.

Threshold	[-]	0.1	0.2	0.3	0.4	0.5	0.6	0.7
avg(e)	[mm]	38	41	21	25	14	14	48
max (e)	[mm]	85	80	50	70	32	43	90
avg(e_ψ)	[deg]	2.2	1.2	1.3	0.8	0.3	0.4	0.4
max (e_ψ)	[deg]	5.5	2.9	3.2	2.0	0.9	1.2	0.9

Let us recall that the matching threshold κ is utilised as in $\xi_1 = \kappa \cdot \xi_2$, where ξ_1 and ξ_2 are the distances from the descriptor that is intended to be matched with respect to the two best matches found for this among a set of candidates, section 3.4. Then when the matching threshold is set to be equal to zero it means that only exact correspondences between descriptor vectors are accepted as valid matches. On the contrary when the matching threshold is chosen to be equal to one it means that two descriptor can be equal, thus ambiguous matches are allowed.

Table 5.2, summarises the pose and attitude estimation errors computed as absolute values with respect to the real pose. The table shows averages in time of the location error avg(e) and yaw angle error avg(e_ψ), and the maximum values of the location error max (e) and the yaw angle error max (e_ψ). As it is seen from the table, the minimum errors are obtained for 0.5.

Figure 5.2 displays the pose and attitude errors versus matching threshold. Although the attitude error seems to decrease as the threshold is augmented, it is seen that the location errors look like a convex function, where values either too low or too high make the error increase.

Figure 5.3 summarises the time consumption corresponding to a full EKF iteration obtained for different threshold values. Displayed values are average values of the time expenses per time-steps.

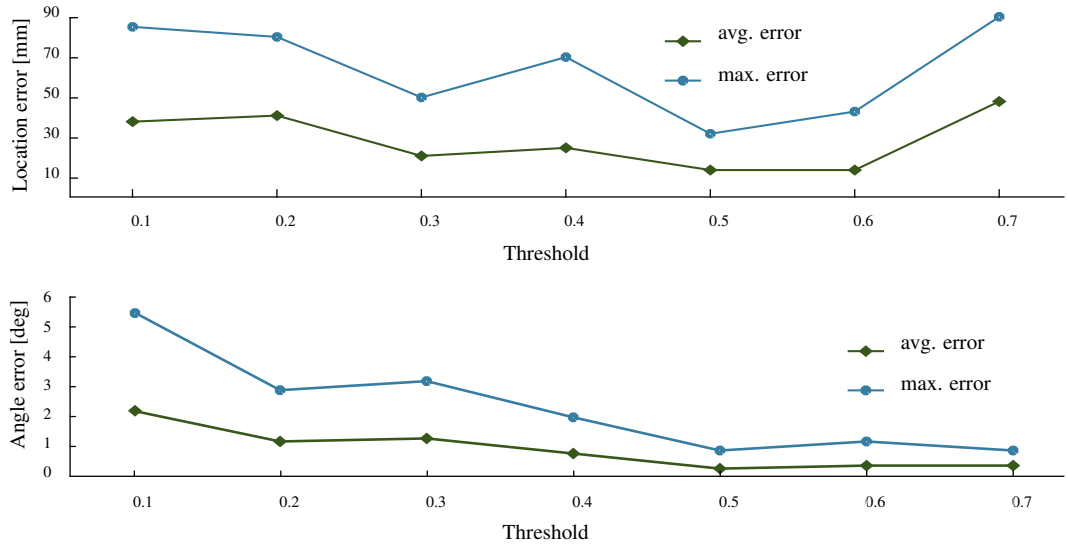


Figure 5.2: Pose and attitude errors varying matching threshold.

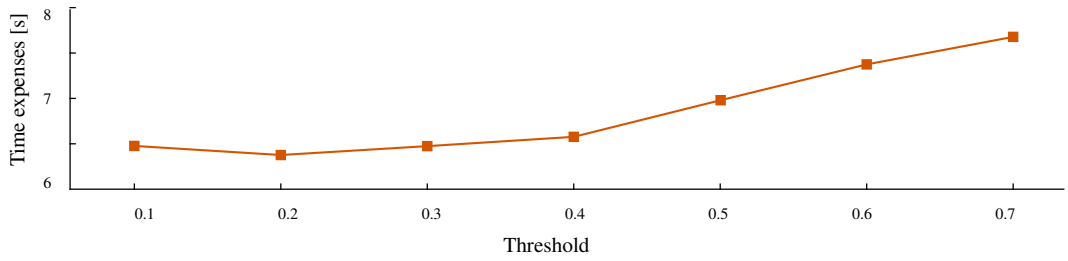


Figure 5.3: Time consumption per time-step [s] varying matching threshold.

Note then the time consumption increases by augmenting the value of the threshold. This is caused by the relaxation on the acceptance condition that allow a higher number of valid matches. The higher the threshold the more the number of accepted features.

Figure 5.4 shows the best pose estimation results obtained from this test. The reduction of the error in the pose estimation with respect to the odometry is at least 8*cm*. Similar improvement is observed on the angle estimation in Figure 5.6b.

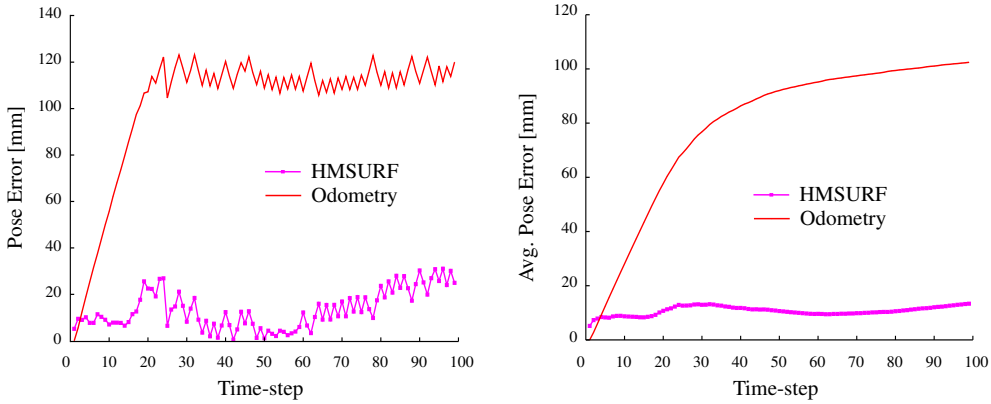
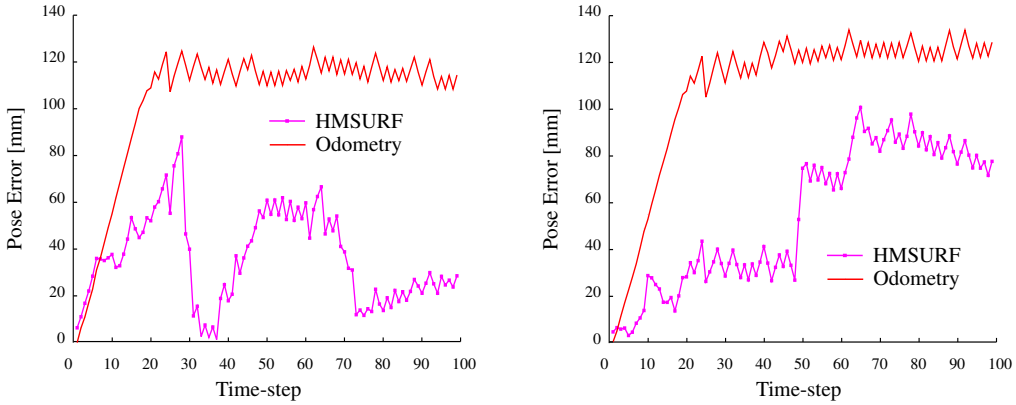


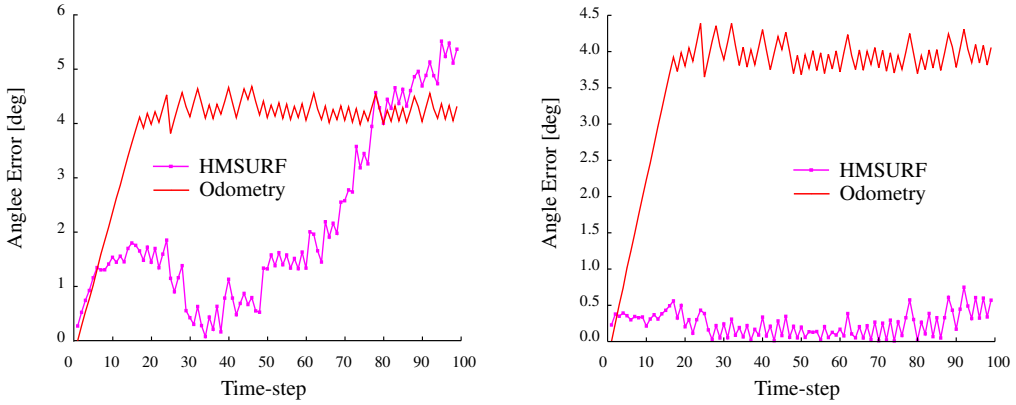
Figure 5.4: Pose error for optimal matching threshold conditions.



(a) Low matching threshold (0.2).

(b) High matching threshold (0.7).

Figure 5.5: Pose error for suboptimal matching threshold conditions.



(a) Low matching threshold (0.2).

(b) Optimal matching threshold (0.5).

Figure 5.6: Influence of the matching threshold on the attitude estimates.

When suboptimal matching thresholds are used accuracy losses can be observed on the estimation results as shown in Figure 5.5. The plot on the left-hand side of the figure was obtained using a low matching threshold (0.2). Conversely the plot on the right-hand side was obtained for a high matching threshold (0.7). It is seen that both cases are cause of pose estimation accuracy losses.

Similar results are obtained for the attitude estimation. Figure 5.6 presents the attitude estimation results obtained for an optimal matching threshold value (0.5) and a suboptimal value (0.2).

The matching threshold used for the stereo pair matching operation has important effects over the whole VSLAM process. Although for the shown cases there is not a case of divergence in the pose estimation, this could be an emerging problem caused by the selection of unappropriated matching threshold.

Low threshold values make the system less sensitive to the visual information. In this case, a number of correct features pairs will be rejected, entailing the loss of information usable for the VSLAM system updates. High matching thresholds allow matching of incorrect feature pairs in ambiguous environments where the feature characterisation and description is not strong enough.

Under these considerations, we recommend the use of intermediate values of matching threshold. The fact that the best results are achieved using a matching threshold of 0.5 for our batch of experiments does not mean that this value will perform well for all cases.

We also want also to emphasise that use of high matching threshold values aided by additional outlier rejection methods to guarantee a unique and reliable matching process should not be discarded for future research.

5.2.2 Influence of the detection threshold

A second experiment has been conducted to analyse the influence of the number of detected features on our HMSURF-VSLAM in terms of performance and time consumption.

The developed implementation allows us to decide the number of features extracted in the detection step. Indeed, the proposed method uses Harris detector for feature extraction and our implementation is able to select any number of features, N, whose cornerness value is the greatest.

For the test here, the number of detected features have been given the values $\{20,\ 40,\ 80,\ 150,\ 200\}$. Figures presented here show series of results named after the number of detected features as HMSURF #, where # is the number of detected points used for the corresponding series. Table 5.3 summarises other experiment conditions.

The VSLAM system has been tested adding different magnitude input perturbations. Two types of noises have been considered: biased and unbiased. The unbiased noises are just added to the input signals, while the biased also add an offset equal to the value of the noise standard deviation.

Table 5.3: Influence of the number of detected features, experimental conditions.

Experiment condition	Value	Units
Features	$\{20, 40, 80, 150, 200\}$	[#]
max. landmarks	20	[#]
Association threshold	0.15	[-]
Matching threshold	0.5	[-]
Velocity	120	[mm/s]
Angular rate	5	[deg/s]
Velocity noise std.	$\{5, 10, 35\}$	[%]
Angular rate noise std.	$\{2, 5, 20\}$	[%]
Error type	biased / unbiased	[-]

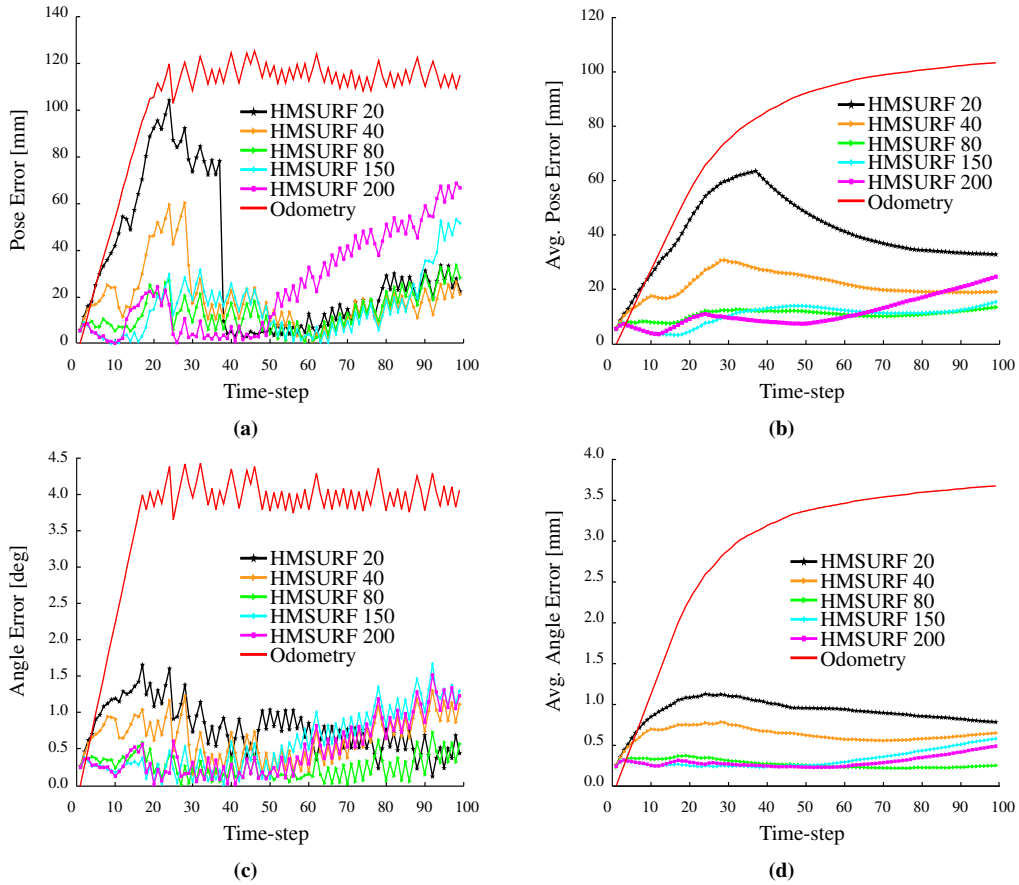


Figure 5.7: Pose and attitude errors varying the number of detected features.

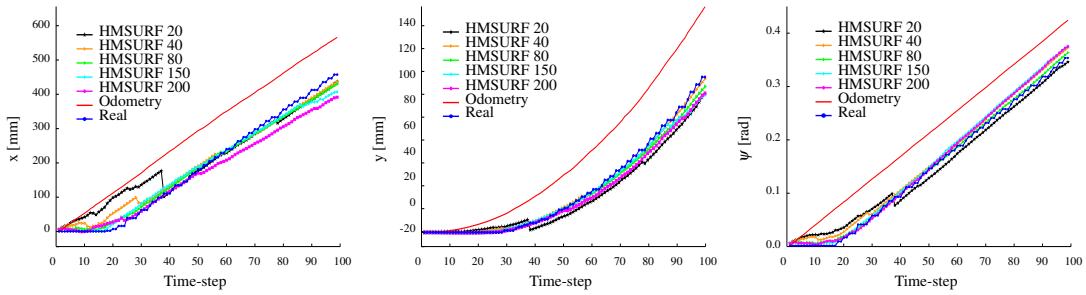


Figure 5.8: Estimates of the robot states varying the number of detected features.

Table 5.4: Summary of estimation results varying the number of features.

		Noise type	[-]	unbiased					biased				
		Features	[#]	20	40	80	150	200	20	40	80	150	200
Noise level	Low	$\text{avg}(e)$	[mm]	50	45	23	20	20	45	50	28	25	21
		$\max(e)$	[mm]	130	90	40	35	50	132	92	43	40	50
		$\text{avg}(e_\psi)$	[deg]	1.4	2.2	1.3	1.3	1.5	1.5	2.3	1.4	1.3	1.7
		$\max(e_\psi)$	[deg]	3.4	3.0	2.0	1.9	2.3	3.4	3.0	2.0	2.1	2.3
	Medium	$\text{avg}(e)$	[mm]	32	20	15	16	22	32	18	12	15	25
		$\max(e)$	[mm]	105	60	35	53	70	113	68	31	36	52
		$\text{avg}(e_\psi)$	[deg]	0.8	0.7	0.3	0.6	0.5	0.7	0.8	0.4	0.7	0.6
		$\max(e_\psi)$	[deg]	1.7	1.3	0.8	1.7	1.5	1.7	1.3	1.2	2.0	1.7
	High	$\text{avg}(e)$	[mm]	32	48	50	65	85	18	30	38	45	72
		$\max(e)$	[mm]	90	120	140	240	205	50	70	105	190	170
		$\text{avg}(e_\psi)$	[deg]	0.3	0.6	0.6	0.7	0.8	0.5	0.8	0.7	0.9	0.9
		$\max(e_\psi)$	[deg]	1.0	2.3	1.7	2.8	2.7	1.6	3.2	2.3	3.2	3.1

Table 5.5: Time expenses varying the number of detected features.

	Noise type	[-]	unbiased					biased				
	Features	[#]	20	40	80	150	200	20	40	80	150	200
Noise level	Low	[s]	0.25	0.40	0.70	1.20	1.60	0.26	0.46	0.75	1.28	1.70
	Medium	[s]	0.22	0.40	0.73	1.25	1.65	0.27	0.42	0.76	1.32	1.75
	High	[s]	0.25	0.42	0.75	1.30	1.72	0.26	0.43	0.74	1.26	1.68

For the first noise level (Low), deviations with respect to the nominal values of 5% and 2% of the velocity and angular rate are considered respectively. This means that, for a scenario where the robot moves with a constant velocity of 120 [mm/s], the perturbed velocity values will be contained in the interval [102, 138] [mm/s] for an unbiased noise and in the interval [138, 156] [mm/s] for a biased noise. These values are computed considering that the 99.7% of the values are within three standard deviations. Analogously, the medium noise corresponds to standard deviations of 10% and 5% of the nominal values of the velocity and angular rate. High noises are generated from standard deviations of 35% and 20% of the velocity and the angular rate nominal values.

Figure 5.7 shows the pose and attitude estimation results obtained for the different feature detection settings. These results are obtained for a medium size biased input perturbation. The right-hand side figures represent the average of the errors, that allows as to see the trend of the errors.

Figure 5.8 shows one of the obtained estimation results for each cartesian coordinate and the bearing angle of the robot of the different possibilities studied in this section.

Table 5.4 summarises estimation errors of the different series, both for pose and for attitude with respect to the real pose. Average in time of the pose estimation error $\text{avg}(e)$ and average in time of the yaw angle estimated error $\text{avg}(e_\psi)$ as maximum pose error $\max(e)$ and maximum yaw angle error $\max(e_\psi)$ are given. Best results are displayed in green, whereas weakest results are shown in red.

Table 5.5 summarises the computational cost of our HMSURF-VSLAM per EKF iteration. As expected, we can see that the more features are detected the higher the time consumption is. Despite the fact that it cannot be considered as a linear dependency, there is a clear trend that allows us to consider that the time expenses on the VSLAM based on HMSURF is proportional to the number of features acquired in the detection step. These time expenses are not only due to the time required for detection and description but also for the landmark association process. Taking into account this proportionality between the time expenses and the number of features used, it seems reasonable to think that a trade-off between both of them should be found to satisfy the accuracy requirements for the pose estimation without exceeding the available time.

From the experimental results the following can be added:

- For extremely intense noise perturbations, both biased and unbiased, HSURF-VSLAM performs better with a lower number of features, 20 in our experiments.
- For low to medium noise perturbations, both biased and unbiased, the estimation errors are reduced, or contained within reasonable boundaries, as the number of detected features augments.

For a trade-off solution, suitable in case of previously unknown noise perturbations, an intermediate number of features have to be used. The case of 80 features in our experimentation would be in this category.

For further analysis the direct effects of the variation of the number of features on the number of associated landmarks can be studied as part of future research.

5.2.3 Influence of the covariance matrix initialisation

When the descriptor of a pair of matched features is not associated with any of the existing landmarks in the VSLAM state vector, it is considered a new landmark that has to be added into the system. This entails modifications both on the state vector and on the process covariance matrix of the system.

The addition of a landmark to the state vector does not imply any difficulty. It consists in reconstructing the landmark from the pair of associated features through the inverse of the observation model and appending the values to the state vector.

The state covariance matrix is augmented by adding m lines and m columns to it, where m are the dimensions of the landmark, normally three. These lines and columns contain the information relative to the covariances between the landmark location and

Table 5.6: Influence of landmarks initialisation experiment conditions.

Experiment condition	Value	Units
Features	$\{40, 80\}$	[#]
max. landmarks	20	[#]
Association threshold	0.15	[-]
Matching threshold	0.5	[-]
Velocity	120	[mm/s]
Angular rate	5	[deg/s]
Velocity noise std.	10	[%]
Angular rate noise std.	5	[%]
Error type	unbiased	[-]

the rest of the system's states. The cross covariance terms that link the landmark position to the other states used to be set to zero, whereas the box corresponding to the terms of the landmark itself are conveniently initialised.

Some implementations initialise this covariance matrix box to using the identity matrix, which means that the 3D landmark position is considered to be exactly known. However, the landmark position is an estimation obtained through the inverse observation model which due to the nonlinearities will not be exact. Then initialising the covariance matrix box for the new landmark using an identity matrix will cause estimation errors. One reason to initialise this matrix to the identity is that the landmarks are considered stationary.

This section shows the benefits of modelling the uncertainties of the added landmarks by initialising the covariance matrix boxes for new landmarks as scalar matrix boxes. This alternative initialisation allows the system to cope with the measurement uncertainties present on the first estimation of the landmark position.

The experimental conditions for HMSURF-EKF VSLAM, used to test the different possibilities for the state covariance initialisation for new landmarks are summarised in Table 5.6.

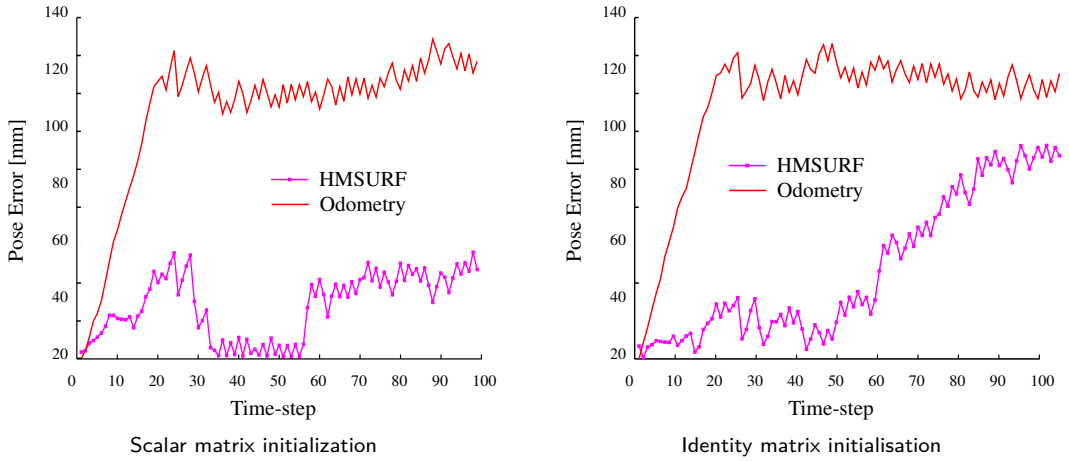


Figure 5.9: Pose error varying landmarks initialisation.

Figure 5.9 shows the type of results obtained using the proposed initialisation that adds a scalar matrix box into the covariance matrix and the results obtained when an identity matrix box is used instead. The examples correspond to a series for which the number of detected features is 40.

From the results shown here, it is clear that the trend as the peak of the errors using the identity matrix for the initialisation of new landmarks are worse than the ones obtained using the proposed strategy. For this test uncertainty on the feature position of 30mm is considered, this means that a scalar matrix containing 900 on the diagonal is used for the initialisation of the new landmarks.

5.3 Comparison of VSLAM using SURF and HMSURF

The previous subsection has shown some of the considerations that have to be taken into account in order for our HMSURF-VSLAM to behave as expected. In this section, two experiments are presented to compare the results obtained using the our new strategy as opposed to a SURF-VSLAM solution.

Table 5.7: SURF vs. HMSURF system experimental conditions.

Parameter	Value	Units
Velocity	120	[mm/s]
Angular rate	5	[deg/s]
Velocity noise std.	$\{10,20,30\}$	[%]
Angular rate noise std.	$\{5,20,30\}$	[%]
Error type	unbiased	[-]

Table 5.8: SURF vs. HMSURF visual module experimental conditions.

Parameter	SURF	HMSURF	Units
Features	80	40	[#]
max. landmarks	$\{10,20,40\}$	$\{10,20,40\}$	[#]
Association threshold	0.05	0.15	[-]
Matching threshold	0.5	0.5	[-]

For the experiment here, both SURF and HMSURF methods have been tuned to run in optimal conditions over the same sequence of stereo images. In order to compare the accuracy of the results fairly the input noises are exactly the same for both visual module possibilities. The motivation for this experiment is the collection of enough information to conjecture which of the visual processes is more likely to give better results for general VSLAM cases.

As in previous experiments in this chapter, the visual methods have been tested under different perturbation conditions. Likewise the maximum number of landmarks used for the mapping process has been given different values to test its influence here. The most relevant system settings and configurations of the visual models are summarised in Table 5.7 and in Table 5.8.

Table 5.9 shows values of the average pose estimation error avg(e) and average of attitude estimation error avg(e_ψ). Likewise, peak values of the pose and attitude estimation errors are also summarised in the table.

Table 5.9: Summary of results for SURF and HMSURF comparison.

Noise Level		Method	[-]	SURF			HMSURF		
		Landmarks	[#]	10	20	40	10	20	40
Low		$\text{avg}(e)$	[mm]	26	25	21	26	19	21
		$\max(e)$	[mm]	50	48	43	68	65	58
		$\text{avg}(e_\psi)$	[deg]	1.3	1.2	1.2	0.6	0.7	0.7
		$\max(e_\psi)$	[deg]	2.4	2.2	2.1	1.4	1.5	1.4
Medium		$\text{avg}(e)$	[mm]	34	36	34	23	25	22
		$\max(e)$	[mm]	73	70	60	47	55	40
		$\text{avg}(e_\psi)$	[deg]	1.3	1.3	1.2	0.5	0.7	0.9
		$\max(e_\psi)$	[deg]	3.7	3.8	3.6	2.1	2.6	3.1
High		$\text{avg}(e)$	[mm]	45	46	42	15	24	18
		$\max(e)$	[mm]	99	93	82	43	55	38
		$\text{avg}(e_\psi)$	[deg]	1.5	1.5	2.7	0.6	0.9	2.5
		$\max(e_\psi)$	[deg]	4.4	4.5	2.7	2.8	3.8	2.8

Table 5.10: Time expenses for SURF and HMSURF.

Noise level	Method	[-]	SURF			HMSURF		
	Landmarks	[#]	10	20	40	10	20	40
Low		[s]	0.58	0.72	0.54	0.44	0.53	0.42
Medium		[s]	0.68	0.69	0.54	0.52	0.51	0.43
High		[s]	0.64	0.62	0.53	0.47	0.48	0.42

Table 5.10 summarises the time expenses per EKF iteration using SURF and HMSURF as visual modules together with our VSLAM solution. One of the derived advantages of HMSURF with respect to SURF is time consumption. The results shown in this section show that the HMSURF-VSLAM system is able to perform better than the SURF-VSLAM system with a lower computational cost. Note that the number of detected features is twice as much for SURF than it is for HMSURF.

At the light of the results we can say that the HMSURF visual module leads to better results for the pose and attitude estimation than SURF when integrated into an EKF-VSLAM algorithm.

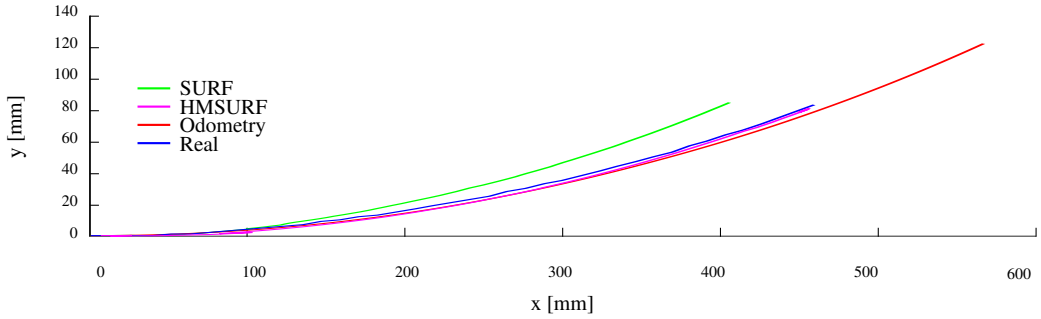


Figure 5.10: Trajectory estimates using SURF and HMSURF.

Figure 5.10 shows the trajectory estimation results obtained when medium intensity noise corrupts the system's input.

There is only one case where HMSURF solution seems to perform slightly worse than SURF. This case corresponds to an extreme scenario where the lack of balance between the number of landmarks used for the map and the number of features detected on the stereo image leads to atypical results. In the rest of scenarios HMSURF performs comparably or even better than SURF.

5.3.1 Harris corners with SURF (HSURF)

At this point it may look reasonable thinking that there is a possible development that has been under-considered or skimmed over when HMSURF was presented, section 3.8. A reader would wonder why Harris corners are not simply detected from original images and combined together with SURF descriptors without need for moment images to be computed.

Although the main reason for the moment images to be incorporated is the additional invariance to illumination changes that they induce, we consider that this Chapter is the right place to add the results obtained using Harris corners with SURF descriptors (HSURF) for the sake of completeness.

Table 5.11: SURF, HSURF & HMSURF system experimental conditions.

Parameter	Value	Units
Velocity	120	[mm/s]
Angular rate	5	[deg/s]
Velocity noise std.	10	[%]
Angular rate noise std.	5	[%]
Error type	unbiased	[-]

Table 5.12: SURF, HSURF & HMSURF visual module experimental conditions.

Parameter	SURF	HSURF	HMSURF	Units
Features	80	40	40	[#]
max. landmarks	20	20	20	[#]
Association threshold	0.05	0.15	0.15	[-]
Matching threshold	0.5	0.5	0.5	[-]

It is important recalling that both HSURF and HMSURF perform feature description over the original images.

This subsection presents a simultaneous comparison of the results obtained using the algorithms HMSURF, HSURF and SURF with our VSLAM implementation. Table 5.11 and Table 5.12 present the experiment conditions used for the test conducted to compare the three VSLAM visual modules.

As it is seen in Figure 5.11 the most accurate estimate for the robot's pose is obtained using HMSURF. This is also supported by the errors shown in Figure 5.12 where the error for HMSURF is lower that the error when SURF or HSURF are used. Looking at the left-hand side plots of this figure, we can say that HMSURF is the visual module that yields to the best VSLAM estimation from the 50th time-step.

Time consumptions using HSURF and HMSURF have been observed to be lower than using SURF. HMSURF provides the lowest time consumption due to the association time reduction that makes up even for the moment images computation.

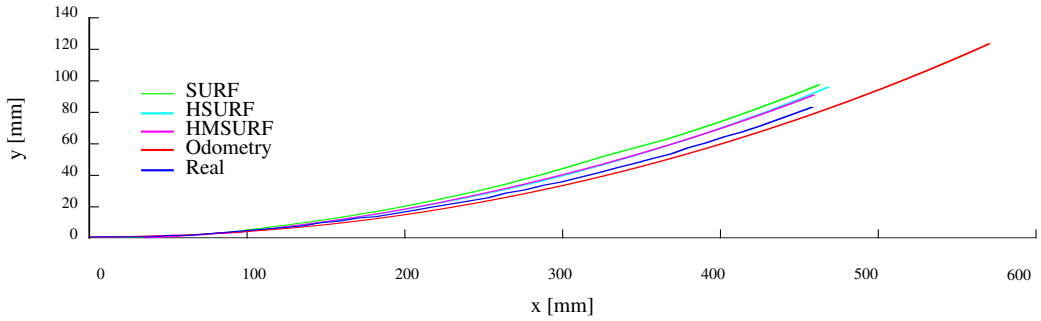


Figure 5.11: Trajectory estimates using SURF, HSURF & HMSURF.

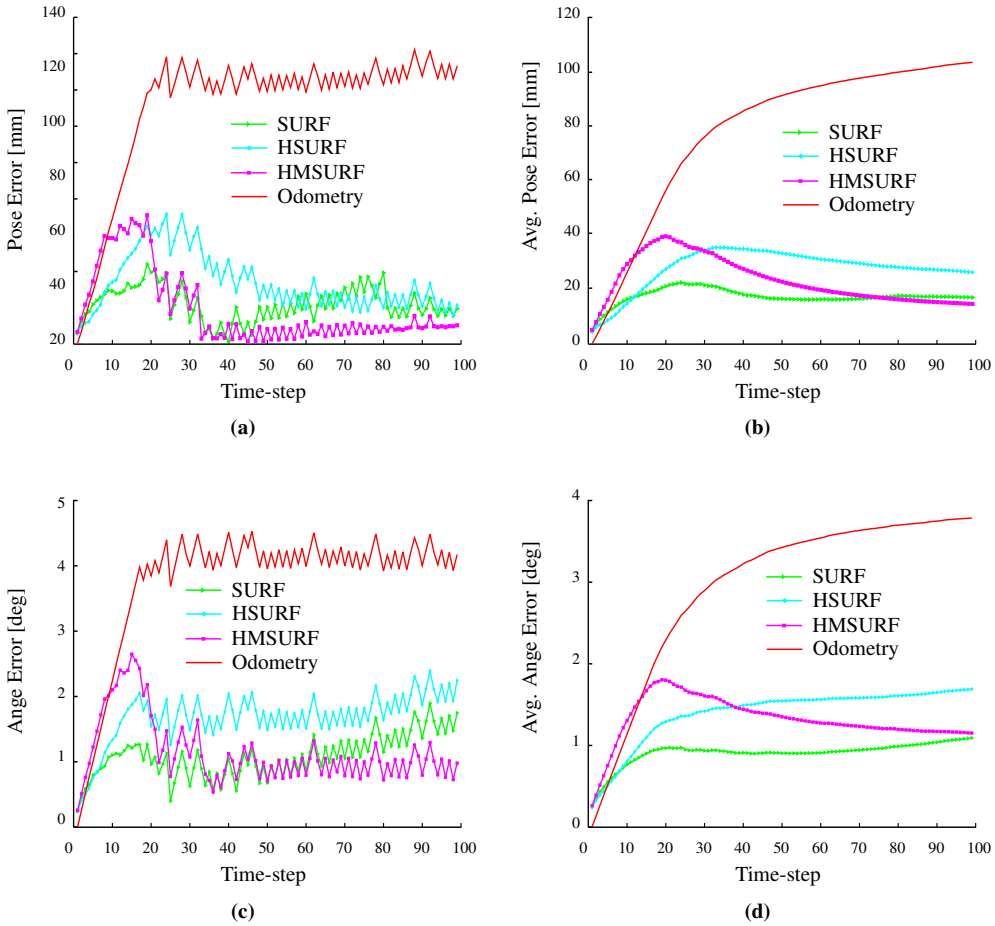


Figure 5.12: Pose and attitude errors using SURF, HSURF & HMSURF.

5.4 HMSURF based VSLAM for long range outdoor trajectory

This experiment used a challenging navigation dataset presenting a long enough trajectory of a ground vehicle equipped with a high quality stereo setup system and a high accuracy inertial navigation system [90, 91]. With this dataset the validation of our HMSURF-VSLAM is done and compared to the ground truth data and the system odometry data. Figure 5.13 shows some images extracted from the dataset used.

Figure 5.13: Sample of images used from large outdoor sequence.

Figure 5.14 shows the HMSURF-VSLAM estimates for the long vehicle trajectory in comparison to the odometry data. Furthermore, our HMSURF-VSLAM approach has shown good capabilities in coping with orientation changes that is one of the main reasons of the inertial navigation data drifts. The VSLAM solutions using HMSURF preforms better that the odometry results as seen in the figure.

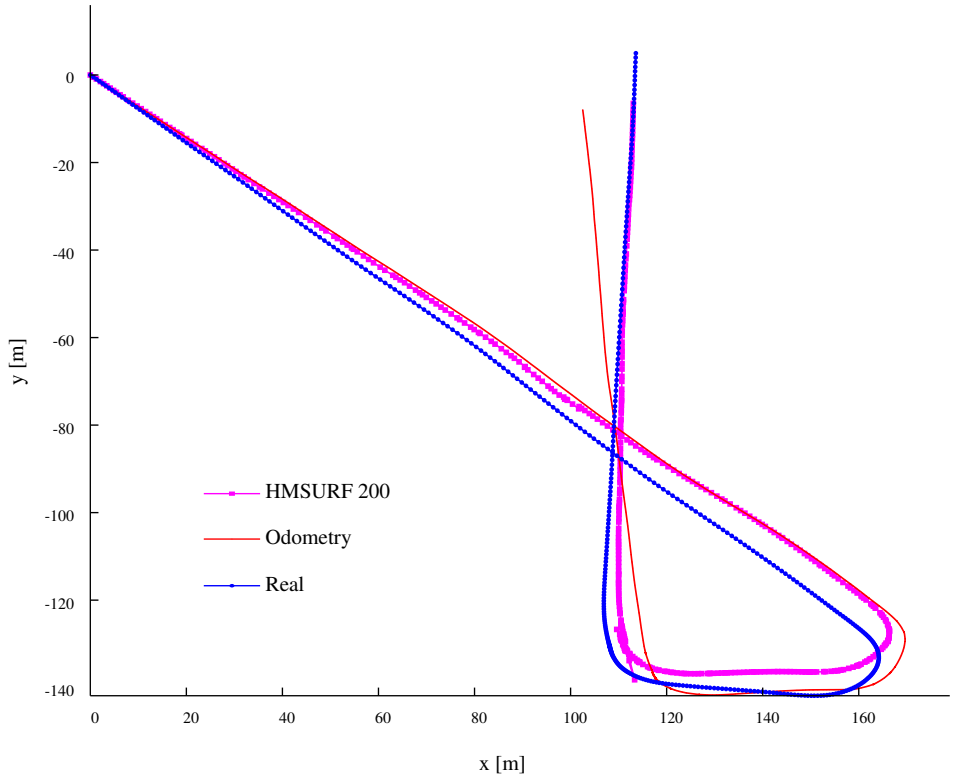


Figure 5.14: Trajectory estimates for large outdoor sequence.

5.5 Conclusion

In this Chapter we have developed an extensive testing of a new solution for VSLAM problem based on stereo cameras and HMSURF.

The visual processing of this solution was based on the integration of the Harris corner detector and the robust SURF descriptor run over moment images instead of the acquired intensity images. Analysis of the results from the experiments conducted, provided a good tool for evaluating how the visual processing is affecting the estimation obtained through VSLAM solutions. These analysis permit us to best tune the visual module parameters to improve results of the VSLAM solutions.

Finally it was shown that the stereo VSLAM solution proposed based on the HMSURF visual processing is efficient and reliable as a VSLAM alternative comparing with SURF-VSLAM.

Chapter 6

Robust Egomotion by Least Squares

6.1 Overview

Visual Odometry has been widely studied over the past decades. This is an alternative solution for autonomous navigation systems to the vision-based filtering approaches, explained in chapter 4 and chapter 5. As opposed to the solution presented before, this family of solutions does not model the state of the systems and its dynamics, relying on the system observations rather than on a transition model.

There are a significant number of applications oriented to assist manned vehicles, including on road vehicles [92, 93]. However, there are other applications were these vehicles are required to operate unmanned and autonomously because of security and safety requirements [94] or due to limited size of the platforms, e.g. an unmanned helicopter [95].

A recent example is the Mars exploration mission part of the European AURORA program which is a current challenging engineering problem where an unmanned robot known as "Rover" is proposed to traverse the terrain of Mars and collect samples in order to analyse the geochemical environment of the red planet.

The absence of GPS on the planet requires the localisation process of the vehicle to be highly dependent on its on-board sensors, one of which will be a stereo camera. Natural features can be detected by such sensors and processed to produce a 3D reconstruction of the environment which can be subsequently used by the robot to compute its relative location. This process is generally known as Structure From Motion (SFM) or egomotion. Some solutions on real time Structure From Motion also called monocular SLAM are available in the literature [45, 96]. These methods are often based on Kalman filtering that do not always provide the desired level of accuracy for motion estimation. The work in this instance is motivated by a robot equipped with a stereo camera set, where the rate of image acquisition is restricted as previously discussed by Johnson *et al.* [97], where a scheme similar to the Visual Odometry (VO) proposed by Nister *et al.* [98] is the chosen solution.

A desire to exploit the available hardware on a reduced size platform free of filtering approaches has motivated the work shown in this chapter. Furthermore we want to quantify the constraints of stereo visual odometry techniques on long range trajectories.

The Chapter is organised as follows: section 6.2 is a review of the visual and motion estimation techniques used in our least square solution. Section 6.3 presents theoretical analysis of the selected techniques on synthetic data scenarios. Section 6.4 shows the effectiveness of our stereo egomotion over a long-range trajectory. Finally, section 6.5 highlights main findings of the work [99].

6.2 Methodology

This section aims to recall the basis of some of the used techniques, while the reasons for taken for our implementations are conveniently justified and explained. Firstly, the management of the visual information is briefly treated in the subsection "Features detection and tracking", subsection 6.2.1, recalling some ideas from chapter 3. Secondly, singular value decomposition (SVD) and Quaternion motion estimation approaches are reviewed here as suitable least square solutions for the stated problem, subsection 6.2.2. A brief introduction to the random sample consensus (RANSAC) algorithm and its specific application in our work is presented at the end of this section.

6.2.1 Feature detection and tracking

Feature detection and tracking is used in this Chapter only when we adapted the motion estimation techniques for real image data. In spite of that, for all the experimental results of this work the need for descriptors is present.

Indeed, the detection step is assumed to be done when synthetic data are used. Nonetheless, a way to do the landmarks tracking is indispensable. In order to resolve this, artificial descriptors of 64 elements have been created and uniquely assigned to each synthetic landmark to label them and allow its recognition at every time.

On the other hand, for real images and based on the positive results achieved when combining Harris corners with SURF descriptors [59] (chapter 3), the selected solution consists of a combination of the good features to track explained in [63] by J. Shi *et al.* and SURF descriptors [28] to characterise the detected features. The reliability of the SURF matching combined with the robust corners detection has previously shown us to give better results than the whole SURF method itself [59].

Two different detection possibilities have been tested for the real data experiments. The first one is HSURF while the second one is HMSURF. These two detection alternatives are introduced and analysed in chapter 3 and chapter 5 respectvely.

6.2.2 Motion estimation

Many solutions can be found to solve the egomotion problem [23]. Some of these solutions are only based on numerical methods for over constrained systems, while other approaches are oriented to the rejection of the data that is not reliable, the *outliers*. This section shows two approaches to estimate the motion of the robot using the information of a set of 3D points in its environment.

The first approach for the motion estimation from the 3D information provided by the stereo imaging system originates from the Singular Value Decomposition (SVD) factorisation method.

The second approach, used in the present work, to calculate an estimated robot motion is the so-called "Quaternion motion estimation" technique.

6.2.2.1 Singular Value Decomposition (SVD)

For every linear system of the form $\boldsymbol{A}\,\boldsymbol{z} = \boldsymbol{b}$ where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ $(m \geq n)$, there exists an SVD factorisation of the system matrix $\boldsymbol{A}$, (6.1), such that the pseudo-inverse matrix $\boldsymbol{A}^{+}$ can be obtained as in (6.1). On these conditions, the solution of the system can be computed as shown in (6.3). Hence, the SVD factorisation is a way of calculating a least square solution for over constraint systems. The rectangular matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times n}$ is a nonnegative and diagonal matrix, containing the *singular values* of the matrix $\boldsymbol{A}$.

$$\boldsymbol{A} = \boldsymbol{U}\ \boldsymbol{\Sigma}\ \boldsymbol{V}^{*} \tag{6.1}$$

$$\boldsymbol{A}^{+} = \boldsymbol{V}\ \boldsymbol{\Sigma}^{+}\ \boldsymbol{U}^{*} \tag{6.2}$$

$$\boldsymbol{z} = \boldsymbol{A}^{+}\boldsymbol{b} \tag{6.3}$$

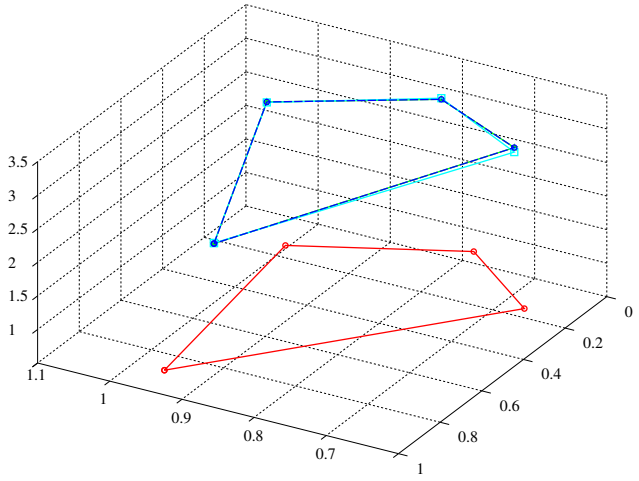
where the matrix $\boldsymbol{\Sigma}^{+}$ is the pseudo inverse of the matrix $\boldsymbol{\Sigma}$. The matrices $\boldsymbol{U} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ are unitary matrices formed by the sets of orthonormal input and output basis vectors respectively. The index * indicates conjugated and transposed.

In the case of a moving robot the rotation matrix, $\boldsymbol{R}$, and the displacement vector, $\boldsymbol{t}$, are the unknowns on the motion estimation problem. Under the assumption of a rigid solid environment, where the position of a known set of landmarks with respect to the mobile reference system is $\{\mathbf{x}_k^{(i)}\}$, and after the motion, $\{\mathbf{x}_{k+1}^{(i)}\}$, we can write the following equation for every landmark (i):

$$\mathbf{x}_k^{(i)} = \boldsymbol{R}_{k\to k+1} \cdot \mathbf{x}_{k+1}^{(i)} + \boldsymbol{t}_{k\to k+1} \tag{6.4}$$

where $\boldsymbol{R}_{k\to k+1}$ represents the rotation from k to $k+1$ and $\boldsymbol{t}_{k\to k+1}$ represents the translation. In order to alleviate the notation they are commonly written as $\boldsymbol{R}$ and $\boldsymbol{t}$.

To solve the system composed of the equations for every landmark through the SVD factorisation, the equations must be recast writing the elements of the unknowns, $\boldsymbol{R}$ and $\boldsymbol{t}$, into a vector $\boldsymbol{z}$. We will normally set a vector $\boldsymbol{z} \in \mathbb{R}^{12}$ composed of nine elements that define a rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$ and three for the displacement vector $\boldsymbol{t} \in \mathbb{R}^{3}$, although its dimension can be reduced when less Degrees Of Freedom (DOFs) are required.



$$\boldsymbol{R} = \begin{bmatrix} 0.99 & -0.12 & -0.09 \\ 0.14 & 0.98 & -0.17 \\ -0.06 & 0.18 & 0.98 \end{bmatrix}$$

$$\boldsymbol{t} = \begin{pmatrix} 0.00 & 0.00 & 2.00 \end{pmatrix}^T$$

$$\hat{\boldsymbol{R}} = \begin{bmatrix} 0.99 & -0.12 & 0.09 \\ 0.14 & 0.98 & -0.17 \\ 0.09 & 0.59 & 1.89 \end{bmatrix}$$

$$\hat{\boldsymbol{t}} = \begin{pmatrix} 0.00 & 0.00 & 0.94 \end{pmatrix}^T$$

Figure 6.1: Singular Value Decomposition for motion estimation.

Figure 6.1 illustrates how motion can be estimated using SVD. On this example, a set of 4 points (red) originally contained in the ground plane OXY, are transformed by applying the translation vector $\boldsymbol{t}$ and the rotation matrix $\boldsymbol{R}$ into a new set (green). White noise $w_i \sim N(0, 0.025 * t_i)$ is added into each of the coordinates of the transformed points. From the set of resulting points (cyan) SVD is applied to obtain $\hat{\boldsymbol{R}}$ and $\hat{\boldsymbol{t}}$. The estimate of the transformation is ultimately used to compute the approximated points (blue).

It can observed from the example shown in Figure 6.1 that SVD does not guarantee the orthogonality of the estimated rotation matrix $\hat{\boldsymbol{R}}$. Likewise, this example at Figure 6.1 shows that, nonetheless the approximation of the points (blue) lie very close to the actual transformed points (green), the estimates of the rotation and translation are dramatically distorted for small perturbations on the input. The root mean square error (RMS) induced by the perturbation here is as low as 0.01.

6.2.2.2 Quaternion motion estimation

This technique, proposed in 1987 by Berthold K. P. Horn [100] is a closed-form solution to the least square problem and a suitable solution for over-constrained systems. This statistical method uses the properties of the quaternion representation to find an approximation for the rotation and translation of the robot.

For two corresponding landmark sets, $\{\mathbf{x}_k^{(i)}\}$ before motion and $\{\mathbf{x}_{k+1}^{(i)}\}$ after the motion, under the assumption that the environment behaves as a rigid solid, as shown in (6.4), the mean vectors and the cross-covariance matrix of the sets are computed as:

$$\boldsymbol{\mu_k} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_k^{(i)} \tag{6.5}$$

$$\boldsymbol{\mu}_{k+1} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_{k+1}^{(i)} \tag{6.6}$$

$$\boldsymbol{\Sigma}_{k,k+1} = \frac{1}{n}\sum_{i=1}^{n}\langle \mathbf{x}_{k+1}^{(i)}, \mathbf{x}_k^{(i)}\rangle - \langle \boldsymbol{\mu}_{k+1}, \boldsymbol{\mu_k}\rangle \tag{6.7}$$

where $\langle \mathbf{x}, \mathbf{y}\rangle$ denotes the dot product of any pair of vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$.

From the calculated cross-covariance, the matrix $\boldsymbol{Q}$ is computed:

$$\boldsymbol{A} = \boldsymbol{\Sigma}_{k,k+1} - \left[\boldsymbol{\Sigma}_{k,k+1}\right]^T \tag{6.8}$$

$$\boldsymbol{\Delta} = \begin{pmatrix} a_{23} & a_{31} & a_{21} \end{pmatrix}^T \tag{6.9}$$

$$\boldsymbol{Q}(\boldsymbol{\Sigma}_{px}) = \begin{bmatrix} \mathrm{tr}(\boldsymbol{\Sigma}_{k,k+1}) & \boldsymbol{\Delta}^T \\ \boldsymbol{\Delta} & \boldsymbol{\Sigma}_{k,k+1} + \left[\boldsymbol{\Sigma}_{k,k+1}\right]^T + \mathrm{tr}(\boldsymbol{\Sigma}_{k,k+1}) \cdot \boldsymbol{I}_3 \end{bmatrix} \tag{6.10}$$

where $\mathrm{tr}(\boldsymbol{\Sigma}_{k,k+1})$ is trace of the matrix $\boldsymbol{\Sigma}_{k,k+1}$ and $\boldsymbol{I}_3 \in \mathbb{R}^{3\times 3}$ is the identity matrix.

It is given that the eigenvector associated to the largest eigenvalue of this matrix $\boldsymbol{Q}$ is the quaternion $\boldsymbol{q_R} = (\ q_0 \quad q_1 \quad q_2 \quad q_3\ )^T$ corresponding to the least squares solution for a rotation transformation from $\{\mathbf{x}_k^{(i)}\}$ to $\{\mathbf{x}_{k+1}^{(i)}\}$. Thus, the rotation matrix associated to the quaternion $\boldsymbol{q_R}$ is written as seen in (6.11). The translation vector is computed based on the rotation matrix.

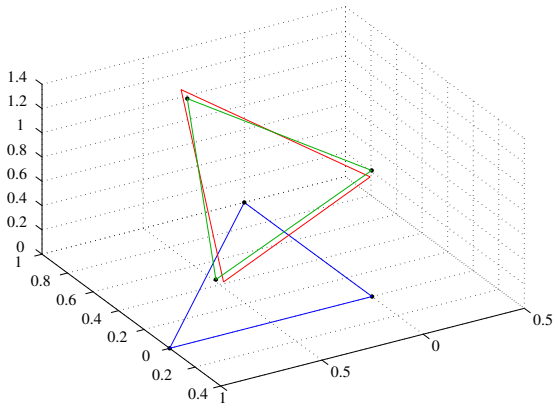
$$\boldsymbol{R} = \begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2+q_2^2-q_1^2-q_3^2 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2+q_3^2-q_1^2-q_2^2 \end{bmatrix} \tag{6.11}$$

$$\boldsymbol{t} = \boldsymbol{\mu}_k - \boldsymbol{R}\,\boldsymbol{\mu}_{k+1} \tag{6.12}$$

Figure 6.2 shows some motion estimates using the method presented in this section. The original transformation composed of the rotation $\boldsymbol{R}$ and the translation $\boldsymbol{t}$, shown at the bottom of the page, is applied to the original set of points (blue). After that, the results are perturbed adding some white noise into the values (green). This perturbation emulates the observation error on landmark positions.

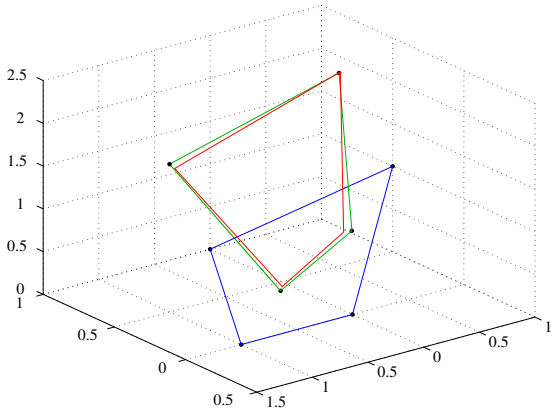
The approximated position of the landmarks obtained from the original set using the estimated motion are displayed in red. Estimation results are shown for different sizes of observed landmark sets, 3, 4 and 5 points.

$$\boldsymbol{R} = \begin{bmatrix} 0.94 & -0.34 & -0.09 \\ 0.32 & 0.93 & -0.17 \\ 0.14 & 0.13 & 0.98 \end{bmatrix} \qquad \boldsymbol{t} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$



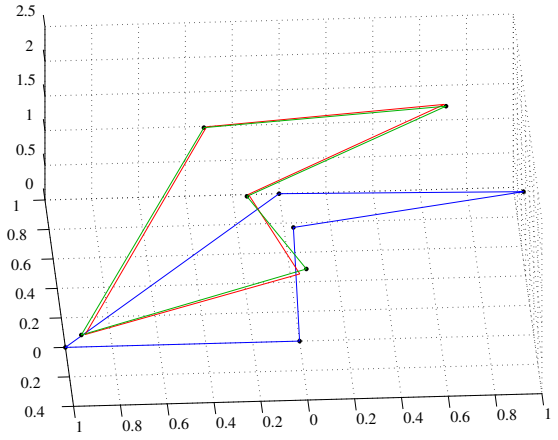
$$\hat{\boldsymbol{R}}_3 = \begin{bmatrix} 0.92 & -0.35 & -0.17 \\ 0.31 & 0.92 & -0.22 \\ 0.23 & 0.15 & 0.96 \end{bmatrix}$$

$$\hat{\boldsymbol{t}}_3 = \begin{pmatrix} -0.00 & 0.01 & 0.98 \end{pmatrix}^T$$



$$\hat{\boldsymbol{R}}_4 = \begin{bmatrix} 0.95 & -0.32 & -0.02 \\ 0.31 & 0.94 & -0.15 \\ 0.07 & 0.14 & 0.99 \end{bmatrix}$$

$$\hat{\boldsymbol{t}}_4 = \begin{pmatrix} -0.08 & -0.00 & 1.00 \end{pmatrix}^T$$



$$\hat{\boldsymbol{R}}_5 = \begin{bmatrix} 0.94 & -0.32 & -0.12 \\ 0.29 & 0.93 & -0.21 \\ 0.18 & 0.16 & 0.97 \end{bmatrix}$$

$$\hat{\boldsymbol{t}}_5 = \begin{pmatrix} 0.00 & -0.01 & 0.98 \end{pmatrix}^T$$

Figure 6.2: Quaternion motion estimation.

6.2.2.3 Random Sample Consensus algorithm

RANSAC, on abbreviation of "Random Sample Consensus", is a well-known iterative algorithm used to estimate parameters of a mathematical model. Based on the idea that not all the observed data is reliable, this method classifies them in two different sets, inliers and outliers.

The target of this iterative method is to obtain, in a reduced number of attempts, a good estimation of the model that describes the dominant transformation of the input-output pairs. However, the algorithm does not guarantee the correctness of the solution. What is guaranteed is obtaining a solution biased to the trend of the data. For a mathematical model with n parameters to be determined the process comprises the next steps:

(1) A random set of n inputs with theirs respective n outputs is used to calculate the first estimation of the transformation.

(2) All the inputs are transformed using the estimation computed at step 1. The distance from the estimated outputs to the real outputs is computed.

(3) A fixed distance threshold is the decision parameter to split the data into two sets: the *support group* and the *non-support group*. If the support group is big enough step 4 is computed, otherwise one of the following options apply:

   (a) If the attempt is the last one allowed, step 4 is computed from the support group containing more members.

   (b) If it is not the last attempt the algorithm resumes from step 1, adding one to the attempts counter.

(4) All the members in the support group are used to compute the final estimation of the parameters. This is an over-constrained system where SVD or any other least square solution can be used.

The RANSAC algorithm is commonly applied in the literature [91, 101] to reject mismatches coming from the stereo matching or from landmarks tracking. Despite of that this is not the only reason why we are using the algorithm in this chapter. Some of the results that we show were obtained using synthetic data. In this case, there are no outliers to be rejected coming from mismatching of features or wrong association of landmarks. However, when the value of the synthetic feature location is truncated to certain precision, reprojection errors appear as a consequence. Then, RANSAC can still be useful as a technique to refine the rotation and translation estimations. Indeed, to generate the synthetic data used in this chapter, a stereo camera set was modelled. This allowed us to test the influence of the precision loss occurred in the digitisation step preformed in the cameras in the projection process. This precision loss that takes place at image level implies quality losses on the accuracy of the 3D reconstructed coordinates.

Hence, RANSAC is not only used for the rejection of outliers appearing in real images sequences, but also over the synthetic data as it has shown to improve results because of the mitigation of the influence of the poorly located landmarks due to re-projection errors.

6.3 Egomotion analysis from synthetic data

This section shows a few egomotion results obtained over synthetic data using the previously explained techniques.

To evaluate motion estimation independently from other aspects influencing the visual navigation performance, i.e. feature extraction and matching, data has been generated such that detection and description of features has been dropped out of this analysis. Instead, information of the features is inputed in the system to perform 3D reconstruction and motion estimation. We made this possible by creating synthetic 3D scenarios composed of nearly one thousand points A model of a camera stereo set, composed of two low-resolution cameras 320×240 [pixels] with 12 [cm] of baseline, has been used to project scenario points in the camera frames. This has allowed us to study the influence on the estimators caused by variations in the sub-pixel precision of the feature locations. Figure 6.3 shows an example image of what the cameras on-board of the virtual robot would capture from the created 3D scenario.

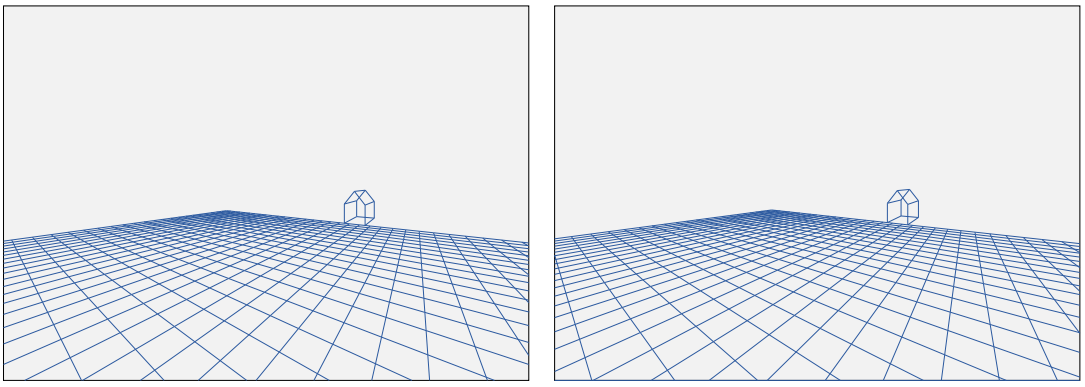


Figure 6.3: Sample of synthetic stereo images

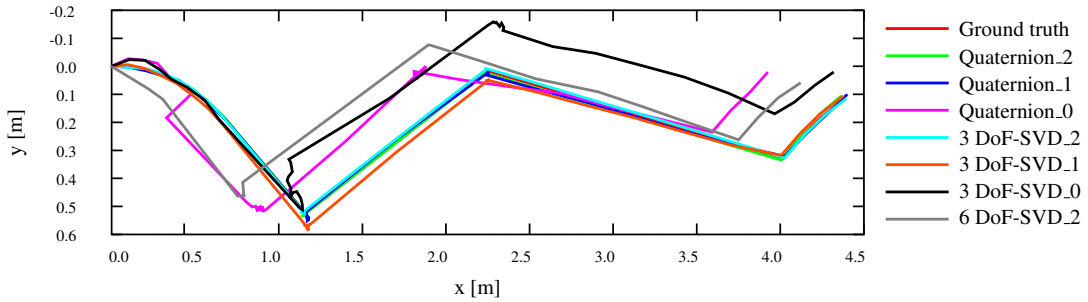


Figure 6.4: Trajectory estimation for different levels of precision.

Generated data corresponds with every kind of planar motions: pure rotation, pure translation and their combinations. The 3D position of the points is available at every time, thus the ground truth can be compared with the estimation results.

Features are labelled with artificial created descriptors, which allow their unique recognition for stereo matching and tracking.

Figure 6.4 shows the results of the trajectory estimation. The series correspond to estimations carried out by SVD and quaternion method, both computing 6 DOF and a DOF-reduced SVD where only 3 DOF are compute. Determining only plane rotation and both plane displacements. There are three series for each of the quaternion + 3-DOF-SVD method. Each of these series corresponds with a different level of precision on the feature locations. The number appended to each series name represents the number of decimal places (1, 2 or 0) to which the feature location value is truncated. For the SVD with 6 DOF only a series corresponding to two decimals places is shown because results obtained with lower degrees of precision were significantly worse.

Figure 6.5 corresponds to the 3D location errors of the robot, computed as the Euclidean distance between the estimated locations and the ground truth location. Figure 6.6 corresponds to the heading angle errors, computed as the absolute difference between the ground-truth heading angle of the robot and its estimations.

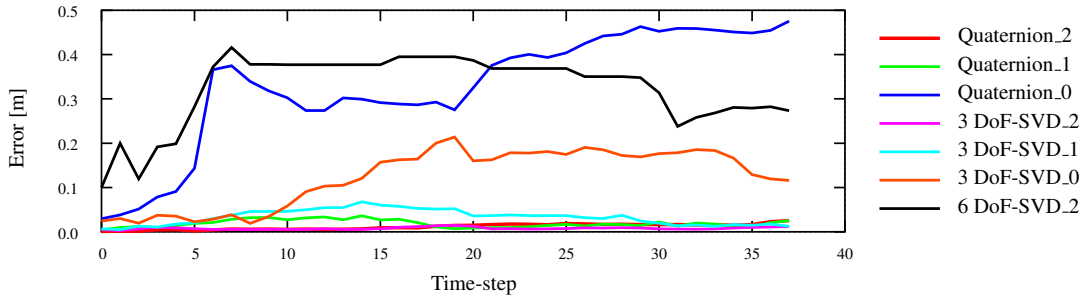


Figure 6.5: Location error for the different estimators and levels of precision.

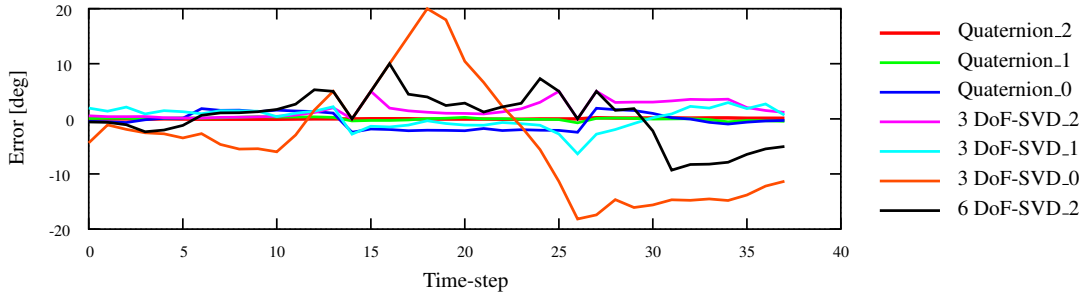


Figure 6.6: Heading angle error for different estimators and levels of precision.

According to the results for trajectory estimation, we see that both methods 3-DOF-SVD and quaternion behave similarly for every level of precision on the input feature locations following closely the ground truth trajectory. It is seen that all the presented methods are able to behave correctly in a qualitative way.

As expected, it is seen that the higher the precision in the features locations the better the results are. This is due to the improvement on the features location, that makes the 3D reconstruction error to diminish as the 2D uncertainty is reduced.

It is also seen from the results, that both quaternion method and 3-DOF-SVD method give similar results, for those series where the location of the features is provided with precision up to the first decimal on the sub-pixel position. However, when the sub-pixel accuracy of the feature location is removed, the case of zero decimals, the 3-DOF-SVD method fails in the heading angle estimation, whereas the quaternion method gives worse results for the positioning.

The 6-DOF-SVD method rarely converges and it only happens after a exhaustive tuning process by varying the RANSAC parameters. Even in the best case, when it converges, the highest level of accuracy in the sub-pixel position of the features is required. In spite of that, the results obtained are worse for this method than the ones for 3-DOF-SVD method or the quaternion method in the cases where only pixel precision is provided.

Although 3-DOF-SVD gives, in every case, comparable results to the ones obtained by the quaternion method, the constraint on the number of DOF will limit its usability for those cases where only the plane motion estimation is required.

6.4 Egomotion results for real data

This section shows navigation results for the previously described motion estimation techniques applied to a vehicle traveling through an urban environment.

This outdoor vehicle sequence collected in Karlsruhe consists of pairs of high quality images, with a resolution of 1344×372 pixels after rectification [90]. The stereo set setup is composed of two independent cameras with a baseline of 57 cm. The picture in Figure 6.7 shows a sample image extracted from the sequence.

The positioning information acquired from an OXTS RT 3000 GPS/IMU is considered ground truth odometry, labelled as GPS/INS in this chapter, and considered as the ground truth reference due to its high accuracy. In Figure 6.8, the projection in horizontal plane of the vehicle trajectory from the GPS/INS information is superimposed with our egomotion estimation results. Motion starts from the leftmost point of the figure, where a traversed distance of about 300 metres is shown.

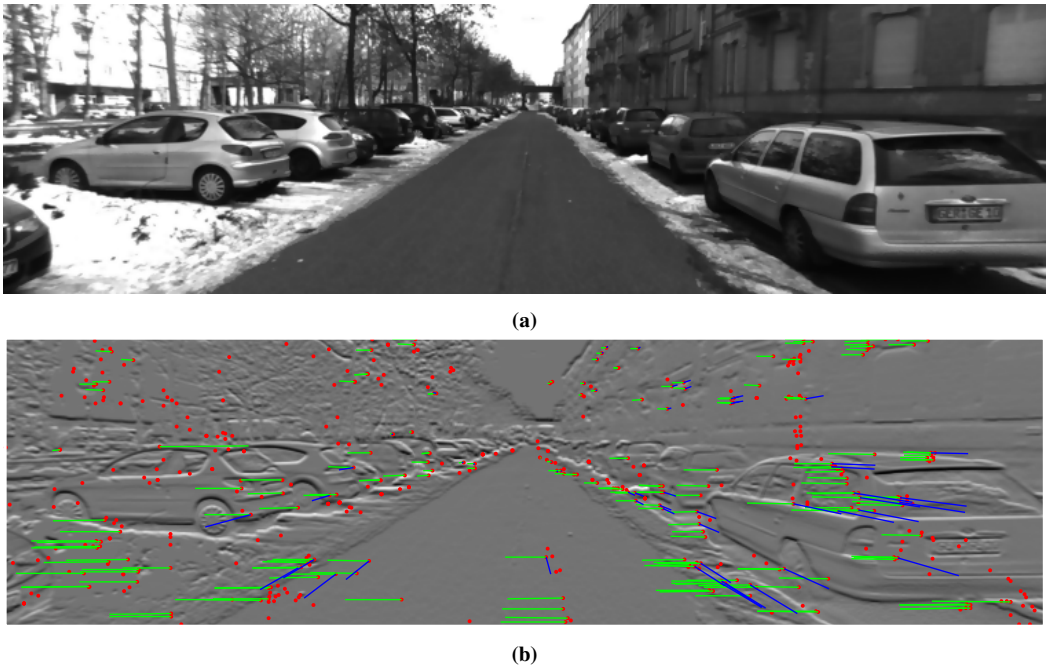


Figure 6.7: Image examples: (a) example image from the sequence used (b) example moment image, where detected features (red), disparities of stereo matched points (green) and landmark association segments (blue) are displayed.

Figure 6.8 shows a series of trajectory estimation results labelled as HMSUF (Harris Moment SURF), Harris SURF (HSURF), and KLT. This later presents the motion estimation results based upon applying KLT as a detection/matching prior applying quaternion motion estimation scheme. Indeed, KLT was used to compute the stereo matching between left and right images required for 3D reconstruction step and also for sequential association. Although there are not major issues when KLT is deployed for stereo matching, the nature of the algorithm, that assumes small differences between the pair of images where the optical flow is computed, makes it less suitable for sequential tracking and provides a more important number of outliers comparing to HSURF and HSURF techniques. As it can be seen from the figure, the best performance obtained is achieved by HMSURF based motion estimation technique.

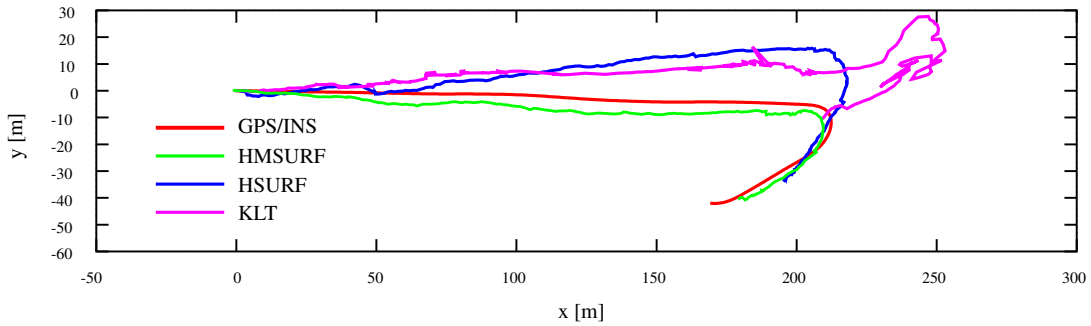


Figure 6.8: Trajectory. GPS/INS and estimated through Quaternion with RANSAC.

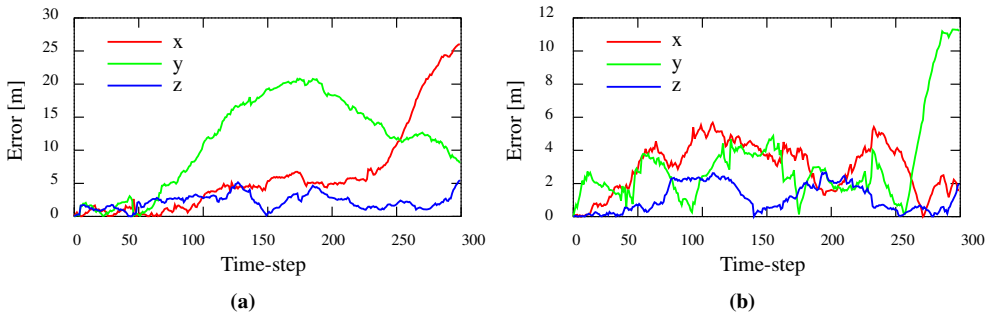


Figure 6.9: Estimation error using Quaternion-RANSAC (x, y, z) w.r.t. GPS/INS vs. time-step, (a) for HSURF and (b) for HMSURF.

For a detailed analysis of error results obtained for each coordinate, Figure 6.9 shows the error values vs time. In this graphic it is seen that the error is lower than 12 metres for HMSURF and lower than 30 metres for HSURF. It is also observed for the HMSURF, that during the main part of the travel the error in x, y and z below 6 metres, only reaching a higher error in y at the end of the test.

The evolution of the z location of the vehicle is shown in Figure 6.10. This is important as it implies that 6 DOF are required for the estimation, therefore, results obtained by SVD for 3 DOF is not a suitable technique due to its limitation to compute only plane motions. The SVD method with 6 DOF is likewise discarded because of the convergence problems related in the previous section.

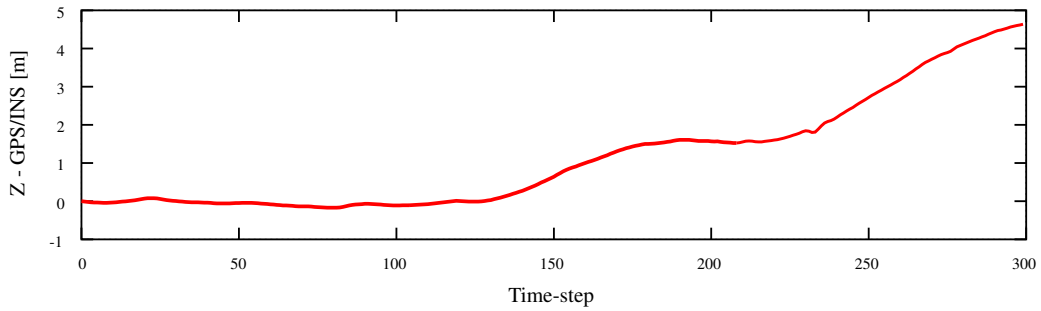


Figure 6.10: z coordinate (GPS/INS data) evolution along the sequence.

Looking at the variation in z, Figure 6.10, it is seen that the vehicle travels up-slope twice. The first one starts around the time-step 125 till the time-step 190, while the second one takes place from the time-step 225 to the end of the sequence. In spite of that, the estimation of the z coordinate, according to the errors shown in Figure 6.9, does not appear to be related to those slopes, neither for HSURF nor for HMSURF detection methods.

In Figure 6.11 the location error as Euclidean distance from the estimation to the GPS/INS location is shown for the 3D positioning. From these results we can say that the studied methods are suitable to estimate and distinguish the different nature of motions: rotation, translation and a combination of them along long range trajectories. In this real context application Quaternion-RANSAC method shows a maximum error below 12 metres when combined with HMSURF and below 28 metres when combined with HSURF, after travelling around 270 metres. From this analysis, it is seen that the combination of HMSURF for detection/description and Quaternion-RANSAC for motion estimation leads to the best results.

The fact that the Quaternion-RANSAC method is not constrained in DOF makes it the appropriate method for previously unknown motions or simply for egomotion systems in unknown environments

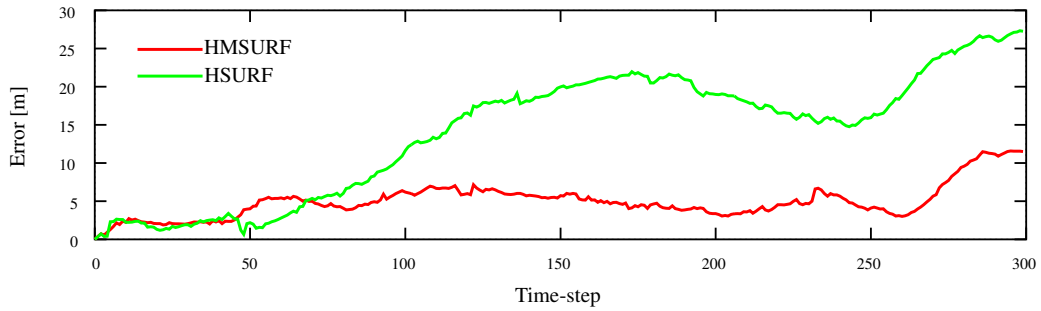


Figure 6.11: Pose estimation error of the vehicle estimation w.r.t. the GPS/INS information.

The implementation used to generate the results shown in this section is written in C/C++ language. This allows real time computation of the results. Currently, visual processing on the images of two consecutive time-steps and motion estimation are computed in about one second when 500 features are detected per image.

6.5 Conclusions

In this chapter, a platform-independent solution for egomotion systems has been presented. The used detector/descriptor technique based on good features to track, detecting either over original images (HSURF) or over moment images (HMSURF), and SURF descriptors provides reliable visual information, which allows adequate motion estimation for large sequences without need for filtering techniques or necessity of constraining the DOF of the mobile system. The quaternion method shows better performance results than SVD for 6 DOF applications, although the SVD computes similar result to the ones got by the former method in cases for 2D motion whether the number of DOF is reduced.

The fact that the proposed techniques work well in dynamic and unknown environments makes them suitable for a range of autonomous navigation systems even for long range trajectories including turnings and slop changes.

Chapter 7

Robust Egomotion by Nonlinear Optimisation

7.1 Overview

Egomotion solutions based on least squares, as done in chapter 6, are known to misbehave and present convergence issues when nonlinearities and unmodeled noises occur. This reasoning motivated us to move from the egomotion solutions presented in the previous Chapter to the nonlinear optimisation solutions shown hereinafter, for which higher levels of accuracy are possible [102, 103].

The family of solutions introduced present two distinctive traits that differentiate them from the other approaches presented before. The first important difference distinguishing these solutions from VSLAM solutions, common to the least square solutions presented in the previous chapter, is the independence with respect to the vehicle infrastructure. This means that whereas for VSLAM the inputs into the system have

to be known, in order to predict the system's state through the kinematic model, the optimisation solutions introduced here are capable of removing such an assumption. Secondly, the mathematical nature of the optimisation methods will allow refined solutions closer to the real behaviour of the system, which redound to alleviation of the nonlinearities influence.

This Chapter emphasises on developing a robust egomotion solution. The proposed improvements discussed here take place at two different levels: detection and motion estimation. Moment image representation has served us to upgrade detection performance, providing our design with more robust features. For the optimisation of motion estimation, we propose a *dual reprojection* strategy as opposed to *single reprojection* strategies, which allows a more stable and general solution.

In this chapter, a novel approach to the stereo egomotion problem is detailed. Constrained by the absence of GPS, we present a solely vision based motion estimation technique as opposed to INS-assisted solutions as purposed by Konolige *et al.* [104]. Detection of image features via Good Features To Track [63] enhanced with robust local description provided by SURF [28] is proposed. The robot motion estimation is computed via a Gauss-Newton (GN) *bundle adjustment* (BA) algorithm. The whole system is capable of estimating its own position without the addition of filtering strategies and it is demonstrated to behave accurately in real environments.

The Chapter is organised as follows: section 7.2 reviews the utilised visual detection and tracking schemes and explains the considerations taken into account for the current solution. Section 7.3 serves as an overview of the *bundle adjustment* technique employed on the motion estimation process. Results obtained using our implementation are shown and discussed in section 7.4. Following on from this, section 7.5 highlights main findings of the work.

7.2 Detection and tracking

Feature detection methods are important computer vision tools, useful to extract meaningful and lightweight information from images. They allow us to represent an image through a reduced set of characteristic points. Although this process entails inherent rejection of the available data, it is used as a base for most real time applications. The Good Features To Track proposed by J. Shi *et al.* respond to the need for the detected features to be feasibly trackable [63]. On the other hand Harris SURF (HSURF) and Harris Moments SURF (HMSURF) techniques have been shown in chapter 3, [59, 105], as successful combinations of the classic and fast feature detector from Harris *et al.* [60] and the robust local descriptors based on Haar wavelet responses from SURF [28]. Similar to this we propose a combination of Good Features To Track together with the robust local SURF descriptor as an appropriate detection scheme.

Here, where the camera system is composed of a stereo camera, the need for matching corresponding features takes place at two different levels. At the stereo pair level, each detected feature $\mathcal{F}_{L,k}^{(i)}$ of pixel coordinates $\boldsymbol{x}_{L,k}^{(i)}$ on the left camera at times k needs to be associated with its corresponding feature $\mathcal{F}_{R,k}^{(i)}$ of pixel coordinates $\boldsymbol{x}_{R,k}^{(i)}$ on the right side camera at times k for the later triangulation of its 3D position. Conversely, at the sequence level, each feature $\boldsymbol{x}_{L,k}^{(i)}$ is meant to be matched with its corresponding $\boldsymbol{x}_{L,k+1}^{(i)}$ at the next timestep $k+1$, in case it exists. In order to tackle both of these matching problems our solution utilises the local SURF descriptor.

Despite the need for retrieving 6 degrees of freedom (DoF) on ensuing motion estimation stages, we consider that the motion between two consecutive frames is roughly planar such that the upright version of the SURF method U-SURF is adequate for this study [28]. Nonetheless pyramidal scaling processes could be combined

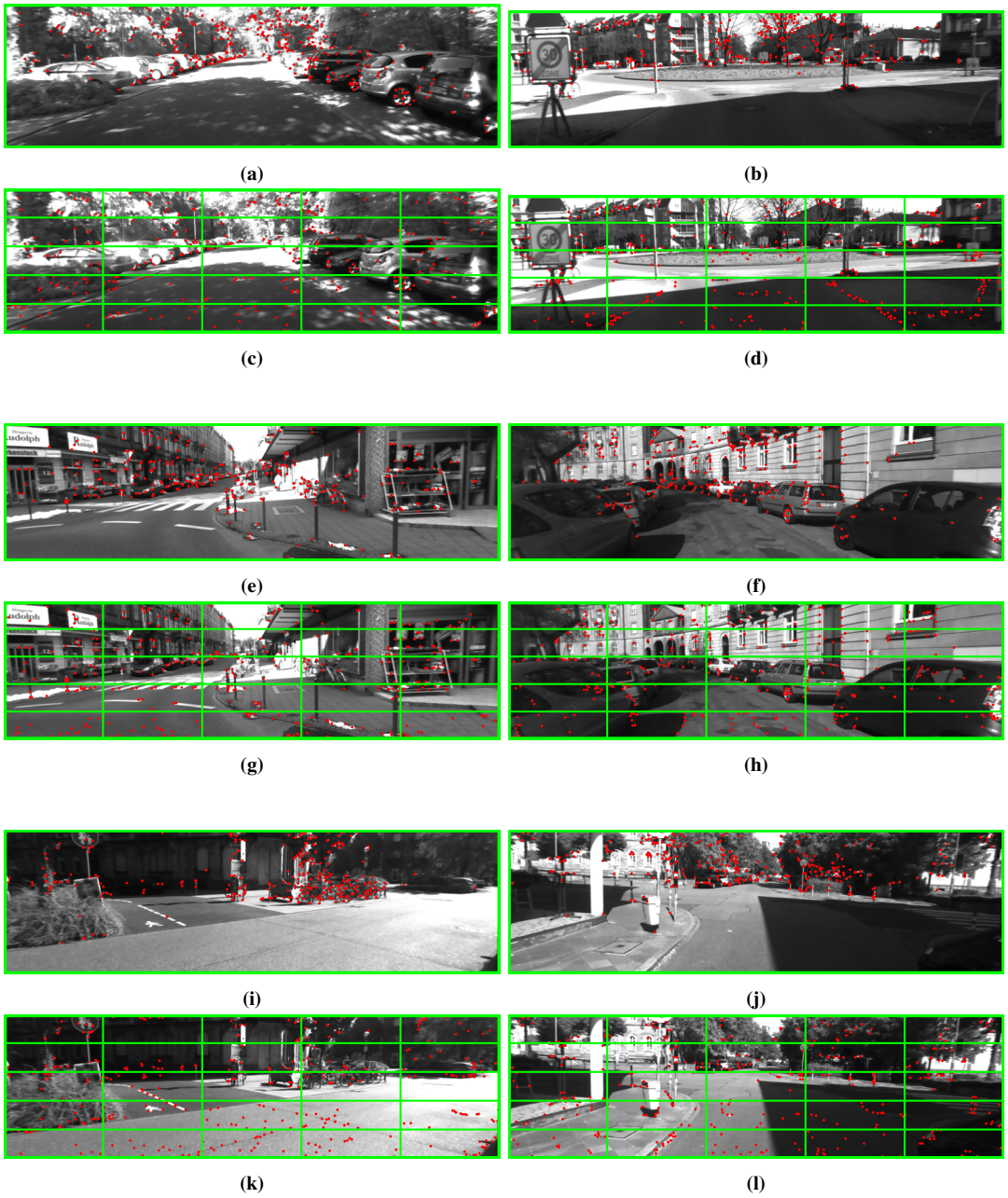


Figure 7.1: Images above show the feature detection (a) when 400 features are selected from the whole image and (b) when 16 features are detected from each of the 25 plotted subregions.

with the used detection algorithm as a means of preserving scale invariance of SURF. It has been seen that the solution is appropriate for our design conditions for either high rate image acquisition or slow moving platforms, where the scale change between two consecutive pairs of images is sufficiently small.

Subsequent outlier rejection steps conducted at *bundle adjustment* motion estimation permit our solution to skim over epipolar geometry constraint checks.

Because detection takes place at a small scale, where only the observed pixel and its neighbours are involved in the process, it is a common finding that some image areas are over populated with features while others remain empty. This is the case for images containing very spotted-like areas, where plenty of detected features are condensed. However, one way to alleviate this effect consists of increasing the minimum distance allowed for adjacent features. Alternatively, in our work images are evenly divided in a grid of subregions such that for each of them the same number of features is selected Figure 7.1.

7.3 Bundle adjustment

Bundle adjustment (BA) is an optimisation technique that is used for 3D reconstruction and other vision problems. Between the broad variety of strategies for BA, extensively detailed on the survey performed by Triggs *et al.*[106], the sliding window version is an appealing version for real time implementation and embedded systems. Instead of opting for a batched version of the algorithm, we decided on adapting a reduced version, where solely two adjacent pairs of stereo images from the sequence are used for motion estimation.

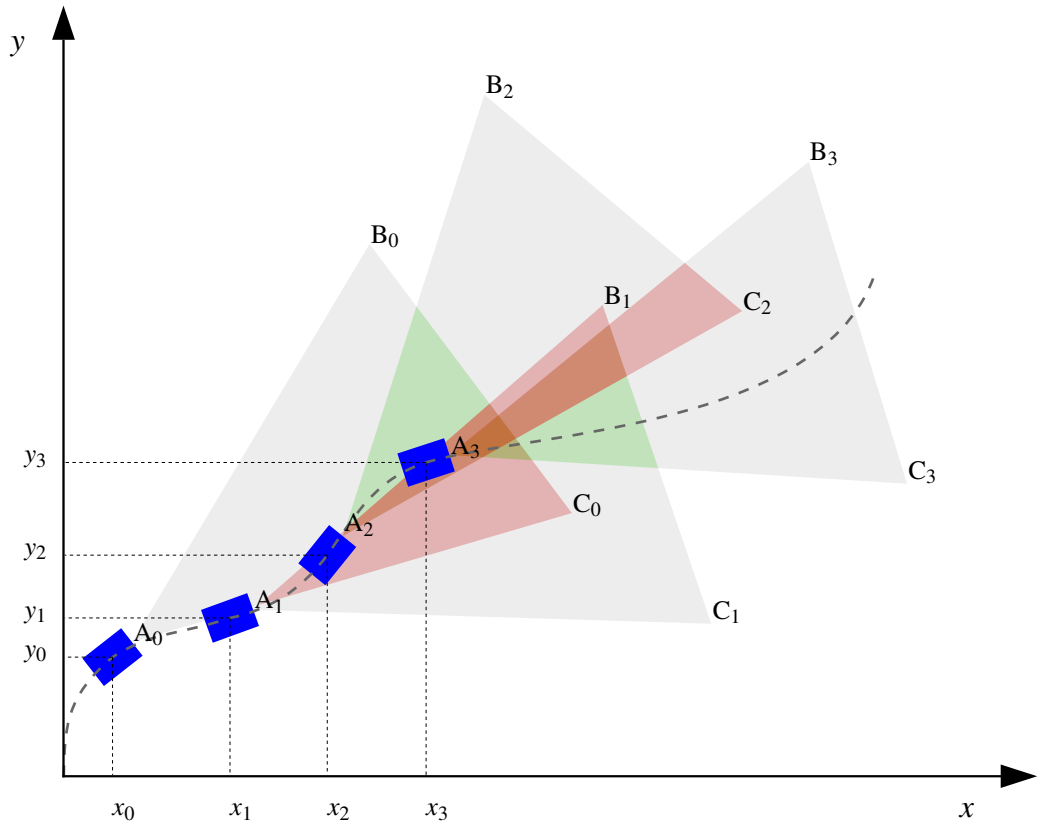


Figure 7.2: Field of view for a single stereo camera system along robot's trajectory.

The following sections review the formulae used for BA based on the Gauss-Newton optimisation technique, revisiting the models used for the camera and transformation representation.

Figure 7.2 displays how the field of view (grey) of the robot (blue) evolves as it moves. In the solution derived here, where optimisation takes place using only the observations from two consecutive poses, only the first order overlaps between consecutive fields of view (red) correspond to- the areas where the useful visual landmarks for the motion estimation will be contained.

7.3.1 Stereo camera projection function

Let us consider a stereo rig composed of two identical perspective cameras, where intrinsic and extrinsic calibration parameters are known (7.2). Then the projection of a 3D feature with coordinates $\mathbf{x} = (x, y, z)^T \in \mathbb{R}^3$, with respect to the left camera, which is considered as the reference camera, is computed as (7.1).

$$s \cdot \boldsymbol{x}_{H,L} = K_L \cdot \mathbf{x} \tag{7.1}$$

$$\boldsymbol{K} = \boldsymbol{K}_L = \boldsymbol{K}_R = \begin{pmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{7.2}$$

$$s \cdot \boldsymbol{x}_{H,R} = \boldsymbol{K}_R \cdot \mathbf{x}_R = \boldsymbol{K}_R \cdot \left(\mathbf{x} - (0, 0, Bl)^T\right) \tag{7.3}$$

where $\boldsymbol{x}_{H,L} = (u_L, v_L, 1) \in \mathbb{R}^3$ and $\boldsymbol{x}_{H,R} = (u_R, v_R, 1) \in \mathbb{R}^3$ are the projected coordinates of the 3D feature $\mathbf{x}$, expressed in homogeneous coordinates on the left and right image frames respectively. $\mathbf{x}_R$ represents the 3D coordinates of feature $\mathbf{x}$ on the right camera. $\boldsymbol{K} \in \mathbb{R}^{3\times3}$ is the matrix of intrinsic parameters that contains the focal lengths α_u, α_v, principal point coordinates u_0, v_0 and depth factor s. B_L is the stereo rig baseline. Note that the feature detection provides 2D features of coordinates $\mathbf{x}_L$ and $\mathbf{x}_R$ from which $\mathbf{x}$ is obtained via triangulation.

Then, under the assumption that the calibration parameters of the camera remain fixed, so that the BA does not have to recompute them again, it is convenient to define the reduced vector $\boldsymbol{y}$ of projected coordinates on the stereo set as the result of applying the projection function $\boldsymbol{f}$ to the 3D feature coordinates $\mathbf{x}$:

$$\boldsymbol{y} = \boldsymbol{f}(\mathbf{x}) = \left\{ \begin{array}{c} u_L \\ v_L \\ u_R \end{array} \right\} = \left\{ \begin{array}{c} \alpha_u \cdot \left(\frac{x}{z}\right) - u_0 \\ \alpha_v \cdot \left(\frac{y}{z}\right) - v_0 \\ \alpha_u \cdot \left(\frac{x - Bl}{z}\right) - u_0 \end{array} \right\} \tag{7.4}$$

Notice that the element v_R from x_R is not included in the vector $\boldsymbol{y} \in \mathbb{R}^3$ of projected coordinates. This is because it is identical to the coordinate v_L. Hence the application $\boldsymbol{f}$ corresponds to a nonlinear application $\boldsymbol{f} : \mathbb{R}^3 \to \mathbb{R}^3$.

7.3.2 Transformation model

The robot's motion is the result of a concatenation of translations and rotations represented by the parameters vector $\boldsymbol{p} = (\alpha, \beta, \gamma, t_x, t_y, t_z)^T \in \mathbb{R}^6$. This collects the 6DoF applied to the robot, α, β, γ are the applied rotations and $\boldsymbol{t} = (t_x, t_y, t_z)^T$ is the translation vector. These parameters allow the definition of the homogeneous transformation matrix $\boldsymbol{T}(\boldsymbol{p}) \in \mathbb{R}^{4\times4}$, which can be written as a composition of a pure translation $\boldsymbol{T}_{xyz}$ followed by three pure rotations, $\boldsymbol{R}_x(\alpha)$, $\boldsymbol{R}_y(\beta)$ and $\boldsymbol{R}_z(\gamma)$.

$$\boldsymbol{T}(\boldsymbol{p}) = \boldsymbol{T}_{xyz}(\boldsymbol{t}) \cdot \boldsymbol{R}_x(\alpha) \cdot \boldsymbol{R}_y(\beta) \cdot \boldsymbol{R}_z(\gamma) =$$

$$= \begin{pmatrix} C_\beta C_\gamma & -C_\beta S_\gamma & S_\beta & t_x \\ C_\alpha S_\gamma + C_\gamma S_\alpha S_\beta & C_\alpha C_\gamma - S_\alpha S_\beta S_\gamma & -C_\beta S_\alpha & t_y \\ S_\alpha S_\gamma - C_\alpha C_\gamma S_\beta & C_\gamma S_\alpha + C_\alpha S_\beta S_\gamma & C_\alpha C_\beta & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{7.5}$$

where $C_\theta = \cos(\theta)$ and $S_\theta = \sin(\theta)$ for any angle θ.

A transformation applied to a 3D feature, expressed in homogeneous coordinates $\mathbf{x}_H = (\mathbf{x}^T, 1)^T \in \mathbb{R}^4$ is:

$$\boldsymbol{T}(\boldsymbol{p}) \cdot \tilde{\mathbf{x}}_H = \mathbf{x}_H \tag{7.6}$$

where $\tilde{\mathbf{x}}_H$ are the homogeneous coordinates of the feature $\mathbf{x}_H$ after the motion.

7.3.3 Gauss-Newton

Errors in the estimate of distortion coefficients for camera lens and projective deformation effects are cause for visual data to behave nonlinearly. This makes Gauss-Newton a suitable method for *bundle adjustment* optimisation.

Consider minimisation of the nonlinear cost function:

$$S(\boldsymbol{p}) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{q} r_j\left(\boldsymbol{p}, \mathbf{x}^{(i)}\right)^2 \tag{7.7}$$

where r_j are the nonlinear residual functions dependent on the vector of parameters $\boldsymbol{p} \in \mathbb{R}^m$ and $\mathbf{x}^{(i)}$ are the system observations. The Gauss-Newton optimisation method states that the vector $\boldsymbol{p}$, the optimal solution for the cost function (7.7), can be iteratively computed by calculating the increment vector $\Delta\boldsymbol{p}$ as:

$$(\boldsymbol{J}^T \cdot \boldsymbol{J}) \cdot \Delta\boldsymbol{p} = -\boldsymbol{J}^T \cdot \boldsymbol{r} \tag{7.8}$$

where $\boldsymbol{r} = (r_1, r_2, \ldots, r_q)^T \in \mathbb{R}^q$ is the vector of residuals and $\boldsymbol{J} \in \mathbb{R}^{q \times m}$ is the Jacobian matrix of the vector of residuals with respect to the vector of parameters and $(\boldsymbol{J}^T \cdot \boldsymbol{J})$ is an approximation of the *Hessian* matrix.

Letting the vector of parameters $\boldsymbol{p}$ model the 6DoF motion through $\boldsymbol{T}(\boldsymbol{p})$ (7.5) and reprojecting the features $\mathbf{x}$ through the projection function $\boldsymbol{f}$ (7.4), different vectors of residuals can be constructed to define the cost function to be optimised.

7.3.4 Cost function: reprojection possibilities

The vector of residuals $\boldsymbol{r}$ has an important role in the minimisation process due to its influence on the Jacobian matrix. At the same time, the residual functions are responsible for capturing the representation of the transformation and the reprojection.

Consider the estimated transformation $\boldsymbol{T}_k(\hat{\boldsymbol{p}})$, representing the evolution a mobile agent from timesteps k to $(k+1)$, defined throught the estimation of the vector of parameters $\hat{\boldsymbol{p}}$. According to the convention used (7.6):

$$\boldsymbol{T}_k(\hat{\boldsymbol{p}}) \cdot \mathbf{x}_{H|k+1} = \hat{\mathbf{x}}_{H|k} \tag{7.9}$$

$$\boldsymbol{T}_k^{-1}(\hat{\boldsymbol{p}}) \cdot \mathbf{x}_{H|k} = \hat{\mathbf{x}}_{H|k+1} \tag{7.10}$$

where $\boldsymbol{T}_k^{-1}(\hat{\boldsymbol{p}})$ is the inverse of the transformation, $\boldsymbol{T}_k(\hat{\boldsymbol{p}})$, $\hat{\mathbf{x}}_{H|k}$ is the 3D estimate of the landmark from a later observation while $\hat{\mathbf{x}}_{H|k+1}$ is an estimate of a later position from a previous observation. These equations are the basis for the following reprojection strategies.

7.3.4.1 Forward-backward reprojection

Is the first of the two possible *single reprojection* strategies. We refer to *forward-backward reprojection*, or simply *forward*, as the process of reprojecting (later) observed features into previous camera frames. The vector of residuals is then computed from the estimated position of the feature $\mathbf{x}_{k+1}$ on the camera frame at timestep k, $\hat{\mathbf{x}}_k$. Equation (7.9) combined with (7.4) yields:

$$\hat{\boldsymbol{y}}_k = \boldsymbol{f}(\hat{\mathbf{x}}_k) = \boldsymbol{f}\left(T_k(\hat{\boldsymbol{p}}) \cdot \mathbf{x}_{H|k+1}\right) \tag{7.11}$$

where $\hat{\boldsymbol{y}}_k$ are the estimated coordinates of the feature on the prior camera frame.

Using *forward* reprojection the cost function to be minimised is calculated from the vector of residuals:

$$\boldsymbol{r}_{fw} = \boldsymbol{y}_k - \hat{\boldsymbol{y}}_k \tag{7.12}$$

where $\boldsymbol{r}_{fw} \in \mathbb{R}^3$ is the vector of residuals based on *forward* reprojection.

7.3.4.2 Backward-forward reprojection

We refer to *backward-forward reprojection*, or simply *backward*, as the process of reprojecting priorly observed features into later camera frames. The vector of residuals is then computed by reprojecting the estimation of the feature $\mathbf{x}_k$ on the camera frame at $(k+1)$, $\hat{\mathbf{x}}_k$. Equation (7.10) combined with (7.4) yields:

$$\hat{\boldsymbol{y}}_{k+1} = \boldsymbol{f}(\hat{\mathbf{x}}_{k+1}) = \boldsymbol{f}(T_k^{-1}(\hat{\boldsymbol{p}}) \cdot \mathbf{x}_{H|k}) \tag{7.13}$$

where $\hat{\boldsymbol{y}}_{k+1}$ are the estimated coordinates of the feature on the previous camera frame.

Using *backward* reprojection the cost function to be minimised is composed of the elements contained in the vector of residuals computed as follows:

$$\boldsymbol{r}_{bw} = \boldsymbol{y}_{k+1} - \hat{\boldsymbol{y}}_{k+1} \tag{7.14}$$

where $\boldsymbol{r}_{bw} \in \mathbb{R}^3$ is the vector of residuals used to compute the cost function based on *backward* reprojection.

7.3.4.3 Dual reprojection

Both of the shown possibilities for the computation of the vector of residuals, *forward* and *backward* reprojection, allow the computation of efficient cost functions for the BA optimisation algorithm. Nevertheless they are intrinsically making certain assumptions about the validity of feature positions. In the *forward* reprojection scheme, the feature coordinates on the previous frame y_k are used as reference and hence not reprojected. On the other hand, when the *backward* reprojection is used the feature coordinates y_{k+1} are the ones utilised as reference. These are reasons for both of the presented schemes to behave differently depending on the different nature of the feature error.

In order to attenuate the effect on the optimisation of unhandled errors due to non-reprojected feature coordinates used as reference, we present an approach based on a combination of both vectors of residuals:

$$\boldsymbol{r}_{dual} = \left(\boldsymbol{r}_{fw}^T, \boldsymbol{r}_{bw}^T\right)^T \tag{7.15}$$

where $\boldsymbol{r}_{dual} \in \mathbb{R}^6$ is the vector of residuals used for computation of the cost function on the *dual* reprojection scheme. The *dual* reprojection approach is in some way closer to the original cost function used for BA, in the sense that it discards less terms from the summation.

7.3.5 Outlier rejection

For a solution like the one proposed here, an outlier rejection scheme is of vital importance for improving accuracy. Although the matching process of SURF descriptors is robust and provides a number of good tracked features appropriate for motion estimation, there is still a group of tracked features that must be excluded in order to optimise egomotion results. This group of undesired features is generally composed of:

- False matches
- Matches on moving objects

The former subgroup contains wrong matches (matches of non-corresponding points) that SURF descriptors fail to distinguish and discard. Notice that although some of these matches could be removed applying epipolar constraints, some others will pass as true matches. The second group collects the good tracked features which actually belong to moving objects or entities on relative motion, e.g. self-shadow, pedestrians and cars on urban environment.

Figure 7.3 shows the outputs of the outlier rejection stage. The images have been extracted from the different sequences composing the datasets used for the experiments presented on the latter section and display different types of outliers, including outliers contained in moving vehicles, pedestrians and moving bicycles.

For these reasons, a RANSAC algorithm is used together with the GN-BA optimisation technique. Through a user defined threshold value ε, the summation of the residuals for each feature $\mathbf{x}^{(i)}$ evaluates as (7.16).

$$\left(\sum_{j=1}^{q} r_j\left(\boldsymbol{p}, \mathbf{x}^{(i)}\right)^2 \right) < \varepsilon \tag{7.16}$$

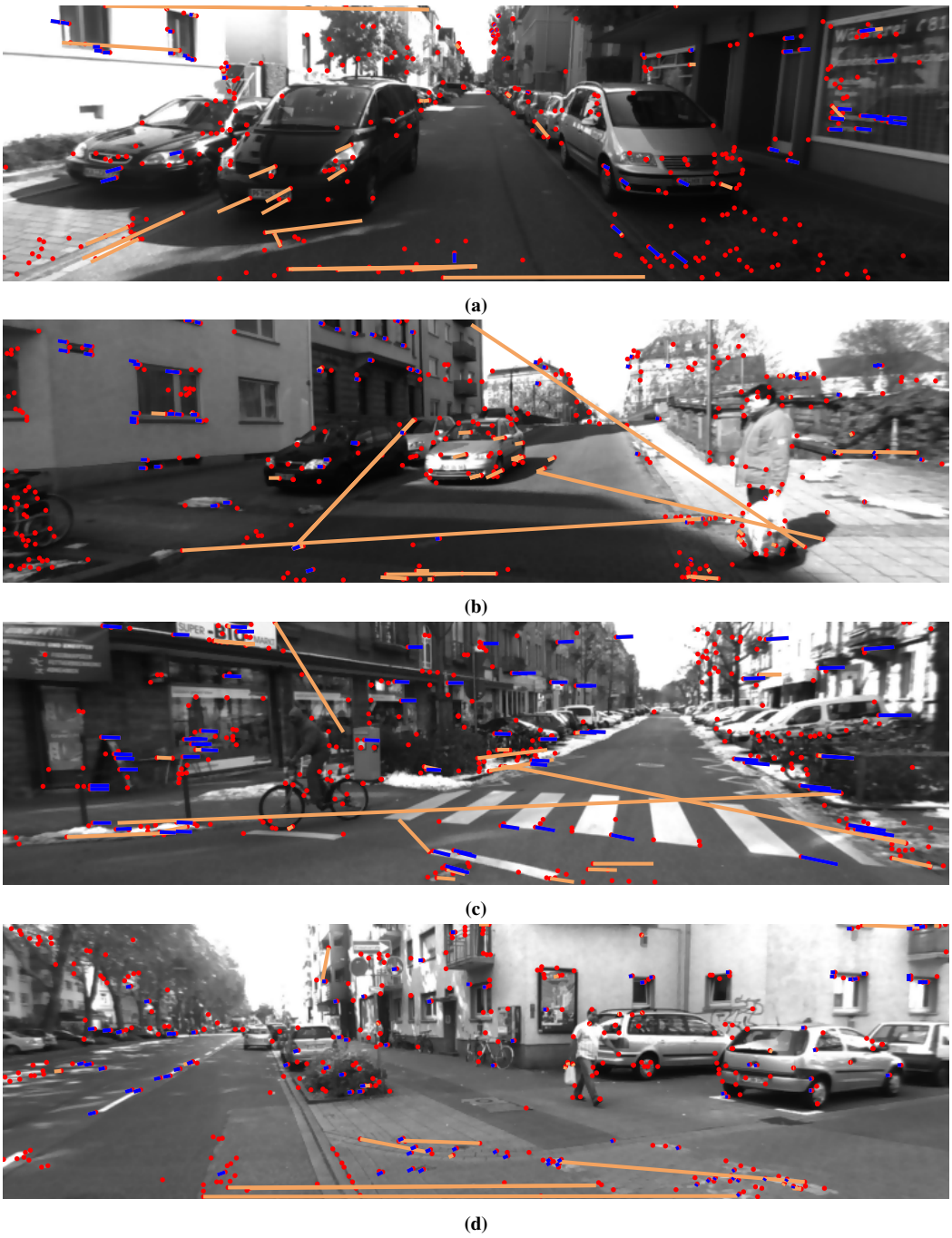


Figure 7.3: The outlier rejection process limits the valid reprojection error via the threshold ε, which allows discarding false matches and features contained on moving objects. Detected points (red), traces corresponding to accepted tracked features (blue) and traces corresponding to rejected outliers (orange) are shown. An outlier rejection threshold ε of value 4 was used here.

Other statistical techniques as Q test or the Peirce's criterion among other, and learning machines, specially unsupervised learning machines, can be employed to detect outliers, but these are computationally expensive and hard to implement. However, RANSAC's performance for data sets where the ratio of inliers with respect to the size of the sample is high make from this an advantageous solution for removing spurious data.

Features not within the constraints are rejected according to (7.16). However, there is a tradeoff for selecting the outlier threshold, since high values let wrong matches pass while smaller values can lead to rejection of valid matches. This means that the threshold ε has to be tuned to limit the outliers acceptance, while keeping a low ratio of inliers rejection. User-end visual assessment of the association rejection has been employed to choose a reasonable range for this threshold. We empirically found the threshold to provide a good performance results when $\varepsilon \in [2,8]$.

Using these outlier rejection settings we normally find that a percentage ranging form 75 to 95 of the association matches are identified as inliers in the motion estimation computation, whereas the rest of the association matches are discarded as outliers, for the tested datasets.

7.4 Experiment results

This section presents the results for motion estimation obtained using only the visual odometry techniques described in this chapter. The data we employed to test and validate our solution consists of a series of outdoor datasets collected from a vehicle travelling in urban environment in the city of Karlsruhe previously mentioned [90, 91].

Each dataset is composed of a set of high quality rectified stereo images.

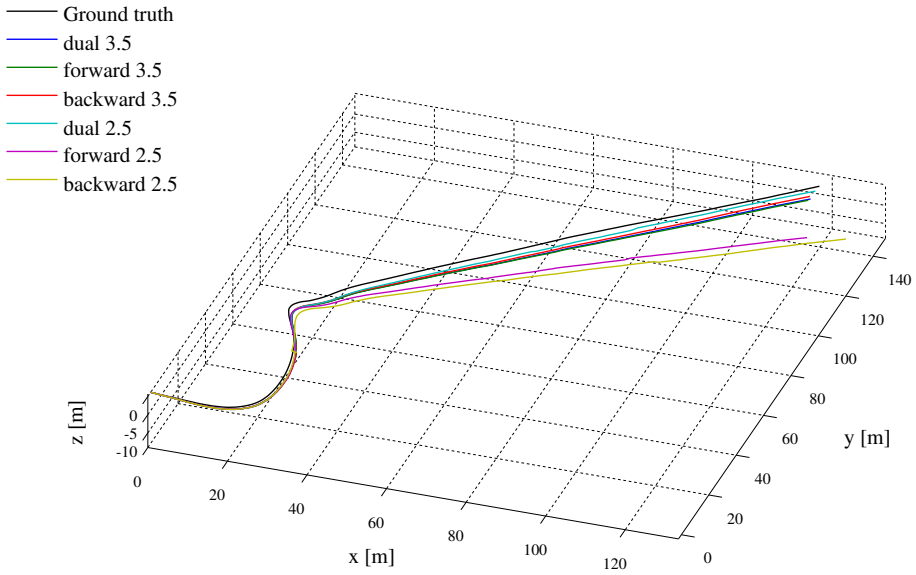
A total of 19 datasets summing a travelled distance of over 10km were put under the microscope, to generate these results. Previously explained *backward*, *forward* and *dual* reprojection strategies have been tested for accuracy and stability analysis.

For the conducted experiments 200 features were detected on each of the stereo pair images. Likewise, two different outlier rejection threshold values were used to generate the result series, $\varepsilon = \{2.5, 3.5\}$. These conditions allowed the implemented solution to converge in majority of the cases, finding only convergence issues when unhandled situations occur, e.g. moving objects occupying all the field of view. It was noted that on a number of cases the *dual* reprojection method demonstrated more stable behaviour presenting no local error issues (seen as error bumps on the figure) that the *forward* reprojection and *backward* reprojection strategies presented.

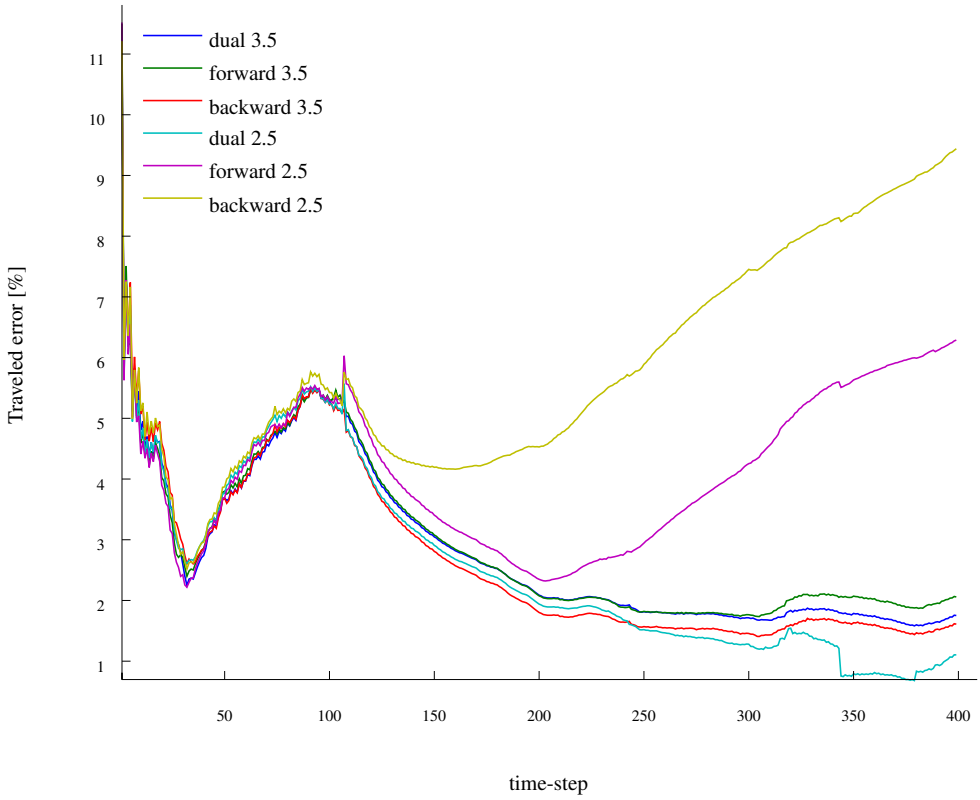
Achieved results are successful, reaching errors smaller than 1% and normally bounded by $5-10\%$ in terms of travelled error, defined as:

$$travelled\ error = 100 \cdot \frac{abs(error)}{travelled\ distance} \tag{7.17}$$

The implementation used to generate the results shown in this section is written in C/C++ language. This allows real time performance. Currently, visual processing on the images of two consecutive time-steps and motion estimation are computed in about one second when 500 features are detected per image.



(a)



(b)

Figure 7.4: Estimated trajectory (a) and travelled error (b) results for the dataset '2010_03_04_drive_0033', Karlsruhe Institute of Technology.

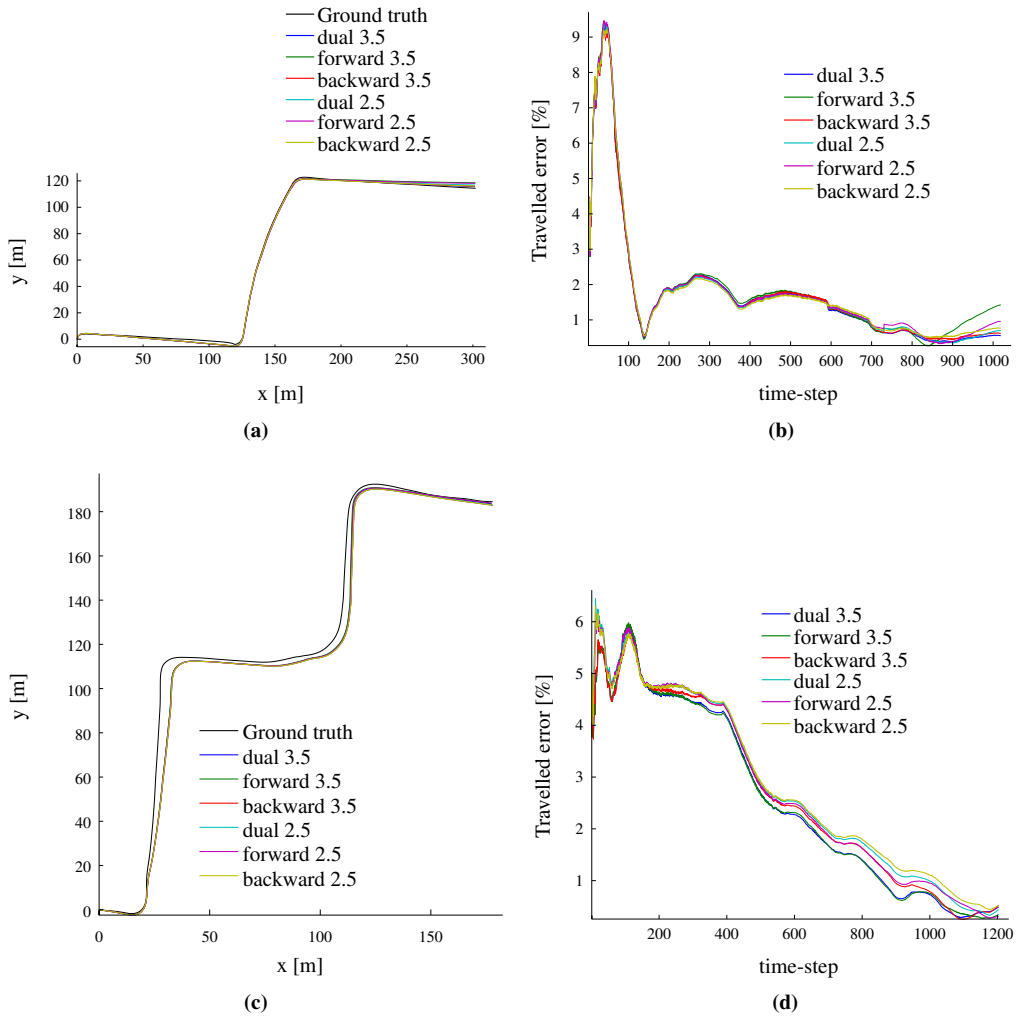


Figure 7.5: Trajectory estimation (a), (c) and errors (b), (d) for datasets '2009_09_08_drive_0015' and '2009_09_08_drive_0016'.

Figure 7.4, Figure 7.5 and Figure 7.6 show the estimation results obtained for some of the sequences contained in the dataset for the strategies and thresholds previously described. Figure 7.4 displays a clear example of how the dual reprojection strategy outperforms both the forward reprojection and the backward reprojection strategies. Indeed, it is seen from this figure, as from the other figures that the dual reprojection scheme always produces one of the two best achieved results. Thus representing the best trade-off solution.

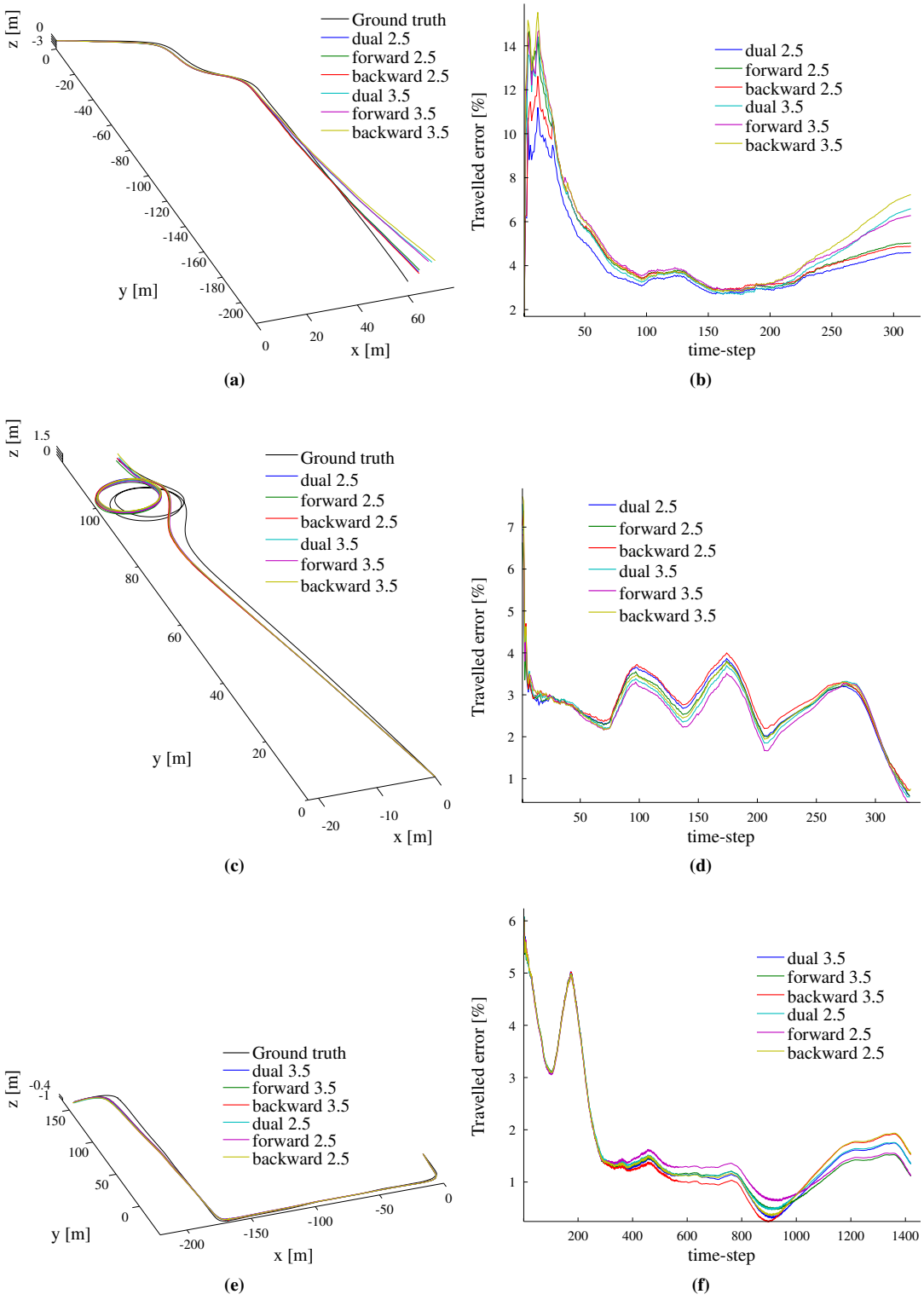


Figure 7.6: Trajectory estimation (a), (c), (e) and errors (b), (d), (f) for datasets '2010_03_05_drive_0023', '2010_03_09_drive_0081' and '2009_09_08_drive_0010'.

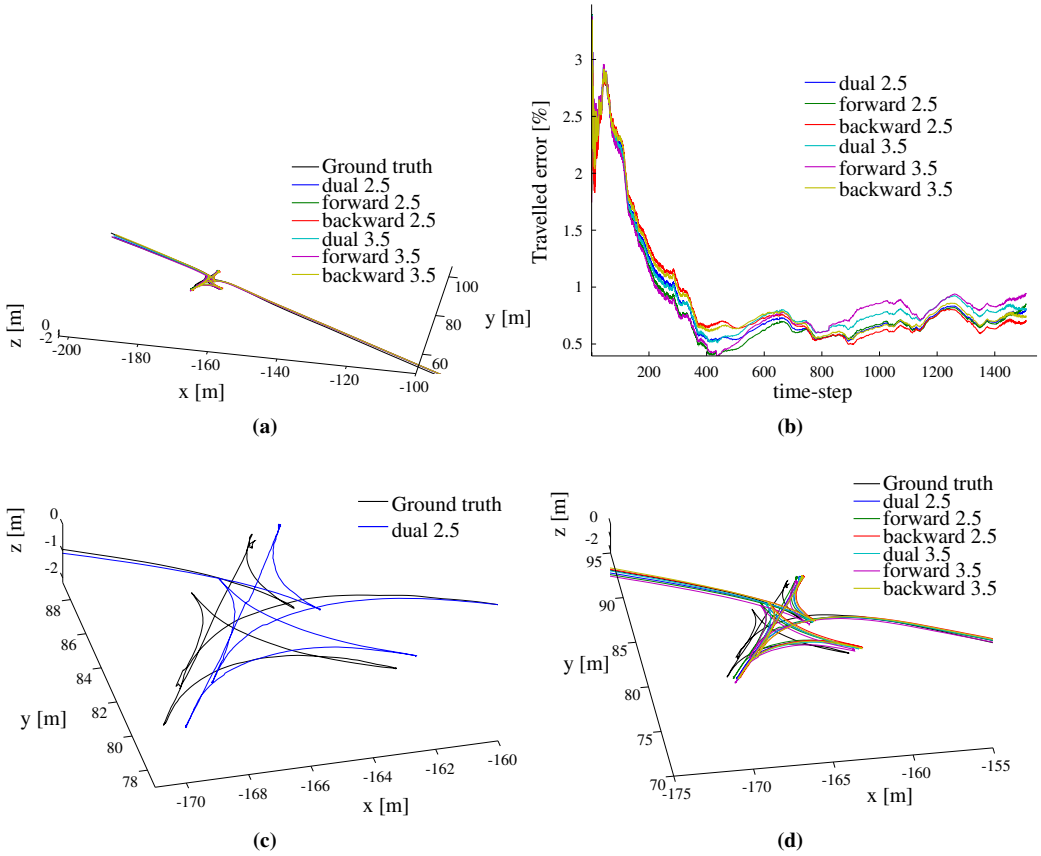


Figure 7.7: Trajectory estimation (a), (b), (c) and errors (d) for dataset '2009_09_08_drive_21'.

Figure 7.7 shows the trajectory estimation results and the travelled error obtained for the sequence of the dataset labeled as '2009_09_08_drive_21'. Figure 7.7c and Figure 7.7d correspond to a zoomed view of the most relevant portion of the sequence where a series of sharp turnings take place. Figure 7.7c plots only one series of results for the sake of clarity, while Figure 7.7d shows the results obtained for all the tested strategies. It is observed the ground truth, obtained using a highly precise GPS-assisted-INS system, presents some irregularities that do not appear on the trajectory estimates. This invites to reconsider the possibility of the estimations to be of higher quality than the ground truth itself, due to INS drift and internal filtering and GPS imprecision.

7.5 Conclusions

In this chapter, a platform-independent solution for egomotion systems has been presented. The used detector/descriptor technique based on Good Features To Track and SURF descriptors is robust against illumination changes via moment image representation. The reliability of the retrieved visual information allows adequate motion estimation for large sequences without need for filtering techniques or necessity of constraining the DoF of the mobile system.

An extensive experimental validation has been conducted. The analysis of the results shows that the most stable version for the cost function used for *bundle adjustment* is the one proposed as a *dual reprojection* scheme in majority of the cases. Conversely, accuracy is similar for all reprojection strategies, unless the cost function leads to error instability, which is avoided by the dual reprojection scheme.

The fact that the proposed techniques work satisfactorily in dynamic and previously unknown environments makes them suitable for a range of autonomous navigation systems, even for long range trajectories including turnings and slope changes.

Chapter 8

Multiple view based egomotion

8.1 Overview

Visual motion estimation is a very active field of research [24, 25]. Notwithstanding, there is a family of solutions that has been, from the author's point of view, undervalued and not appropriately taken into consideration. Namely the case for use of multiple stereo views.

Photogrammetry techniques, as bundle adjustment techniques, do not restrain the motion estimation problem to single stereo view approaches for mobile robot's motion estimation. However, solutions are mostly based on either mono-camera or single stereo camera techniques [91, 107, 108, 109].

Multiple stereo view sequences can be easily obtained from planetary rovers, either equipping them with several stereo cameras or just mounting a single stereo camera together with a pan and tilt mechanism as in [110, 111]. This latter addition allows the rover to augment its sensory capabilities, at a very low price.

The absence of multiple stereo view datasets in the literature, moved us to acquire our own experimental data. Edge technology systems have been used to generate accurate ground truth data together with the imagery data and other sensor's data. This allows rigorous validation of the results.

This Chapter presents two novel approaches to exploit the capabilities of multiple stereo view imagery. The first solution is based on optimisation techniques. It works generating a single estimation result from the multiple views data. The second solution is based on Covariance Intersection. Starting the motion estimates obtained for the single stereo view subsequences, this second solution leads with the problem of fusing those estimations together.

The Chapter is organised as follows: section 8.2 describes the data collection set up used to acquire the datasets presented for the validation of results. Section 8.4 explains the fundamentals used for motion estimation, when a multiple stereo view optimisation process is used. Section 8.5 explains the second data fusion method proposed to use the multiple stereo view data, by Covariance Intersection. Lastly, section 8.8 highlights main findings of the work.

8.2 Data acquisition and ground truth

The data collection strategy designed for the series of experiments presented here emphasises on the acquisition of not only a complete set of sensor's data from the onboard devices mounted on the rover (inertial measurement unit, wheel odometers and stereo cameras), but also on the acquisition of reliable ground truth data, suitable for high accuracy validation purposes.

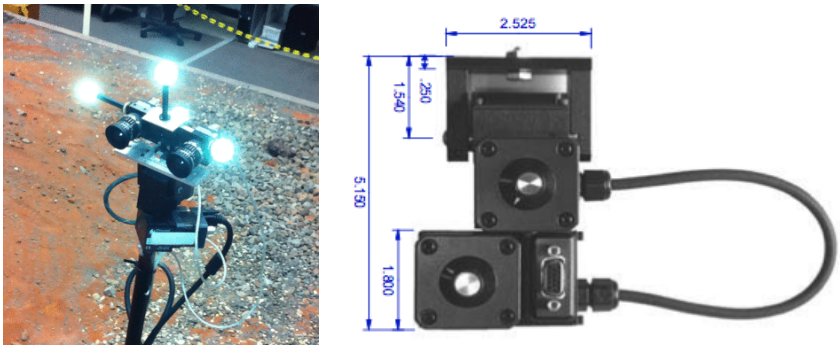
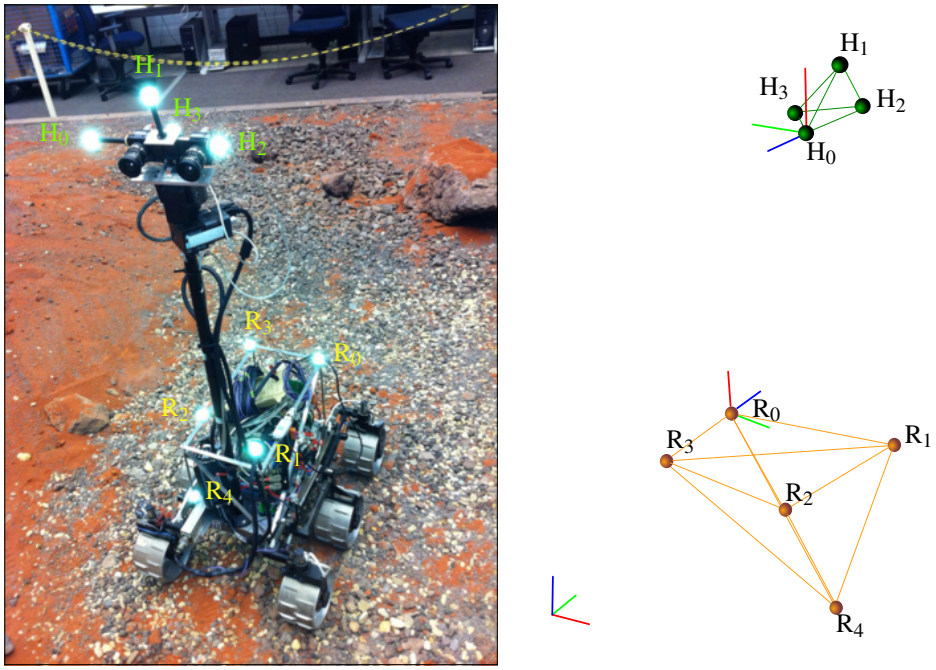


Figure 8.1: Pan and Tilt Unit, ExoMader.

This reliable reference data is provided by indoor Vicon MX Positioning System composed of eight to eleven, depending on the dataset, Infra Red (IR) cameras [112].These IR cameras, acquire images that the Vicon system's core processes to retrieve the 3D location of a series of markers placed on the robot and its camera header. Such markers are spheric pieces whose surface is highly reflective on the IR spectrum, for easy detection. In this manner the markers are tracked at frame rates of up to 100 Frames Per Second (FPS) with high levels of accuracy.

The Vicon Nexus software allows the creation of 3D user-defined models that are defined by grouping several markers. The interaction between the markers of a model can also be defined, so that the tracking process does not only take place at a discrete 3D point level, where each marker behaves as an independent entity, but also as solid rigid structures. This tracking mode allows attitude tracking as well as global positioning. These capabilities make the Vicon system a powerful solution to track objects moving within the 3D control volume in real time and a useful tool for generating precise reference data [112, 113].



(a) (b)

Figure 8.2: Experimental setup in the ExoMaDeR rover (a) and positioning system model (b).

Given that in our experimental setup the camera header sits over a pan-tilt unit (PTU) mounted on the top of a mast of 75 [cm] of height, Figure 8.1, it seems reasonable defining two different solid bodies to be tracked independently. In this manner pan and tilt rotations of the camera rig are also available data stored with the ground truth information, as opposed to storing the robot pose only. Figure 8.2 shows a sketch and photo of the used setup. In order for the odometry and imagery timeframe to be consistent with the reference information, positioning data is wirelessly retrieved online from the rover platform at 20 FPS (Frames Per Second), Figure 8.5.

Imagery is the main piece of information acquired from the robot sensing systems for our experimental purposes. The robot is equipped with a stereo camera composed of two identical uEye USB cameras mounted on a rigid frame. The acquired images are taken on a resolution of 1280 $\times$ 1024 [pixels]. In order for multiple stereo

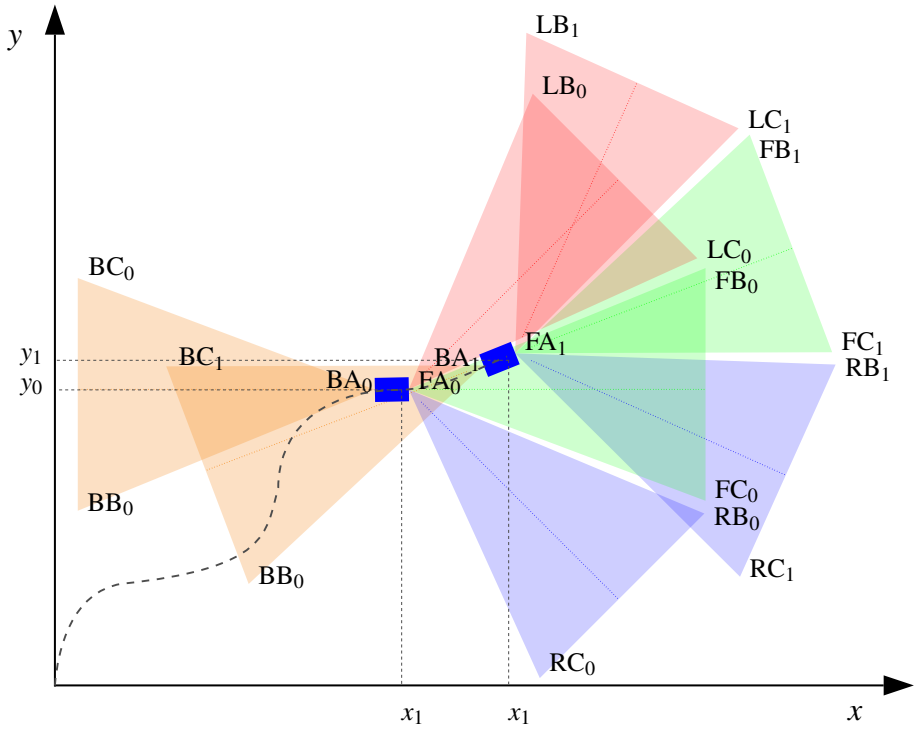
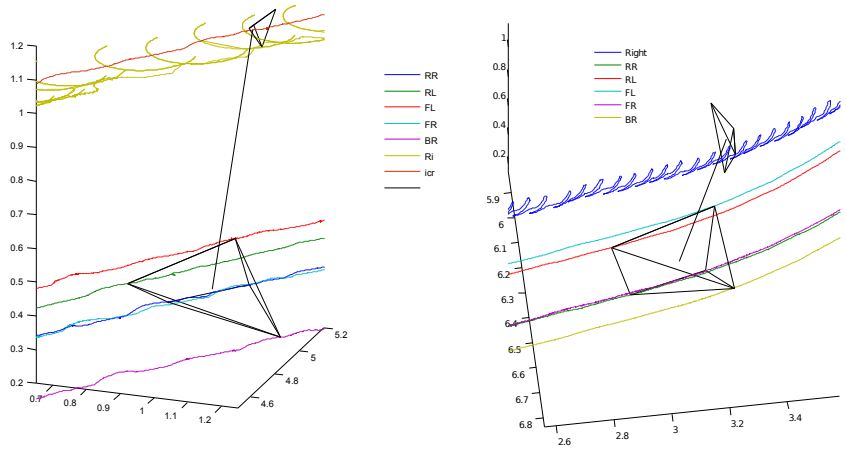


Figure 8.3: Fields of view on a multiple view and multiple camera setups.

views images to be taken taken, the PTU is actioned moving the stereo rig to each view position. Images are taken at evenly distributed distances along the trajectory, for poses spaced 15 [cm] at most. Figure 8.6 and Figure 8.7 display some image samples of the left images of the stereo pairs for datasets where three and four different views are acquired.

There are two acquisition scenarios that can be distinguished when multiple stereo view images are collected. In a first scenario, all the stereo images for the different views are acquired while the robot remains on the same location and attitude. When this happens the fields of view of each stereo view evolve as shown in Figure 8.3. This schematic example shows the fields of view of four different views (or cameras) for two different poses 0 and 1. The first letter denominates the view, (L)eft, (F)ront,



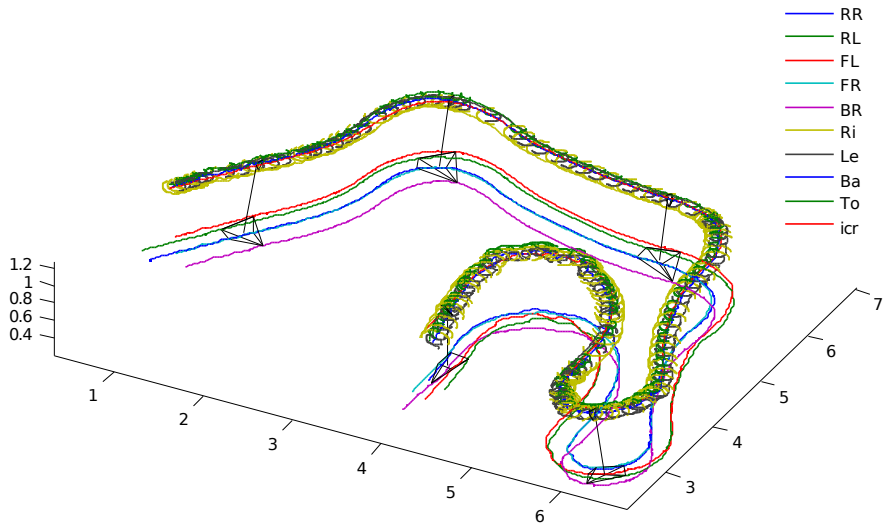
(a) (b)

Figure 8.4: Reference data: detail on markers evolution.

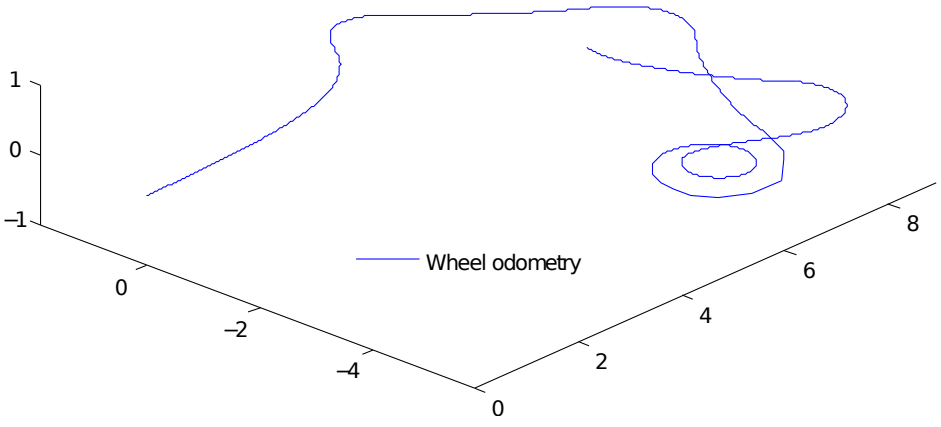
(R)ight and (B)ack, the second letter $\{A,B,C\}$ labels the vertices of the field of view and the subscripted number references the pose. In this scenario, the markers on the camera header will move describing the arc of a circumference, unless contained in the instant centre of rotations of the camera header, Figure 8.4a. A second scenario occurs when the rover moves while the images are being acquired. In this case the PTU is actioned while the robot moves, so that the markers in the camera header evolve as in Figure 8.4b. Note that in this scenario each view is acquired from a different pose.

A positive tilt angle was applied to the camera rig, for all the datasets, so that the cameras could face areas closet to the robot.

The dataset generation presented here contributes to the construction of reliable datasets for indoor experiments. Furthermore, it fulfils the need for generating datasets containing multiple stereo view image sequences. Finally, it is important noting that the rover travelling conditions on the Planetary Test Best are comparable to rough terrain navigation conditions where planar motion cannot be assumed.



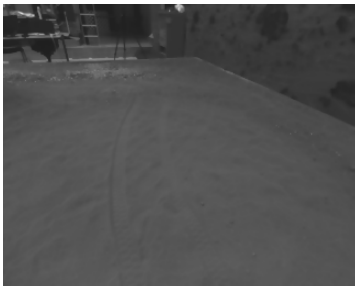
(a) Reference Data. Trajectory of all markers on a dataset.



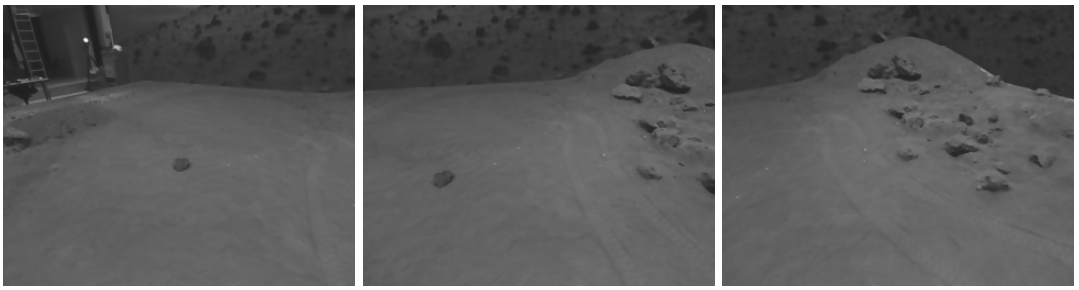
(b) Wheel Odometry

Figure 8.5: Experimental datasets.

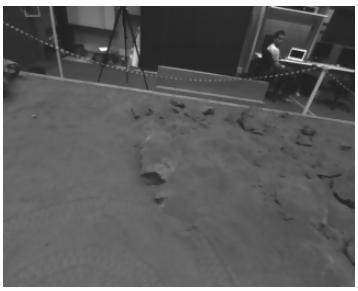
Left view, ψ=30 [deg] Front view, ψ=0 [deg] Right view, ψ=-30 [deg]



Back view, ψ=180 [deg]



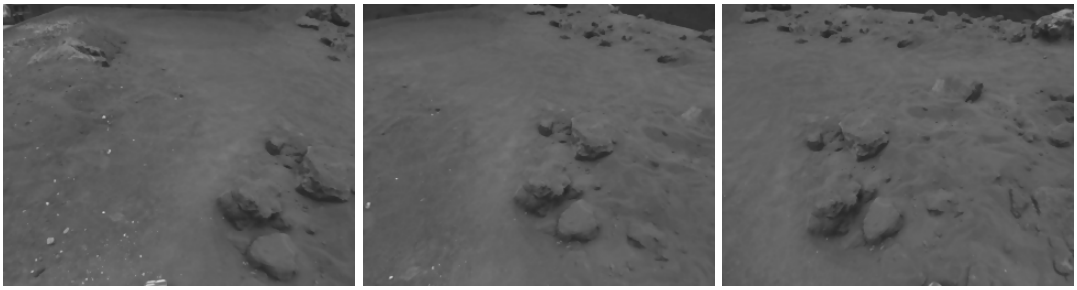
Left view, ψ=30 [deg] Front view, ψ=0 [deg] Right view, ψ=-30 [deg]



Back view, ψ=180 [deg]

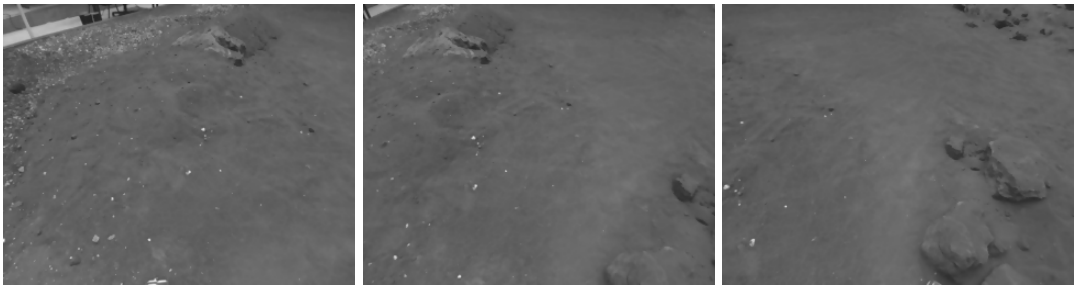
Figure 8.6: Left images from the stereo pairs taken at different pan settings whilst in a common pose. Tilt angle for all displayed images is -28.286 [deg]

Left view, ψ=30 [deg] Front view, ψ=0 [deg] Right view, ψ=-30 [deg]



Left view, ψ=30 [deg] Front view, ψ=0 [deg] Right view, ψ=-30 [deg]

Left view, ψ=30 [deg] Front view, ψ=0 [deg] Right view, ψ=-30 [deg]



Left view, ψ=30 [deg] Front view, ψ=0 [deg] Right view, ψ=-30 [deg]

Figure 8.7: Left images from the stereo pairs taken at different pan settings whilst robot remains in the same pose. Tilt angle for all displayed images is -43.714 [deg]

8.3 Visual scheme

The egomotion approach developed here is uniquely based on calibrated and already rectified input images. Therefore the importance for the image processing module to extract reliable and meaningful information from the perceived scene is crucial.

A reduced detection solution, based on the Good Features To Track from J. Shi *et al* [60, 63], has shown to cope with the responsibility of providing stable and robust features against illumination changes when detection is conducted over moment images [59, 68, 70]. After detection is performed on the images of the stereo pairs, stereo matching and tracking are required steps. In order to solve both of the problems at a time, feature descriptors are normally used at this stage. Based on previous successful experiences, local SURF descriptors are used here [28, 105].

Note that the multiple view system hereby analysed is only composed by a single stereo camera rig, which attached to a pan and tilt unit, allows the rover to acquire images whilst oriented at different attitudes.

8.4 Multiple stereo view motion estimation

Optimisation techniques, and more specifically non-linear optimisation techniques, are usually employed to solve the motion estimation problem.

Among the wide range of alternatives in which solutions like bundle adjustment can be developed, we have considered the study of a Gauss Newton based approach similar to the one proposed in the previous chapter. Even though batch versions of the algorithm can improve the accuracy of the solution, they also entail additional computational expenses. For this reason, the version of bundle adjustment algorithm

derived here consists on the sliding window formulation for a system where solely the two adjacent time-step images are used for the estimation.

The following section review the used formulae for the implemented BA based on Gauss-Newton, revisiting the models used for the camera and transformation representation and also the cases of single and multiple camera systems.

8.4.1 Multiple stereo views optimisation

While the projection model for the camera remains modestly simple in the case of single camera systems where both intrinsic and extrinsic parameters can be expressed as coupled terms, via (7.1) and (7.4), without need for these sets of parameters to be independently known, cases as the one presented for the mobile robot here where the stereo rig is mounted over a pan and tilt unit (PTU), the knowledge of the extrinsic parameters is rather necessary not only for clarity purposes, but also to develop the full kinematic model of the system.

In our specific setup, where there is only one stereo camera set, the calibration of the system reduces to the estimation of the intrinsic calibration for the stereo set plus an extrinsic calibration to relate the camera position with respect to the pan and tilt unit. An additional set of extrinsic parameters relating the position of the PTU with respect to the robot body can be also modelled. This latest set of extrinsic parameters is necessary when measurements from every sensor are required to be expressed in a common reference frame for fusion purposes.

The kinematic model that represents the transformations from any stereo camera view to the upright position of the camera in the robot, where no roll β, tilt α and pan γ are applied, depends only on the pan and tilt (variable parameters) and the geometry of

the pan and tilt stereo system. Then the extrinsic transformation $\boldsymbol{T}^{(}\boldsymbol{p}^{(k)})$ for the stereo view (k) is computed as:

$$\boldsymbol{T}(\boldsymbol{p}) = \boldsymbol{R}_z(\gamma) \cdot \boldsymbol{R}_x(\alpha) \cdot \boldsymbol{T}_{xyz}(\boldsymbol{t}_1) \cdot \boldsymbol{R}_y(\beta) \cdot \boldsymbol{T}_{xyz}(\boldsymbol{t}_2) \tag{8.1}$$

where:

- $\boldsymbol{R}_x(\alpha), \boldsymbol{R}_y(\beta), \boldsymbol{R}_z(\gamma)$ are the transformations defined by the pure rotations due to tilt, roll (fixed) and pan respectively.
- $\boldsymbol{T}_{xyz}(\boldsymbol{t}_1)$ is a transformation defining the pure translation that represents the position of the centre of the stereo camera with respect to the Instant Centre of Rotation where pan and tilt are applied.
- $\boldsymbol{T}_{xyz}(\boldsymbol{t}_2)$ is a transformation defining the pure translation that represents the position of the left camera with respect to the centre of stereo camera.

When a BA technique is employed to compute the mobile platform motion for the case when a single stereo camera is on-board, the cloud of 3D points that is reconstructed from the stereo images and then inputed into the system can relate to any reference frame so that the motion is expressed in such a reference frame. Conversely, when multiple stereo views are utilised on BA to compute the motion, all the input 3D points have to be expressed in a common reference frame in order be fused to compute the common and unique motion that takes place in the robot. The selection of a reference frame to express the observations acquired from a unique on-board sensor can be trivially chosen. This does not mean that any reference frame can be convenient, neither that any reference frame will lead to the same results on later processing stages, but only that there is a wide range of possibilities, for instance the case of a

local reference frame attached to the device. One of the reference frames in which the 3D coordinates are most naturally and intuitively represented for stereo cameras is normally placed on the left camera of the pair. Nonetheless, the left camera frame is a convenient frame to represent points in terms of formulation, because it reduces the complexity associated to a robot centred representation where extrinsic models are required, it is also insufficient when information gathered from different sensors is intended to be fused.

A case as the one presented in this work, where a stereo camera changes its orientation actioned by a PTU can also be conceptualised as a system equipped with several cameras. In this manner, each camera configuration (or equivalently each view) is considered as a different stereo camera that relates to the main system through its own set of extrinsic parameters. We can rewrite the cost function to minimise as follows:

$$S(\boldsymbol{p}) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\sum_{j=1}^{q} r_j^{(k)}\left(\boldsymbol{p}, \mathbf{x}_k^{(i)}\right)^2 \tag{8.2}$$

where K represents the number of cameras (or number of views) utilised to compute the estimation, n_k represents the number of points that where detected and associated for two consecutive poses in the stereo camera (view) k, and $r_j^{(k)}$ are the $j = 1, 2, \ldots q$ residual functions.

Let us rewrite (7.4) here, where the reduced vector of projected coordinates is defined, (8.3).

$$\boldsymbol{y} = \boldsymbol{f}(\mathbf{x}) = \begin{pmatrix} u_L & v_L & u_R \end{pmatrix}^T \tag{8.3}$$

The vectors of residuals $\boldsymbol{r}^{(k)}$ depend on the corresponding projection functions $\boldsymbol{f}^{(k)}$, for each of the cameras, that embedds both the intrinsic and extrinsic parameters.

$$\boldsymbol{y}_k = \boldsymbol{f}^{(k)}(\mathbf{x}_k) = \boldsymbol{f}\left(\boldsymbol{T}^{-1}(\boldsymbol{p}^{(k)}) \cdot \mathbf{x}_k\right) \tag{8.4}$$

$$\boldsymbol{r}^{(k)} = \boldsymbol{y}_k - \hat{\boldsymbol{y}}_k \tag{8.5}$$

where $\boldsymbol{y}_k$ corresponds to the reduced vector of projected coordinates in the view k, $\hat{\boldsymbol{y}}_k$ are estimates, and the vectors of parameters $\boldsymbol{p}^{(k)}$ are used to represent the extrinsic transformation from the camera k to a common frame. In this manner, every feature observed from any of the different views is represented in the same reference frame prior to the computation of motion estimates. It is important emphasising that the different vectors of residuals can be constructed to define the cost function to be optimised, as previously explained in section 7.3.4 [114].

Note that the cost function used here is not as general as the proposed on the bundle adjustment complete formulation, but conversely it only takes into account the evolution of points observed always from the same cameras. In this manner points that at time-step t where, for instance, on the FOV of the front camera and at $t+1$ appear on one of the side views are not considered by the cost function (8.2).

Figure 8.3 shows the fields of view of four different camera view of a system like the one used for this work. In this graphic example, overlap between front view (green) and side views (red, blue) is not shown for the sake of clarity. However, the pan angles used on our experiments produce overlap between front and side views.

The implementation used to compute multiple stereo view egomotion results is written in C/C++ language. This allows calculation of motion estimates every four second when 500 features are detected per image. In a rover's mission, where images are obtained at 0.2 [Hz], this can be considered real time computation.

8.5 Fusing multiple stereo views: A filtering approach

Another data fusion alternative for mobile robot systems equipped with several sensors can be developed using filtering techniques. Kalman Filter (KF), or more generally Extended Kalman Filter (EKF), techniques are commonly used to integrate the information acquired from different sensors as it is the case for GPS/INS systems [115, 116, 117].

However, our solution points to another different direction. Instead of fusing raw visual data by means of filtering techniques, our approach focuses on fusing several estimations into a common one. This is, given a multiple stereo view sequence of images it is possible to compute several the motion estimations, one per stereo view. For instance, for the case of the robot obtaining stereo images at two different views (left view, where the pan angle is such that the camera header faces left, and right view), two estimations of the robot's motion can be calculated using only the subset of images that correspond to each side. Nonetheless, there is only one robot, so that a unique estimation can be computed as a result of combining the different estimates.

8.5.1 Covariance intersection for fusing estimations

Covariance intersection (CI) is a commonly used technique to fuse estimations [118, 119]. Let us consider $\boldsymbol{a}$ and $\boldsymbol{b}$ be two estimations of the vector of parameters $\boldsymbol{p}$. Then, the CI methods allows the computation of $\boldsymbol{c}$, as a combination of $\boldsymbol{a}$ and $\boldsymbol{b}$, as follows:

$$\boldsymbol{c} = \boldsymbol{C} \cdot \left[\omega \cdot \boldsymbol{A}^{-1} \cdot \boldsymbol{a} + (1-\omega) \cdot \boldsymbol{B}^{-1} \cdot \boldsymbol{b} \right] \tag{8.6}$$

$$\boldsymbol{C}^{-1} = \omega \cdot \boldsymbol{A}^{-1} + (1-\omega) \cdot \boldsymbol{B}^{-1} \tag{8.7}$$

where the free parameter ω is the weighting coefficient that leads to the minimisation of either the determinant or the trace of the covariance matrix $\boldsymbol{C}$, associated to the estimate $\boldsymbol{c}$. Matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are the covariance matrices associated to the estimates $\boldsymbol{a}$ and $\boldsymbol{b}$ respectively [120].

Then, a Kalman filter scheme is added to model the evolution of the parameters that estimate the robot motion, so that the acceleration of the parameters is supposed to be constant [121]:

$$\begin{Bmatrix} \boldsymbol{v}_{t+1} \\ \boldsymbol{a}_{t+1} \end{Bmatrix} = \begin{bmatrix} \boldsymbol{I}_6 & \boldsymbol{I}_6 \cdot \Delta T \\ 0_6 & \boldsymbol{I}_6 \end{bmatrix} \cdot \begin{Bmatrix} \boldsymbol{v}_t \\ \boldsymbol{a}_t \end{Bmatrix} + \mathbf{w}_t \tag{8.8}$$

$$\boldsymbol{p}_{t+1} = \begin{bmatrix} \boldsymbol{I}_6 \cdot \Delta T & 0_6 \end{bmatrix} \cdot \begin{Bmatrix} \boldsymbol{v}_t \\ \boldsymbol{a}_t \end{Bmatrix} + \mathbf{v}_t \tag{8.9}$$

where $\boldsymbol{v}_{t+1}$ represents the velocity of the parameters $\boldsymbol{p}_{t+1}$, computed dividing this by the increment of time ΔT, and $\boldsymbol{a}_{t+1}$ is the acceleration of the parameters. Matrices $\boldsymbol{I}_6$ and 0_6 are the identity and zero matrices of size six. The process noise and the measurements noise are $\mathbf{w}_t$ and $\mathbf{v}_t$ respectively.

In this manner, vectors of parameters $\boldsymbol{p}_t^{(k)}$ are computed for each stereo view k according to the KF defined in (8.8) and (8.9). Then, equations (8.6) and (8.7) are applied sequentially for as many estimates as available views the system is configured to acquire. This means that the CI method is used $K-1$ times when K views are fused.

8.6 Experiment results

This section shows the robot egomotion results obtained by the data fusion methods presented in this chapter. The datasets utilised here were acquired in the Planetary Utilisation Testbed at the European Space Technology Center (ESTEC) of the European Space Agency (ESA).

Using a validation approach of a similar nature to the one presented by [122], where 6 points along the trajectory are taken as reference to compare the similarity between the reference data and the pose estimates. However, for this validation we opted for using as much ground truth data as we had available, this is up to 70k samples per dataset. In order to do this the estimated data is upsampled interpolating the missing estimation values using the timing data available for both image acquisition and positioning data. Nevertheless, this interpolation process produces results with low estimation errors.

Figure 8.9a shows the trajectory estimates obtained with a multiple stereo view approach and the estimates obtained using a single stereo view approach, superimposed with the ground truth data. Absolute 3D errors between these series and reference data are shown in Figure 8.9b. Figure 8.9c, corresponds to the averaged of travelled error, computed as the percentage obtained from the ratio between the average of the absolute 3D error and the total travelled distance.

It is seen from the results how a multiple stereo view solution is able to accurately estimate the robot evolution for a sequence corresponding to 20 [m] of travelled distance, where images taken at a total of 56 different poses are used. The figures show how the multiple camera solution provides convergent results while the single camera solution fails in at least two occasions along the trajectory.

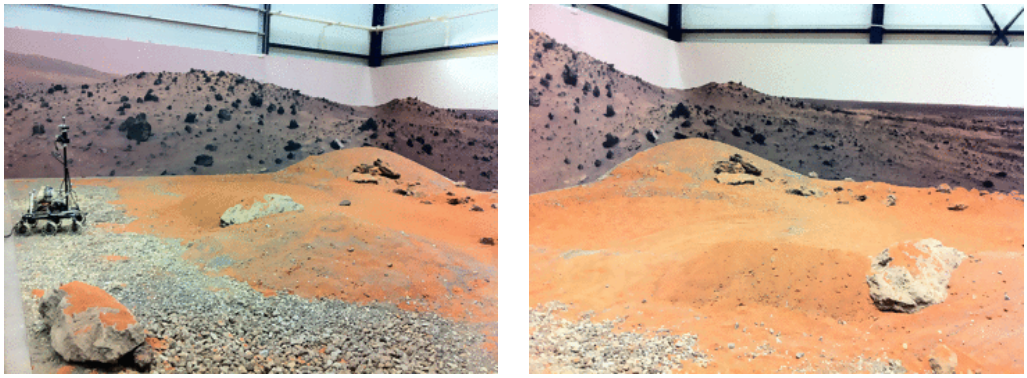


Figure 8.8: Planetary Test-bed Unit, ESTEC, The Netherlands.

It can be observed from Figure 8.9c that the averaged travelled error decreases monotonically along the trajectory reaching a final value of 0.38%. This decreasing behaviour derived from the fact that the error remains contained within the same range whereas the travelled distance increases.

Figure 8.10 shows similar results for a sequence composed of three views were images are taken at a higher frame rate. As it can be seen from these figures, the estimation computed using only the Front stereo view images presents convergence issues. Note that the single stereo view estimations are generated using sets of 200 points, whereas the estimation based on multiple stereo view optimisation is computed using only 50 points from each of the three views. This means that even for a smaller amount of input data, the multiple stereo estimation can provide more accurate results that also guarantee a better convergence.

Figure 8.11 presents the estimation results obtained for single stereo view optimisation and also the multiple stereo view estimation obtained from fusing the three solutions obtained from each of the different views. It is seen from the figures how the error of the fused estimation is smaller than estimation errors obtained for each of the estimations independently.

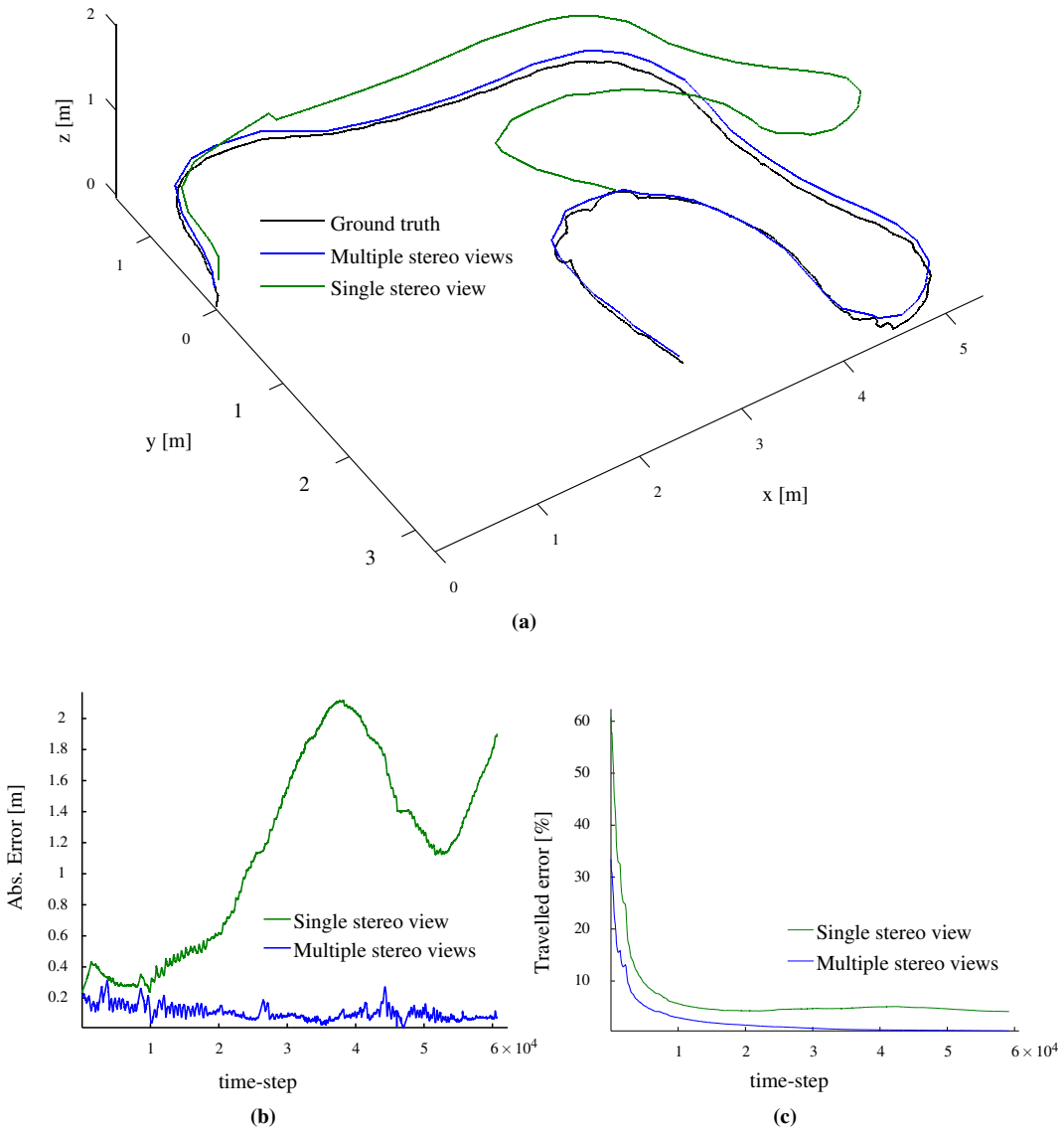
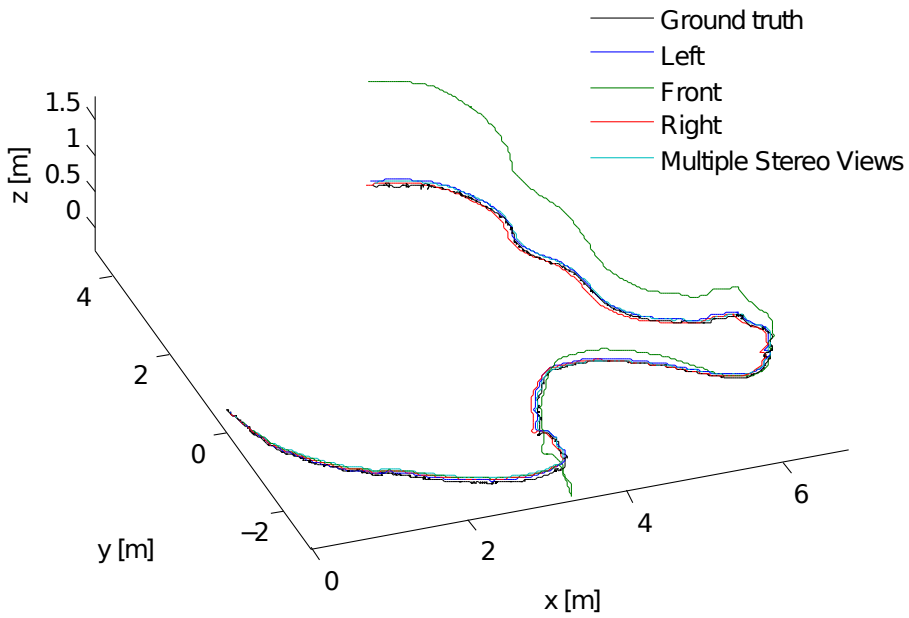
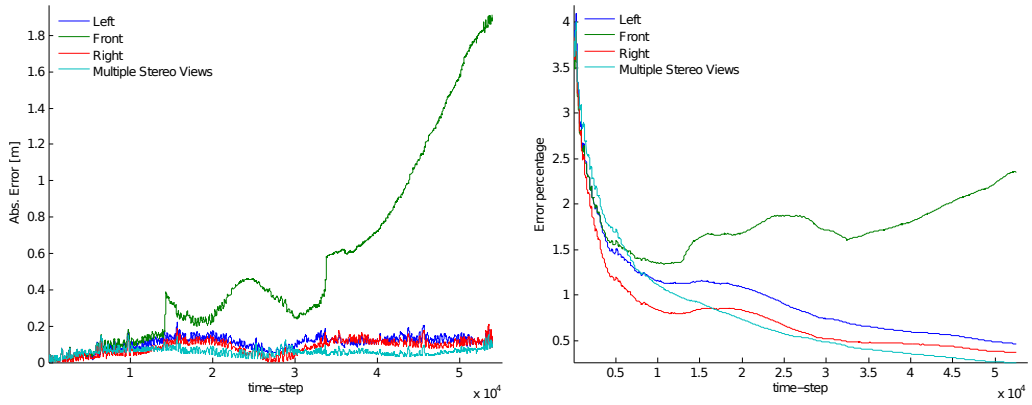


Figure 8.9: Estimation results for single stereo and multiple stereo views approaches. Multiple stereo view estimations are computed using multiple stereo view optimisation.



(a)



(b) (c)

Figure 8.10: Estimation results for single stereo and multiple stereo views approaches, multiple stereo view estimations are computed using multiple stereo view optimisation with a reduced size input set.

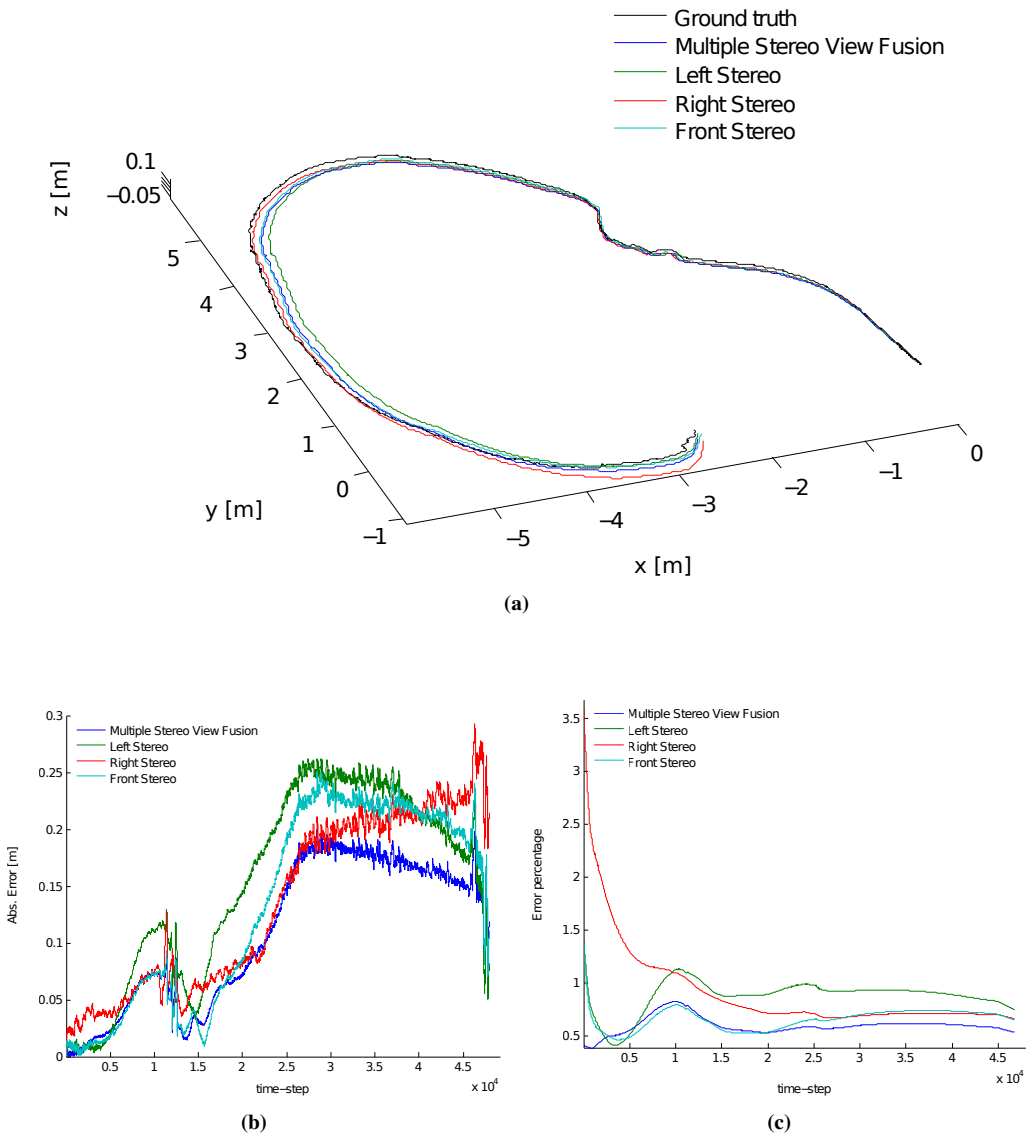


Figure 8.11: Estimation results for single and multiple stereo view fusion approaches.

8.7 Advanced use of multiple stereo views

This section explores new ideas based on the usage of multiple stereo view images and the propagation of the error. When a robot able to acquire multiple view images moves it is easy to think that certain overlaps will occur between the images obtained at the different views, Figure 8.3. This is not a new concept itself, as it is indeed basis for photogrammetry, mosaicing and egomotion. However, this section does not focus on the use of the information due to overlap of the field of view that takes place on the images collected from a single camera view, instead, it focuses on *how the overlap of the field of view of different stereo views can be exploited to improve egomotion.*

Furthermore, this section studies a new way of incorporating covariance propagation techniques to the egomotion solutions previously presented in this work, without need of formulating a full SLAM problem.

8.7.1 Covariance propagation

The use of covariance propagation is widely employed in different fields of science and engineering. This allows modelling the uncertainty of the system's measurements and how that uncertainty propagates through to other states and variables of the system. The application of covariance propagation to algorithms and computer vision has been studied in the literature along the last decades [123, 124, 125].

Here, the intention is propagating the position of the 3D points observed from the robot at a certain pose to a later pose reference frame. In order to do that, the propagation of the covariance of the detected 2D features through the image projective model, the propagation of the covariance through the 3D reconstruction model, the propagation of the covariance through the transformation that defines relative poses and the

propagation of the covariance through the reprojection algorithm of the estimated new position of the 3D points are necessary.

8.7.1.1 Covariance propagation on 3D reconstruction

Assuming that error level of the position of the detected features used to compute the 3D points observed through a stereo camera, the following can expressions can be written to model the covariance propagation:

$$\boldsymbol{g}_1(\boldsymbol{x},\boldsymbol{w}) = \boldsymbol{g}_1(\{u_L, v_L, u_R\}, \{Bl, f, u_0, v_0\}) = \left\{ \begin{array}{c} Bl\,\dfrac{(u_0-u_L)}{u_L-u_R} \\ Bl\,\dfrac{(v_0-v_L)}{u_L-u_R} \\ -Bl\,\dfrac{f}{u_L-u_R} \end{array} \right\} \tag{8.10}$$

$$\boldsymbol{J}_{\boldsymbol{x}} = \left[\begin{array}{ccc} -Bl\,\dfrac{(u_0-u_R)}{(u_L-u_R)^2} & 0 & Bl\,\dfrac{(u_0-u_L)}{(u_L-u_R)^2} \\ -Bl\,\dfrac{(v_0-v_L)}{(u_L-u_R)^2} & -\dfrac{Bl}{u_L-u_R} & Bl\,\dfrac{(v_0-v_L)}{(u_L-u_R)^2} \\ Bl\,\dfrac{f}{(u_L-u_R)^2} & 0 & -Bl\,\dfrac{f}{(u_L-u_R)^2} \end{array} \right] \tag{8.11}$$

$$\boldsymbol{J}_{\boldsymbol{w}} = \left[\begin{array}{cccc} \dfrac{u_0-u_L}{u_L-u_R} & 0 & \dfrac{Bl}{u_L-u_R} & 0 \\ \dfrac{v_0-v_L}{u_L-u_R} & 0 & 0 & \dfrac{Bl}{u_L-u_R} \\ -\dfrac{f}{u_L-u_R} & -\dfrac{Bl}{u_L-u_R} & 0 & 0 \end{array} \right] \tag{8.12}$$

$$C_{xyz,0} = J_w \cdot C_{st} \cdot {J_w}^T + J_x \cdot C_{uv,0} \cdot {J_x}^T \tag{8.13}$$

where

- $\boldsymbol{x}$ is the reduced vector of projected coordinates
- $\boldsymbol{w}$ is the vector of parameters for the calibration of the stereo pair
- $\boldsymbol{g}_1(\boldsymbol{x},\boldsymbol{w})$ is the reconstruction function
- Bl is the baseline
- u_0, v_0 are the coordinated of the principal point of the camera
- f is the focal length
- (u_L, v_L) and (u_R, v_R) are pixel coordinates on cameras (L)eft and (R)ight
- C_{st} is the covariance of the calibration parameters of the stereo camera
- $C_{uv,0}$ is the covariance a the 2D feature
- $C_{xyz,0}$ is the covariance of the reconstructed point $\mathbf{x} = \boldsymbol{g}_1(\boldsymbol{x},\boldsymbol{w})$
- $\boldsymbol{J}_{\boldsymbol{x}}$ and $\boldsymbol{J}_{\boldsymbol{w}}$ are the Jacobian matrices of $\boldsymbol{g}_1(\boldsymbol{x},\boldsymbol{w})$ with respect to $\boldsymbol{x}$ and $\boldsymbol{w}$ respectively

8.7.1.2 Covariance propagation on a transformation model

As for the propagation of the covariance on the algorithm used for the 3D reconstruction, the covariance propagation is also applied to the transformation that describes position and attitude transition from one pose to another.

$$\boldsymbol{g}_2(\boldsymbol{x},\boldsymbol{p}) = \boldsymbol{g}_2(\{x,y,z\},\{\alpha,\beta,\gamma,t_x,t_y,t_z\}) =$$

$$= \begin{bmatrix} C_\beta C_\gamma & -C_\beta S_\gamma & S_\beta \\ \left(C_\alpha S_\gamma + C_\gamma S_\alpha S_\beta\right) & \left(C_\alpha C_\gamma - S_\alpha S_\beta S_\gamma\right) & -C_\beta S_\alpha \\ \left(S_\alpha S_\gamma - C_\alpha C_\gamma S_\beta\right) & \left(C_\gamma S_\alpha + C_\alpha S_\beta S_\gamma\right) & C_\alpha C_\beta \end{bmatrix} \cdot \{\boldsymbol{x_0} - \boldsymbol{t}\} \tag{8.14}$$

$$\boldsymbol{J_x} = R_{xyz} = \begin{bmatrix} C_\beta C_\gamma & -C_\beta S_\gamma & S_\beta \\ C_\alpha S_\gamma + C_\gamma S_\alpha S_\beta & C_\alpha C_\gamma - S_\alpha S_\beta S_\gamma & -C_\beta S_\alpha \\ S_\alpha S_\gamma - C_\alpha C_\gamma S_\beta & C_\gamma S_\alpha + C_\alpha S_\beta S_\gamma & C_\alpha C_\beta \end{bmatrix} \tag{8.15}$$

$$\boldsymbol{J_p} = \begin{bmatrix} \boldsymbol{J}_\alpha & \boldsymbol{J}_\beta & \boldsymbol{J}_\gamma & \boldsymbol{J}_{t_x} & \boldsymbol{J}_{t_y} & \boldsymbol{J}_{t_z} \end{bmatrix} \tag{8.16}$$

$$\boldsymbol{J}_\alpha = \begin{bmatrix} \left(S_\alpha S_\gamma - C_\alpha C_\gamma S_\beta\right)(t_y - y) - \left(C_\alpha S_\gamma + C_\gamma S_\alpha S_\beta\right)(t_x - z) \\ \left(C_\gamma S_\alpha + C_\alpha S_\beta S_\gamma\right)(t_y - y) - \left(C_\alpha C_\gamma - S_\alpha S_\beta S_\gamma\right)(t_x - z) \\ C_\alpha C_\beta\,(t_y - y) + C_\beta S_\alpha\,(t_x - z) \end{bmatrix} \tag{8.17}$$

$$\boldsymbol{J}_\beta = \begin{bmatrix} C_\gamma S_\beta\,(t_x - x) + C_\alpha C_\beta C_\gamma\,(t_x - z) - C_\beta C_\gamma S_\alpha\,(t_y - y) \\ C_\beta S_\alpha S_\gamma\,(t_y - y) - C_\alpha C_\beta S_\gamma\,(t_x - z) - S_\beta S_\gamma\,(t_x - x) \\ C_\alpha S_\beta\,(t_x - z) - C_\beta\,(t_x - x) - S_\alpha S_\beta\,(t_y - y) \end{bmatrix} \tag{8.18}$$

$$\boldsymbol{J}_\gamma = \begin{bmatrix} C_\beta S_\gamma\,(t_x - x) - \left(C_\gamma S_\alpha + C_\alpha S_\beta S_\gamma\right)(t_x - z) - \left(C_\alpha C_\gamma - S_\alpha S_\beta S_\gamma\right)(t_y - y) \\ \left(C_\alpha S_\gamma + C_\gamma S_\alpha S_\beta\right)(t_y - y) + \left(S_\alpha S_\gamma - C_\alpha C_\gamma S_\beta\right)(t_x - z) + C_\beta C_\gamma\,(t_x - x) \\ 0 \end{bmatrix} \tag{8.19}$$

$$\boldsymbol{J}_{t_x} = \begin{bmatrix} -C_\beta C_\gamma \\ C_\beta S_\gamma \\ -S_\beta \end{bmatrix} \tag{8.20}$$

$$\boldsymbol{J}_{t_y} = \begin{bmatrix} -C_\alpha S_\gamma - C_\gamma S_\alpha S_\beta \\ S_\alpha S_\beta S_\gamma - C_\alpha C_\gamma \\ C_\beta S_\alpha \end{bmatrix} \tag{8.21}$$

$$\boldsymbol{J}_{t_z} = \begin{bmatrix} C_\alpha C_\gamma S_\beta - S_\alpha S_\gamma \\ -C_\gamma S_\alpha - C_\alpha S_\beta S_\gamma \\ -C_\alpha C_\beta \end{bmatrix} \tag{8.22}$$

$$\boldsymbol{C}_{xyz,1} = \boldsymbol{J}_x \cdot \boldsymbol{C}_{xyz,0} \cdot \boldsymbol{J}_x{}^T + \boldsymbol{J}_p \cdot \boldsymbol{C}_p \cdot \boldsymbol{J}_p{}^T \tag{8.23}$$

where

- $\boldsymbol{x}$ is the vector 3D coordinates
- $\boldsymbol{p}$ is the vector of parameters a transformation
- $\boldsymbol{g}_2(\boldsymbol{x}, \boldsymbol{p})$ is the transformation function
- t_x, t_y and t_z are translation of the transformation
- α, β and γ are Euler angles of the transformation
- S_θ and C_θ are the sin and cos functions for any angle θ
- C_p is the covariance of the transformation parameters
- $C_{xyz,0}$ is the covariance of the position before the transformation

- $C_{xyz,1}$ is the covariance of the position after the transformation
- $\boldsymbol{J_x}$ and $\boldsymbol{J_p}$ are the Jacobian matrices of $\boldsymbol{g}_2(\boldsymbol{x},\boldsymbol{p})$ with respect to $\boldsymbol{x}$ and $\boldsymbol{p}$ respectively

8.7.1.3 Covariance propagation on the projective model

Similar to the previous, the propagation of the covariance is applied to the transformation that takes places on the stereo header.

$$\boldsymbol{g_3}(\boldsymbol{x},\boldsymbol{w}) = \boldsymbol{g}_3(x,y,z,Bl,f,u_0,v_0) =$$

$$\boldsymbol{g}_3(\{x,y,z\},\{Bl,f,u_0,v_0)\} = \begin{bmatrix} u_0 + f\dfrac{x}{z} \\ v_0 + f\dfrac{y}{z} \end{bmatrix} \tag{8.24}$$

$$\boldsymbol{J}_x = \begin{bmatrix} \dfrac{f}{z} & 0 & -f\dfrac{x}{z^2} \\ 0 & \dfrac{f}{z} & -f\dfrac{y}{z^2} \end{bmatrix} \tag{8.25}$$

$$\boldsymbol{J}_w = \begin{bmatrix} 0 & \dfrac{x}{z} & 1 & 0 \\ 0 & \dfrac{y}{z} & 0 & 1 \end{bmatrix} \tag{8.26}$$

$$\boldsymbol{C}_{uv,1} = \boldsymbol{J}_x \cdot \boldsymbol{C}_{xyz,1} \cdot \boldsymbol{J}_x{}^T + \boldsymbol{J}_w \cdot \boldsymbol{C}_{st} \cdot \boldsymbol{J}_w{}^T \tag{8.27}$$

where

- $\boldsymbol{x}$ is the vector 3D coordinates to project
- $\boldsymbol{w}$ is the vector of parameters for the calibration of the stereo pair
- $\boldsymbol{g}_3(\boldsymbol{x},\boldsymbol{p})$ is the reprojection function

- S_θ and C_θ are the sin and cos functions for any angle θ
- C_{st} is the covariance of the calibration parameters of the stereo camera
- $C_{xyz,1}$ is the covariance of the position
- $C_{uv,1}$ is the covariance of the reprojected coordinates

Applying the propagation of the covariance on the 2D features observed on the cameras at the pose k and the estimation of rover's motion between poses k and $k+n$, for any number of transitions n, the position of the features at k can be propagated to the reference frame. This leads to ellipsoids of confidence, where the points observed at k propagate, Figure 8.12.

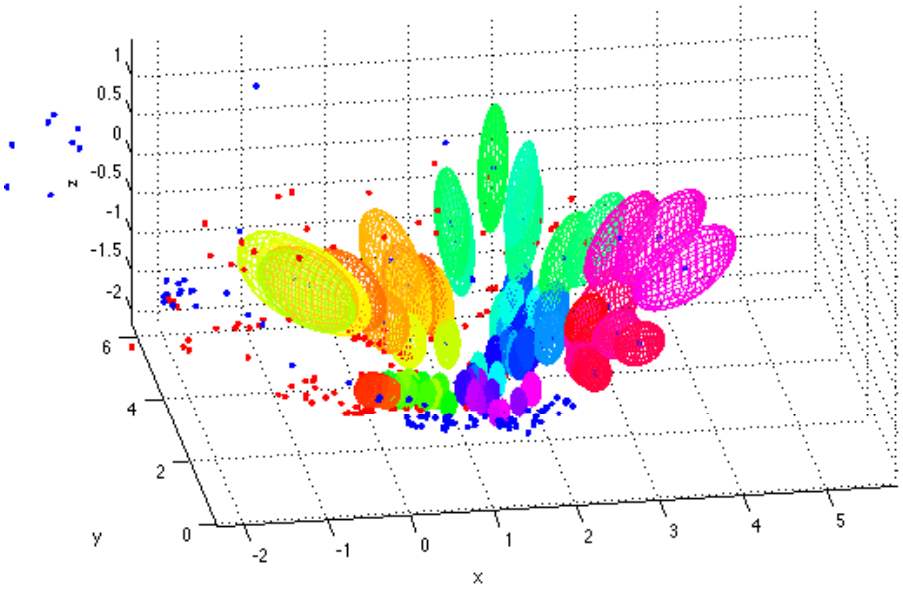


Figure 8.12: Propagation of the covariance of the points detected on a Right stereo view pose (pan angle of 30 [deg]) to a later reference frame the rear stereo view (pan angle of 180 [deg]).

8.7.2 Multiple stereo view image registration on Egomotion

Image registration or image alignment is the process by which the geometrical relationship between two images is determined. Many alternative models can be applied to solve this problem, as affine transformations and projective transformation. But the later is the most versatile.

Here, we propose using the rear stereo views combined with the frontal stereo views to close the visual loop online from the robot. This task is performed through the association of the features that appear first in frontal views and that are reobserved after on the rear stereo view. When that happens, a registration algorithm can be applied to guarantee the validity of the results of the mobile system online.

A prior task to the image registration when discrete features are used, as opposed to using the whole image information, is the association of equivalent features

8.7.2.1 Brute Force

A brute force process, in terms of matching of association, consists of making all the possible comparisons of the descriptors of the two sets of features to be associated. This task is entails a minimum of $n \times m$ comparisons, where n and m are the sizes of the sets of features to associate.

8.7.2.2 Covariance Guided Association

Using the concepts previously explained in this section, it is possible to reduce the number of candidates used to compute the association pairs. This is possible due to the calculation of the confidence ellipses where the 2D features should propagate from one image to another.

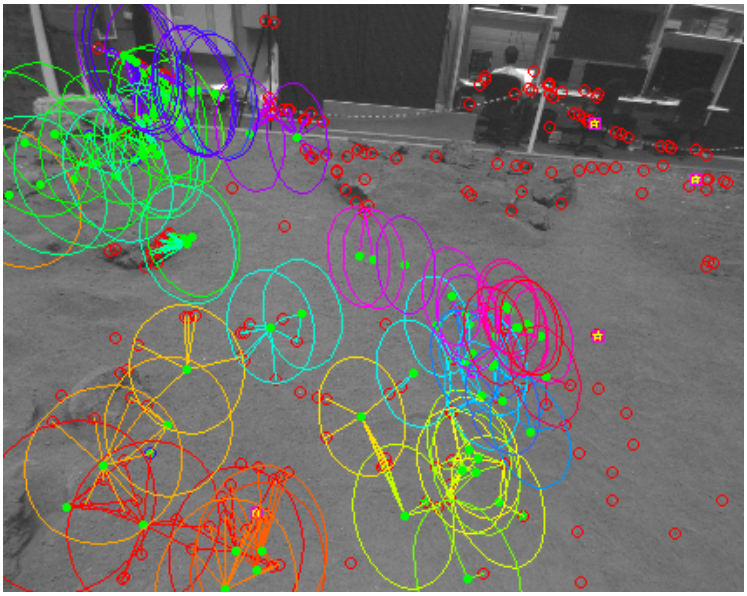


Figure 8.13: Association candidates on Covariance Guided Association. The features from another view propagate as ellipses. Association of the previous features with the ones on the previous frame is only possible is the new features lay within the ellipse of trust.

Figure 8.13 shows how the candidates are computed using the ellipses of trust obtained by means of the propagation of the covariance.

Figure 8.14 shows the associated features found using Brute Force (magenta) and Covariance Guided Association (cyan). For this case, a number associations of 41 was obtained using Covariance Guided Association, whereas only 15 associations could be found using Brute Force. Then Ransac registration algorithm was applied to compute the homography given by those matches. A number of 15 inliers out of 41 associated pairs were found using Guided Association strategy, whereas only 7 inliers out of 15 were found Brute Force.

This allows us to consider that the Covariance Guided Association process is the best solution to identify the associated features. Which can then be applied to image registration of multiple stereo view sequences to check online the validity of the results.

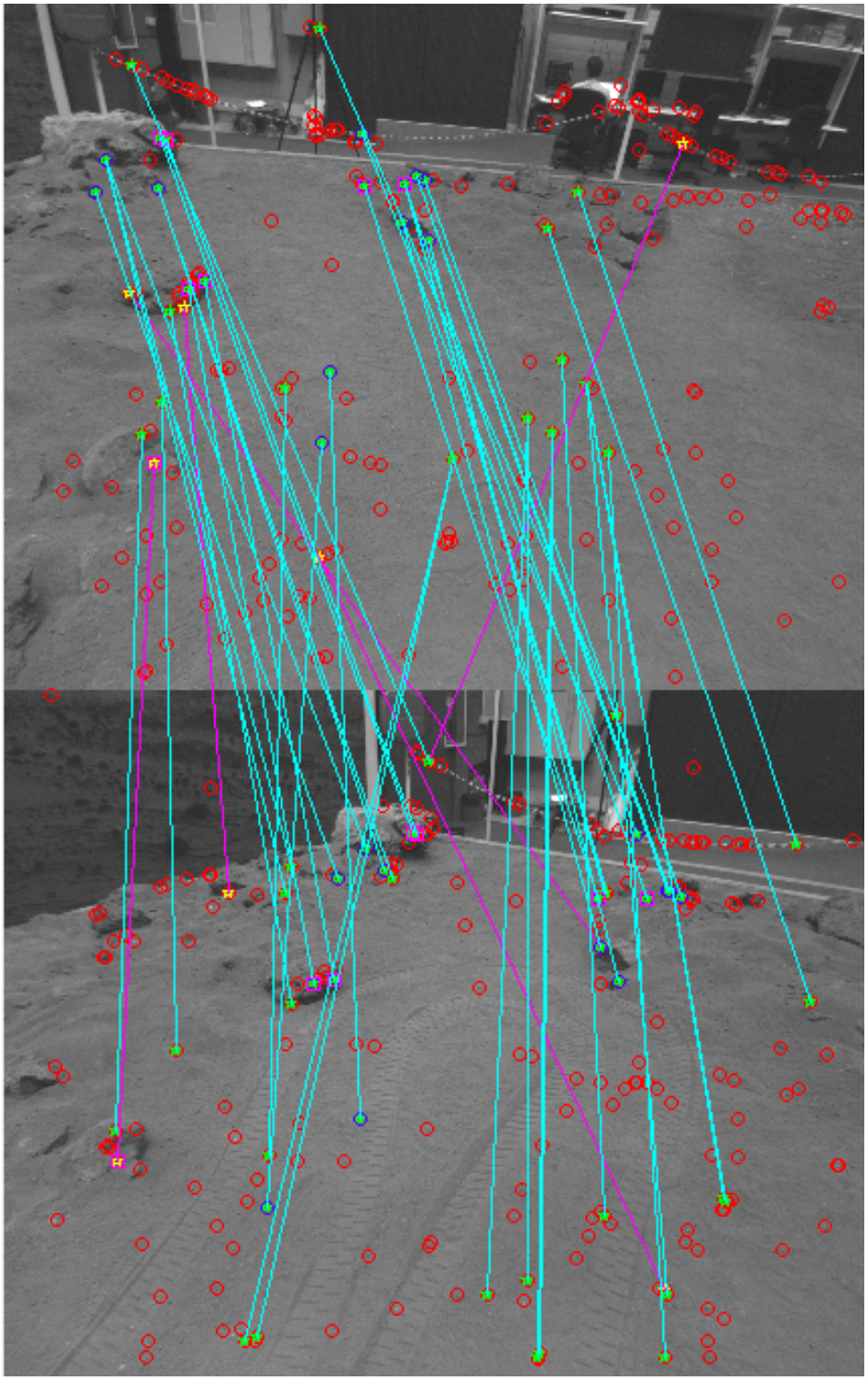


Figure 8.14: Association for multiple view image registration. Lines in cyan represent the associated features using descriptor matching aided by propagation of the covariance, whereas the magenta lines correspond to the matches obtained through a brute force process.

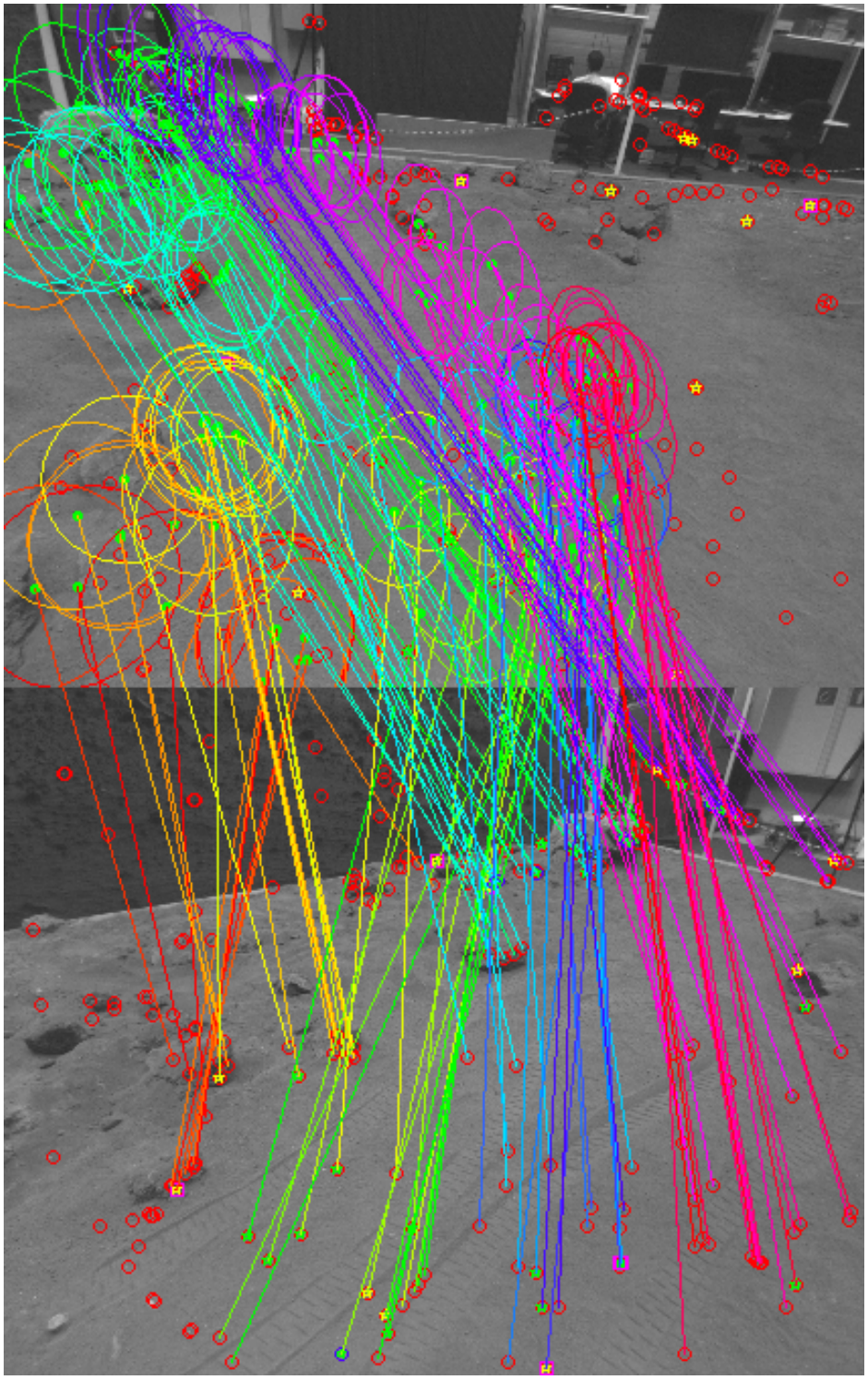


Figure 8.15: Propagation of the covariance for multiple stereo view images. The ellipses represent the area where the features propagate to with a confidence of 97%. These areas are used on the Covariance Guided Association process

8.8 Conclusions

As a continuation of the motion estimation work that we presented in the previous chapters, two different approaches based on our robust visual scheme have been proposed here to improve the accuracy of motion estimations when multiple stereo views are available.

Although solutions to the motion estimation problem based on single stereo cameras are eligible to provide accurate results, they present limitations that a system based on multiple stereo views is able to overcome. Multiple stereo view systems can be easily implemented by adding extra pairs of stereo cameras or just by incorporating pan-tilt mechanisms to the camera header.

A first solution is proposed to deal with the data fusion problem derived from using imagery from different stereo camera views. This solution, based on optimisation techniques, has been proved to provide accurate results and to present better convergence proprieties that single stereo based solutions. One of the hypothesis to formulate this approach is that all the views are acquired from a static pose of the robot. Nevertheless, the results show that, although desirable, this is not a strong requirement for the solution to work. A second solution to fuse the multiple motion estimations, obtained from single stereo view sequences, through the Covariance Intersection method has also been proved to be an successful technique to add accuracy to the end estimation results when multiple stereo views are obtainable. Conversely to the other solution presented here, this filtering approach assumes that all the motion estimations obtained from the multiple views are valid.

The proposed approaches have been validated representative datasets, for which the latest techniques to generate ground truth data were used.

Chapter 9

Conclusions and Future work

This thesis has studied various vision based techniques targeting the robustness and applicability of stereo imagery solutions. After thorough analysis and investigations, solutions have been provided where it was thought necessary and challenging.

In Chapter 3 low level image processing is tackled. With the goal of providing reliable and useful image characterisation image moments techniques are combined with robust local image descriptors. This innovative combination results in a reliable, yet efficient, method that can be later used for higher processing levels. It solves problems of robustness against illumination changes and robust feature identification.

Chapter 4 and Chapter 5 study the integration of the presented visual methods to VSLAM, as well as some map management enhancements introduced into the algorithms. First, the introduction of the concept *rate of usability* allows the VSLAM determining which landmarks are less likely to be later used, in order to alleviate the system's growth. Then, the addition of HMSURF as a visual module shows to provide promising results.

In Chapter 6 motion estimation techniques based on linear methods are put under the microscope to investigate robust methodologies eligible for long range Egomotion. Along with the different visual techniques an alternative based on optical flow is also analysed. The combination of a RANSAC algorithm together with a quaternion motion estimation technique is shown to produce precise results.

Chapter 7 reviews the optimisation methods used for robust Egomotion. Different feature reprojection alternatives are analysed to determine what is the best recourse. Extensive experimental results are presented to decide which photogrammetry option leads to lower levels of error. Our dual reprojection method for a single inter-frame sliding window is highlighted as the best alternative, providing more stable and accurate results.

Chapter 8 studies the possibility of exploiting multiple stereo view sequences, used in order to improve the reliability of visual Egomotion. Two methods are proposed by the author to cope with the data fusion problem derived from using imagery acquired at different views. The proposed methods have shown to provide even higher levels of accuracy and robustness. A thorough experimental validation has been undertaken to guarantee the quality of the analysis.

Future work

As every research work, this thesis has had to be limited by time and funds. Nonetheless, new implications are derived from the investigated topics and proposed solutions that can be starting points or basis for further research projects.

In chapter 3, it would be interesting testing the illumination robustness of features, when moment images are combined with other sort of feature descriptors. New solutions to the feature detection and description problems are continuously appearing in the literature to guarantee higher levels of invariance. Also, the possibility of generating local descriptors using image moment representation is a topic that could be further investigated.

Chapter 4 and chapter 5 and other alternative implementations for the VSLAM algorithm, as the graph-SLAM solutions, could be adapted to integrate the visual modules presented here. Also, the integration of our VSLAM solutions with other sort of emerging devices, as active camera devices for motion sensing (Kinect sensor), could be analysed.

In chapter 6 linear estimation techniques were proved to be a suitable vehicle to compute the motion estimation, that could be later used as preemptive data for nonlinear optimisation techniques.

Chapter 7 some alternatives to the used feature reprojection strategies could be analysed, in order to determine improved trade-offs, for different sizes of the sliding window. Other optimisation techniques can be adapted for the motion estimation, as particle swarm optimisation or convex optimisation.

Methods developed in chapter 8 could be extended to use the multiple stereo view images for a new variety of purposes. If multiple stereo views are obtained so that their fields of view overlap, visual processing techniques can be investigated for online camera re-calibration. Likewise, frontal stereo views can be combined with rear stereo to perform visual loop-closure and guided covariance association. This could lead in later studies to subsequent incremental re-estimation of the motion estimates, using the backward view.

Bibliography

[1] R. Smith and P Cheeseman. On the representation and estimation of spatial uncertainty. *International Journal of Robotics Research (IJRR)*, 5(4):56–68, 1986. 2

[2] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. *Autonomous robot vehicles*, 1:167–193, 1990. 2

[3] J.J. Leonard and H.F. Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *Proceedings of the IEEE/RSJ International Workshop on Intelligent Robots and Systems. 'Intelligence for Mechanical Systems'*, pages 1442 –1447 vol.3, nov 1991. 2

[4] Juan Manuel Sáez, Andrew Hogue, Francisco Escolano, and Michael Jenkin. Underwater 3D SLAM through Entropy Minimization. In *Proceedings of the IEEE Conference on Robotics and Automation (ICRA)*, pages 3562–3567, 2006. 2, 63

[5] J.D. Tardós, J. Neira, P.M. Newman, and J.J. Leonard. Robust mapping and localization in indoor environments using sonar data. *The International Journal of Robotics Research*, 21(4):311, 2002. 2

[6] P Hoppen, T Knieriemen, and E Puttkamer. Laser-Radar Based Mapping and Navigation for an Autonomous Mobile Robot. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 948–953, Cincinnati, OH, 1990. 2, 12, 62

[7] Bill Green. Tutorial on Simultaneous Localization and Mapping (SLAM). http://prism2.mem.drexel.edu/ billgreen/slam/slam.html. 2

[8] David Schleicher, Luis M. Bergasa, Manuel Ocaña, Rafael Barea, and María Elena López. Real-time hierarchical outdoor SLAM based on stereovision and GPS fusion. *IEEE Transactions on Intelligent Transportation Systems*, 10(3):440–452, 2009. 2, 63

[9] Dan Simon. *Optimal State Estimation: Kalman, H Infinity and Non Linear Approaches* . Wiley-Interscience, 2006. 2, 64, 66

[10] Greg Welch and Gary Bishop. An Introduction to the Kalman Filter. Technical report, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 1995. 2

[11] Abdelkrim Nemra and Nabil Aouf. Robust Airborne 3D Visual Simultaneous Localization and Mapping with Observability and Consistency Analysis. *Journal of Intelligent & Robotic Systems*, 55(4-5):345–376, 2009. 3, 63, 75

[12] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002. AAAI. 3

[13] S. Thrun, W. Burgard, and D. Fox. A Real-Time Algorithm for Mobile Robot Mapping With Applications to Multi-Robot and 3D Mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2000. 3

[14] S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, and A. Ng. Simultaneous mapping and localization with sparse extended information filters: Theory and initial results. *Algorithmic Foundations of Robotics V*, pages 363–380, 2004. 3, 63

[15] L.M. Paz, J.D. Tardós, and J. Neira. Divide and Conquer: EKF SLAM in O(n). *IEEE Transactions on Robotics*, 24(5):1107–1120, 2008. 3

[16] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo localization for mobile robots. *Artificial intelligence*, 128(1):99–141, 2001. 3

[17] J. Neira and J.D. Tardós. Data Association in Stochastic Mapping Using the Joint Compatibility Test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, December 2001. 3

[18] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental Smoothing and Mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, Dec 2008. 3

[19] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J.J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research (IJRR)*, 31:217–236, Feb 2012. 3

[20] J. L. Blanco, J. Gonzalez, and J. A. Fernandez-Madrigal. An optimal filtering algorithm for non-parametric observation models in robot localization. In

Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pages 461 –466, May 2008. 3, 63

[21] P. Piniés, L.M. Paz, D. Gálvez-López, and J.D. Tardós. CI-Graph simultaneous localization and mapping for three-dimensional reconstruction of large and complex environments using a multicamera system. *Journal of Field Robotics*, 27(5):561–586, 2010. 3

[22] F. Dellaert and M. Kaess. Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing. *International Journal of Robotics Research (IJRR)*, 25(12):1181–1204, Dec 2006. 3

[23] Niko Sünderhauf and Peter Protzel. Stereo Odometry - A Review of Approaches. Technical report, Chemnitz University of Technology, 2007. 4, 121

[24] D. Scaramuzza and F. Fraundorfer. Visual odometry [tutorial]. *IEEE Robotics & Automation Magazine*, 18(4):80–92, 2011. 4, 158

[25] F. Fraundorfer and D. Scaramuzza. Visual Odometry: Part II: Matching, Robustness, Optimization, and Applications. *IEEE Robotics & Automation Magazine*, 19(2):78–90, 2012. 4, 158

[26] Carlo Tomasi. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision (IJCV)*, 9:137–154, 1992. 4, 26, 34, 58

[27] David G. Lowe. Fitting parametrized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 13(5):441–450, 1991. 4, 32

[28] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. *Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346–359*, 2008. 4, 26, 41, 54, 55, 120, 138, 139, 167

[29] Y. Meng, B. Tiddeman, et al. Implementing the Scale Invariant Feature Transform (SIFT) Method. *Department of Computer Science University of St. Andrews*, 2008. 4

[30] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th international conference on Multimedia*, MULTIMEDIA '07, pages 357–360, New York, NY, USA, 2007. ACM. 4

[31] S. Leutenegger, M. Chli, and R. Y. Siegwart. BRISK: Binary Robust invariant scalable keypoints. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2548–2555. IEEE, November 2011. 4

[32] P. F. Alcantarilla, A. Bartoli, and A. J. Davison. KAZE Features. In *European Conference on Computer Vision (ECCV)*, 2012. 4

[33] P.F. Alcantarilla, L.M. Bergasa, and A.J. Davison. Gauge-SURF Descriptors. *Image and Vision Computing Volume 31, Issue 1, January 2013, Pages 103–116*, 2013. 4

[34] Murat Gevrekci and Bahadir K. Gunturk. Illumination robust interest point detection. *Computer Vision and Image Understanding*, 113(4):565 – 571, 2009. 4

[35] Arturo Gil, Oscar Martinez Mozos, Monica Ballesta, and Oscar Reinoso. A

Comparative Evaluation of Interest Point Detectors and Local Descriptors for Visual SLAM. *Machine Vision and Applications (MVA)*, 2009. Published online. 5

[36] N. Muhammad, D. Fofi, and S. Ainouz. Current state of the art of vision based SLAM. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 7251 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, February 2009. 5

[37] L. Matthies and S. Shafer. Error Modeling in stereo navigation. *IEEE Journal of Robotics and Automation*, 3(3), 1987. 5

[38] L. Matthies. *Dynamic stereo vision*. Cmu-cs-89-195, Carnegie Mellon University. Computer Science Department, 1989. 5

[39] Andrew Johnson, James F. Montgomery, and Larry Matthies. Vision guided landing of an autonomous helicopter in hazardous terrain. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3977–3982, May 2005. 5

[40] Peter Corke, Dennis Strelow, and Sanjiv Singh. Omnidirectional visual odometry for a planetary rover. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Japan, 2004. 5

[41] Jinwoo Choi, Sunghwan Ahn, and Wan Kyun Chung. Robust sonar feature detection for the SLAM of mobile robot. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 3415 – 3420, aug. 2005. 12

[42] M.W.M.G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and

M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229 –241, jun 2001. 12

[43] D.M. Cole and P.M. Newman. Using laser range data for 3D SLAM in outdoor environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1556 –1563, May 2006. 12, 62

[44] P. Newman, D. Cole, and K. Ho. Outdoor slam using visual appearance and laser ranging. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1180 –1187, may 2006. 12

[45] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, June 2007. 12, 63, 119

[46] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 12, 19

[47] Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, March 1998. 12, 17

[48] Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, pages 666–673, 1999. 14

[49] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Proceedings of the IEEE Computer Society Con-*

ference on Computer Vision and Pattern Recognition, pages 1106–1112. IEEE, June 1997. 14

[50] J.Y. Bouguet. Camera calibration toolbox for Matlab, 2004. 22

[51] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 22, 51

[52] Stephen M. Smith and J. Michael Brady. SUSAN - A New Approach to Low Level Image Processing. *International Journal of Computer Vision*, 23:45–78, 1997. 10.1023/A:1007963824710. 26

[53] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Proceedings of the 9th European conference on Computer Vision - Volume Part I*, ECCV'06, pages 430–443, Berlin, Heidelberg, 2006. Springer-Verlag. 26

[54] I. Landesa-Vázquez, F. Parada-Loira, and J.L. Alba-Castro. Fast real-time multiclass traffic sign detection based on novel shape and texture descriptors. In *13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1388 –1395, sept. 2010. 26

[55] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1 –8, june 2007. 26

[56] Zhu Hong and Zhang Guoying. Texture Feature Clustering and Segmentation

Algorithm Based on Pixel. In *2nd International Conference on Information Engineering and Computer Science (ICIECS)*, pages 1 –4, dec. 2010. 26

[57] B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 674–679, Vancouver, Canada, 1981. 26

[58] Lowe D. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004. 26

[59] Diego Rodriguez and Nabil Aouf. Robust Harris-SURF features for robotic vision based navigation. In *13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1160 –1165, September 2010. 26, 44, 120, 139, 167

[60] C. Harris and M. Stephens. A Combined Corner and Edge Detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988. 27, 28, 49, 139, 167

[61] H. P. Moravec. Towards automatic visual obstacle avoidance. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, page 584, Cambridge, Mass, 1977. MIT. 28

[62] N.D.B. Bruce and P. Kornprobst. Harris corners in the real world: A principled selection criterion for interest points based on ecological statistics. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2160–2167. IEEE, 2009. 28

[63] Jianbo Shi and Tomasi. Good features to track. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600. IEEE Comput. Soc. Press, June 1994. 34, 120, 138, 139, 167

[64] S. Parkes, M. Dunstan, D. Matthews, I. Martin, and V. Silva. LIDAR-based GNC for Planetary Landing: Simulation with PANGU. In *DASIA 2003*, volume 532, page 18, 2003. 36, 52

[65] SM Parkes, I. Martin, M. Dunstan, and D. Matthews. Planet surface simulation with PANGU. In *Eighth International Conference on Space Operations*, pages 1–10, 2004. 36, 52

[66] European Space Agency. *Validation and verification approach for European safe precision landing guidance, Navigation and Control (GNC) Technologies*. Georgia Institute of Technology, 2008. 36, 52

[67] M. Muja and D.G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *Proceedings of the International Conference on Computer Vision Theory and Applications (VISSAPP)*, pages 331–340, 2009. 37

[68] R. Mukundan and K.R. Ramakrishnan. *Moment Functions in Image Analysis: Theory and Applications*. World Scientific, 1998. 43, 167

[69] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(10):1615–1630, 2005. 44, 45

[70] M. Kharbat, N. Aouf, A. Tsourdos, and B. White. Robust Brightness Description for Computing Optical Flow. In *Proceedings of the British Machine Vision Conference (BMVC)*, page 10, 2008. 44, 167

[71] S. Birchfield. KLT: An implementation of the Kanade-Lucas-Tomasi feature tracker, 1998. 51

[72] R. Hess. An open-source SIFTLibrary. In *Proceedings of the international conference on Multimedia*, pages 1493–1496. ACM, 2010. 51

[73] C. Evans. Notes on the opensurf library. *University of Bristol, Tech. Rep. CSTR-09-001, January*, 2009. 51

[74] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proceedings of the Europea Conference in Computer Vision (ECCV)*, pages 128–142, 2002. 54

[75] FA Moreno, JL Blanco, and J. Gonzalez. Stereo vision specific models for particle filter-based slam. *Robotics and Autonomous Systems*, 57(9):955–970, 2009. 55, 63

[76] B. Barshan and H.F. Durrant-Whyte. Inertial navigation systems for mobile robots. *IEEE Transactions on Robotics and Automation*, 11(3):328–342, 1995. 62

[77] Johann Borenstein and Liqiang Feng. Measurement and Correction of Systematic Odometry Errors in Mobile Robots. In *IEEE Transactions on Robotics and Automation*, pages 869–880, 1996. 62

[78] Jonghyuk Kim and Salah Sukkarieh. Real-time implementation of airborne inertial-SLAM. *Robotics and Autonomous Systems 55.1 (2007): 62-71.*, 55(1):62–71, January 2007. 63

[79] M. Bryson and S. Sukkarieh. Building a Robust Implementation of Bearing-only Inertial SLAM for a UAV. *Journal of Field Robotics*, 24(1-2):113–143, 2007. 63

[80] S. Kim and S.Y. Oh. SLAM in indoor environments using omni-directional vertical and horizontal line features. *Journal of Intelligent & Robotic Systems*, 51(1):31–43, 2008. 63

[81] M L Benmessaoud, A Laramrani, Karim Nemra, and A K Souici. Single-Camera EKF-vSLAM. In *Proceedings of World Academy of Science, Engineering and Technology (PWASET)*, volume ISSN 1307-6884, July 2008. 63, 76

[82] S. Se, D. Lowe, and J. Little. Vision-based mobile robot localization and mapping using scale-invariant features. In *Proceedings of the IEEE International Conference on Robotics ans Automation (ICRA)*, volume 2, pages 2051–2058 vol.2, 2001. 63

[83] A. Nemra and N. Aouf. Experimental airborne nh vision-based simultaneous localization and mapping in unknown environments. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 224(12):1253–1270, 2010. 63

[84] R. Martinez-Cantin and J.A. Castellanos. Unscented SLAM for large-scale out-

door environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3427 – 3432, aug. 2005. 63

[85] Steven Holmes, Georg Klein, and David W Murray. A square root unscented Kalman filter for visual monoSLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3710–3716, 2008. 63

[86] Óscar Martínez Mozos, Arturo Gil, Mónica Ballesta, and Óscar Reinoso. Interest Point Detectors for Visual SLAM. In *Current Topics in Artificial Intelligence (CAEPIA)*, pages 170–179, 2007. 63

[87] S. Frintrop, P. Jensfelt, and H. I. Christensen. Attentional Landmark Selection for Visual SLAM. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006. 63

[88] Mohammed Boulekchour. Neural Network-aided vSLAM . Master's thesis, Cranfield University, 2008. 74

[89] Abdelkrim. Nemra. *Roboust 3D airbone visual simultaneous localization and mapping*. PhD thesis, Cranfield University, 2010. 96

[90] Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient Large-Scale Stereo Matching. In *Asian Conference on Computer Vision*, Queenstown, New Zealand, November 2010. 115, 132, 151

[91] Bernd Kitt, Andreas Geiger, and Henning Lategahn. Visual Odometry based on Stereo Image Sequences with RANSAC-based Outlier Rejection Scheme. In *IEEE Intelligent Vehicles Symposium*, San Diego, USA, June 2010. 115, 128, 151, 158

[92] A. Broggi, P. Medici, E. Cardarelli, P. Cerri, A. Giacomazzo, and N. Finardi. Development of the Control System for the Vislab Intercontinental Autonomous Challenge. In *13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 635 –640, September 2010. 118

[93] A. Napier, G. Sibley, and P. Newman. Real-time bounded-error pose estimation for road vehicles using vision. In *13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1141 –1146, September 2010. 118

[94] J. Puddy and P. Smith. Safety challenges in deploying unmanned ground vehicles in real world environments. In *Proceedings of the 5th IET International Conference on System Safety 2010*, pages 1 –6, October 2010. 118

[95] M.A. Olivares-Mendez, P. Campoy, C. Martinez, and I. Mondragon. A Pan-Tilt Camera Fuzzy Vision Controller on an Unmanned Aerial Vehicle. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 2879 –2884, October 2009. 118

[96] H. Strasdat, J. M. M. Montiel, and Andrew J. Davison. Real-time monocular SLAM: Why filter? In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2657–2664. IEEE, May 2010. 119

[97] A.E. Johnson, S.B. Goldberg, Yang Cheng, and L.H. Matthies. Robust and Efficient Stereo Feature Tracking for Visual Odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 39 –46, May 2008. 119

[98] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE*

Computer Society Conference on, volume 1, pages I–652 – I–659 Vol.1, 27 June-2 July 2004. 119

[99] D. Rodriguez, N. Aouf, and M. Richardson. Moments-based stereo camera egomotion analysis and results for long-range trajectories. *The Imaging Science Journal*, 2012. 119

[100] Berthold K. P. Horn, Hugh M. Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America A. JOSA A 4.4 (1987): 629-642.*, 5(7):1127, July 1987. 124

[101] Giil Kwon, Yeong Nam Chae, and H.S. Yang. Temporal and spatial 3D motion vector filtering based visual odometry for outdoor service robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3804 –3809, October 2010. 128

[102] Z. Zhang and Y. Shan. Incremental motion estimation through modified bundle adjustment. In *Proceedings International Conference on Image Processing (ICIP)*, volume 2, pages II – 343–6 vol.3, sept. 2003. 137

[103] Gabe Sibley, Christopher Mei, Ian Reid, and Paul Newman. Adaptive Relative Bundle Adjustment. In *Proceedings of the Robotics Science and Systems Conferece (RSS)*, Seattle, USA, June 2009. 137

[104] Kurt Konolige, Motilal Agrawal, and Joan Solà. Large-Scale Visual Odometry for Rough Terrain. In *Robotics Research, 201-212*, pages 201–212, 2011. 138

[105] A. Eudes and M. Lhuillier. Error propagations for local bundle adjustment. In

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2411 –2418, June 2009. 139, 167

[106] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle Adjustment - A Modern Synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, pages 298–372, London, UK, 2000. Springer-Verlag. 141

[107] O. Pink, F. Moosmann, and A. Bachmann. Visual features for vehicle localization and ego-motion estimation. In *IEEE Intelligent Vehicles Symposium*, pages 254 –260, june 2009. 158

[108] M. Cazorla, D. Viejo, J. Nieto, E. Nebot, et al. Large scale egomotion and error analysis with visual features. *Journal of Physical Agents*, 4(1):19–24, 2010. 158

[109] B. Musleh, D. Martin, A. de la Escalera, and J.M. Armingol. Visual ego motion estimation in urban environments based on U-V disparity. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 444 –449, june 2012. 158

[110] Andrew J. Davison and D. Murray. Simultaneous Localization and Map-Building Using Active Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(7):865 – 880, July 2002. 158

[111] W.S. Kim, A.I. Ansar, and R.D. Steele. Rover mast calibration, exact camera pointing, and camera handoff for visual target tracking. In *Proceedings of the 12th International Conference on Advanced Robotics (ICAR)*, pages 384 –391, july 2005. 158

[112] J. Hidalgo, P. Poulakis, J. Köhler, A. Barrientos, and J. Del-Cerro. Estec testbed

capabilities for the performance characterization of planetary rover localization sensors first results on imu investigations. In *11th Symposium on Advanced Space Technologies in Robotics and Automation*, 2011. 160

[113] Jörg Stückler and Sven Behnke. Integrating Depth and Color Cues for Dense Multi-Resolution Scene Mapping Using RGB-D Cameras. In *Proceedings of the IEEE International Conference on Multisensor Fusion and Information Integration (MFI)*, 2012. 160

[114] Diego Rodriguez and Nabil Aouf. Robust EgoMotion for Large-Scale Trajectories. In *IEEE International Conference on Multisensor Fusion and Information Integration (MFI)*, Hamburg, Germany, September 2012. 171

[115] RG Brown. Integrated navigation systems and Kalman filtering: a perspective. *Navigation Vol. 19, No. 4, Winter 1972-1973, pp. 355-362*, 19(4):355–362, 1973. 172

[116] Yunchun Yang, J. Farrell, and M. Barth. High-accuracy, high-frequency differential carrier phase GPS aided low-cost INS. In *IEEE Position Location and Navigation Symposium*, pages 148 –155, 2000. 172

[117] Y. Li, J. Wang, C. Rizos, P. Mumford, and W. Ding. Low-cost tightly coupled GPS/INS integration based on a nonlinear Kalman filtering design. In *Proceedings of the National Technical Meeting of the Institute of Navigation*, pages 958–966, San Diego, 2006. 172

[118] J.K. Uhlmann, S.J. Julier, and M. Csorba. Nondivergent simultaneous map building and localization using covariance intersection. In *AeroSense*, pages 2–11. International Society for Optics and Photonics, 1997. 172

[119] S.J. Julier and J.K. Uhlmann. Using covariance intersection for SLAM. *Robotics and Autonomous Systems*, 55(1):3–20, 2007. 172

[120] P.O. Arambel, C. Rago, and R.K. Mehra. Covariance intersection algorithm for distributed spacecraft state estimation. In *Proceedings of the American Control Conference*, volume 6, pages 4398 –4403 vol.6, 2001. 173

[121] Andreas Geiger, Julius Ziegler, and Christoph Stiller. StereoScan: Dense 3d Reconstruction in Real-time. In *IEEE Intelligent Vehicles Symposium*, Baden-Baden, Germany, June 2011. 173

[122] Chris Harris and Carl Stennett. Practical issues in automatic 3D reconstruction and navigation applications using man-portable or vehicle-mounted sensors. In *SPIE Security + Defence Exhibition*, Edinburg, September 2012. 174

[123] R.M. Haralick. Propagating covariance in computer vision. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision Image Processing*, volume 1, pages 493 –498 vol.1, October 1994. 179

[124] W. Hoff and T. Vincent. Analysis of head pose accuracy in augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 6(4):319 –334, October-December 2000. 179

[125] G. Di Leo, C. Liguori, and A. Paolillo. Covariance Propagation for the Uncertainty Estimation in Stereo Vision. *IEEE Transactions on Instrumentation and Measurement*, 60(5):1664 –1673, May 2011. 179