Real-time Metric Localisation with Wearable Vision Systems

Author: Daniel Gutiérrez Gómez

Supervisor: José Jesús Guerrero Campo

Departamento de Informática e Ingeniería de Sistemas (DIIS)
Instituto de Investigación en Ingeniería de Aragón (I3A)
Escuela de Ingeniería y Arquitectura (EINA)
Universidad de Zaragoza

February 2016

Resumen

Con el rápido desarrollo de la electrónica y la informática en los últimos años, las cámaras se han convertido en dispositivos omnipresentes en nuestra vida diaria, hasta tal extremo que hoy en día casi todo el mundo dispone de una en todo momento acoplada a su teléfono móvil. Lo que hace especialmente atractivas las cámaras para las personas es su capacidad para capturar rápidamente una gran cantidad de información del entorno codificada en una imagen o vídeo, lo que nos permite inmortalizar momentos especiales en nuestra vida o compartir en pocos segundos gran cantidad de información con otras personas. Sin embargo, mientras que la tarea de extraer la información de una imagen puede ser trivial para nosotros, en el caso de un computador se requieren algoritmos complejos con una alta carga computacional para transformar una imagen en información útil para la máquina. En este sentido, el mismo rápido desarrollo de la electrónica y la informática que permitió la universalización de las cámaras, ha permitido también la posibilidad de aplicación en tiempo real de algoritmos cada vez más complejos y potentes. Entre los campos de investigación actuales en la comunidad de visión por computador, esta tesis está particularmente involucrada con algoritmos de localización métrica y reconstrucción 3D. Estos algoritmos son un componente clave en muchas aplicaciones prácticas, tales como la navegación de robots, realidad aumentada 3D o la reconstrucción de modelos del entorno. El objetivo de esta tesis es profundizar en la localización visual y la reconstrucción del entorno a partir de sensores de visión, prestando especial atención tanto a cámaras convencionales como no convencionales que se pueden llevar o ser manejadas por una persona con facilidad. En esta tesis se aportan contribuciones en los siguientes aspectos de los procesos de odometría visual y SLAM (del inglés, Simultaneous Localisation and Mapping):

- **SLAM monocular generalizado:** Los algoritmos de SLAM actuales suelen estar diseñados para cámaras convencionales, que pueden ser modeladas por un modelo simple de cámara estenopeica pero cuyo campo de vista es bastante limitado. Sistemas de visión consistentes en la combinación de una cámara y un espejo de forma cónica, conocidas como cámaras catadióptricas, ofrecen un campo de vista mucho más amplio, pero a cambio requieren de un modelo de calibración más complejo, que generaliza la proyección de los sistema de proyección centrales. Nuestra propuesta en este área es la adaptación para cámaras catadióptricas de un SLAM monocular en tiempo real diseñado inicialmente para cámaras convencionales.

- **Problema de escala en visión monocular:** Un problema de los algoritmos de localización y reconstrucción con sistemas monoculares es su incapacidad de proporcionar la escala real del movimiento de la camara y el entorno observado. Para resolver este problema, se debe obtener información adicional, bien de sensores adicionales, bien conocida *a priori*. Nuestra propuesta en este área es un algoritmo para estimar la escala, y además evitar la deriva en la misma, en un SLAM monocular realizado con una cámara portable, obteniendo la velocidad al andar del usuario a partir de su frecuencia de paso.

- **Odometría densa RGB-D:** Los algoritmos recientes para la estimación de la odometría a partir de cámaras RGB-D estiman el movimiento de la cámara por medio de una minimización pixel a pixel del error fotométrico y geométrico entre dos imágenes. Sin embargo, en muchos casos propiedades importantes del modelo de error del error de profundidad son ignoradas, lo que puede afectar a la precisión del cálculo de la odometría. En esta tesis proponemos un método para odometría con cámaras RGB-D que usa la profundidad inversa, cuya distribución de probabilidad es más cercana al modelo de error del sensor, para la parametrización del error geométrico, mejorando los resultados del estado del arte.

- **Reconocimiento robusto de lugares:** En SLAM visual un módulo de reconocimiento de lugares es un componente clave para la relocalización de la cámara cuando se pierde el seguimiento o para cerrar bucles en lugares revisitados. Sin embargo, en el reconocimiento de lugares se asume frecuentemente que la escena no sufre cambios entre dos visitas, lo cual puede ser fuente de fallos, afectando a la robustez de los algoritmos de reconocimiento de lugares. Nuestro trabajo en este área y en el contexto de sensores RGB-D propone descartar partes de la escena con una entropía alta en las normales de sus superficie para aumentar la robustez del reconocimiento de lugares frente a cambios a largo plazo en la escena.

- **Optimización de grafos de localizaciones:** En diferentes métodos de SLAM, la optimización de grafos de localizaciones es un enfoque usado frecuentemente para imponer restricciones de cerrado de bucle entre pares de localizaciones. Los problemas de grafos de localizaciones suelen mezclar rotación y traslación en la misma función de optimización. Dado que traslación y rotación se miden en unidades diferentes es necesario la normalización de estas variables, lo cual se puede realizar siempre de una manera rigurosa. También, parece innecesario tener que optimizar la orientación junto con la traslación, cuando el objetivo principal es deformar una curva que representa la trayectoria de la cámara. A partir de estas observaciones proponemos una reparametrización del problema de optimización de grafos, eliminado las orientaciones del vector a optimizar y haciendo posible la optimización solo de la posición de la cámara.

Abstract

Under the rapid development of electronics and computer science in the last years, cameras have become omnipresent nowadays, to such extent that almost everybody is able to carry one at all times embedded into their cellular phone. What makes cameras specially appealing for us is their ability to quickly capture a lot of information of the environment encoded in one image or video, allowing us to immortalize special moments in our life or share reliable visual information of the environment with other persons. However, while the task of extracting the information from an image may by trivial for us, in the case of computers complex algorithms with a high computational burden are required to transform a raw image into useful information. In this sense, the same rapid development in computer science that allowed the widespread of cameras has enabled also the possibility of real-time application of previously practically infeasible algorithms.

Among the current fields of research in the computer vision community, this thesis is specially concerned in metric localisation and mapping algorithms. These algorithms are a key component in many practical applications such as robot navigation, augmented reality or reconstructing 3D models of the environment.

The goal of this thesis is to delve into visual localisation and mapping from vision, paying special attention to conventional and unconventional cameras which can be easily worn or handled by a human. In this thesis contribute in the following aspects of the visual odometry and SLAM (Simultaneous Localisation and Mapping) pipeline:

- **Generalised Monocular SLAM:** State of the art visual SLAM algorithms are usually designed for conventional cameras, which can be modelled by a simple pin-hole camera model but have a narrow field of view. Vision systems consisting in a combination of a camera and a conic-shaped mirror, catadioptric cameras, offer a larger field of view, but in turn require of the use of a more complex projection model, which generalises the projection of central projection systems. Our proposal in this field is the adaptation to catadioptric cameras of a real time monocular SLAM system initially designed for conventional cameras.

- **Scale problem in monocular vision:** One problem of localisation and mapping algorithms for monocular systems is their inability to provide the real scale of the camera motion and the observed map. To solve this problem, extra information has to be obtained either from additional sensors or from known or assumed priors about the environment. Our proposal in this field is an algorithm for scale estimation and scale drift avoidance when monocular SLAM is performed with a wearable camera by capturing the walking speed of the user, from the step frequency.

- **Dense RGB-D odometry:** Recent algorithms for odometry estimation with RGB-D sensors, estimate the camera motion by performing pixelwise minimisation of the photometric and/or the geometric error. However, in many cases important properties of the error model of the depth sensor are ignored, which could affect the performance of

odometry computation. In this thesis we propose a direct RGB-D odometry method which uses the inverse depth for the parametrisation of the geometric error, whose probability distribution is closer to the error model of a depth sensor, improving state of the art results.

- **Robust place recognition:** In visual SLAM, a place recognition module is a key component for camera relocalisation when lost, or to close loops at revisited areas. However during place recognition, it is frequently assumed that a scene does not suffer changes between visits, which can be a source of fails, affecting robustness of place recognition algorithms. Our work in this field and in the context of RGB-D sensors proposes the pruning of scene parts with a high entropy in the surface normals in order to increase the robustness of place recognition under long term changes in the scene.

- **Pose-graph optimisation:** In SLAM approaches, pose-graph optimisation is a frequently used approach to enforce loop constraints between pairs of camera poses. Pose-graph problems usually mix the camera rotation and translation in the same optimisation problem. The different units in which rotation and translation are measured make neccesary a normalisation of these variables, but it is not always possible to do this in a rigorous way. In addition to this, it seems also unnecesary having to optimise the orientation together with the translation, when the main objective might be bending the curve which represents the trajectory of the camera. Stemming from these observations we propose a reparametrisation of the pose-graph problem removing the orientations from the optimised vector and making possible the optimisation only on the position part of the poses.

Acknowledgements

First of all I would like to thank my supervisor, Josechu Guerrero, for relying on me and giving me the opportunity of achieving a PhD; and for his continuous confidence, support and guidance during all these years. I would like to thank also Alejandro Rituerto and Luis Puig for guiding and helping me during the first part of my research. Thanks to the Spanish Goverment, Ministerio de Educación, Cultura y Deporte for the FPU scholarship (AP-2012-5507), which allowed the development of this thesis.

I would like to express my sincere gratitude to Walterio Mayol-Cuevas for hosting me at the University of Bristol and for our fruitful collaboration; and to Daniel Cremers for giving me the opportunity to perform a stay in the TUM. Thanks also to all the wonderful people I had the chance to know during my stays in Bristol and Munich, making me feeling like at home and sharing beers, meals and Karaoke Wednesdays.

Many thanks to my lab fellows for creating such a great working atmosphere. Thank you for those hilarious chats at the coffee breaks and the amazing nights of tapas we enjoyed during all these years.

Muchas gracias también a mis amigos fuera del lab por estar siempre allí para lo que sea, y sobre todo muchas gracias a mi familia, a mis padres y a mi hermano, por apoyarme en todo momento y por dármelo todo.

Contents

Chapter 1

Introduction

1.1. Computer Vision and Metric Localisation

The use of cameras on sensing platforms arouses great interest due to the low cost of this kind of sensors and the high amount of information which is encoded in one image. However, while extracting this information might appear as a trivial task for humans, in the case of computer systems a raw image must undergo many processing steps, dependent on the application, to allow the extraction of some meaningful information. In many cases these steps involve costly operations, so the use of images in the fields of computer science and robotics has been intimately linked to the performance of the computer systems. In the recent years, the increasing power and universalization of computers has lead to an extended use of computer vision techniques in robotics and computer science.

In addition to the improvement on CPU performance, the advances in camera miniaturization and mobile computing have lead to an emerging interest in the use of wearable cameras [Mann, 1997], which are now available as consumer products [1] [2]. Although these commercial products are focused on recreational use, wearable cameras can also provide assistance to impaired people. For example, in [Hodges et al., 2011] it is presented a wearable device, the Sense-Cam, which captures images of the wearer's daily life to help people with memory disorders; and in [Mann et al., 2011] authors design a prototype consisting of a vibrotactile helmet with an attached Kinect range camera to allow visually impaired people to avoid obstacles.

In a platform including a vision system, a precise odometric localisation of the camera and representation of the environment is an important step, prior to the performance of higher level tasks. In robotics this problem is known as Simultaneous Localisation and Mapping (SLAM) and its resolution has many practical applications such as augmented reality [Klein and Murray, 2007], smart navigation for robots, consistent 3D reconstructions of cities from images [Agarwal et al., 2009], or dense 3D reconstructions of indoor environments [Whelan et al., 2015].

For such localisation and navigation purposes, panoramic cameras with an horizontal field of view of 360º provide a better performance than conventional ones. The immediate benefit of using panoramic cameras is the passive perception of most of the environment independently of the camera orientation. Among panoramic cameras, catadioptric cameras, which consist in one camera-mirror system, offer more compactness, simplicity and a lower cost than multicamera systems. However catadioptric images capture the environment with a severe deformation and provide a coarsest resolution. Furthermore the resolution varies with the distance to the principal point. This makes that many computer vision techniques must be adapted to this kind of sensors.

[1] http://gopro.com
[2] http://memoto.com

We have referred to the problem of localisation and scene reconstruction as SLAM. However, although the problem to be solved is essentially the same, its name actually depends on the perspective from which it is addressed. In the computer vision literature it is referred to as Structure from Motion (SfM), while the term SLAM is applied in the field of robotics. In Structure from Motion the stress is put into reconstructing a 3D scene from a sparse set of images, for which batch non-linear optimization techniques known as Bundle Adjustment (BA) are used [Triggs et al., 2000].

On the other hand, Visual SLAM techniques focus in estimating the camera motion in real time from images taken at video rate. Traditionally, the V-SLAM problem has been addressed from a bayesian filtering perspective either by using an Extended Kalman Filter [Davison, 2003] or particle filters [Montemerlo et al., 2002]. Probabilistic filters process images sequentially, keeping track of the map and the last camera pose. Previous poses and measurements are marginalised out by updating an uncertainty matrix which spans over the current pose and the map features. This marginalisation step of previous poses allows for real-time operation at the expense of getting lower precision and establishing strict limits on the map size due to the cost of the matrix update, which is quadratic with the state vector size.

However, later approaches like the iSAM [Kaess et al., 2008] of Kaess et al. and the PTAM [Klein and Murray, 2007] of Klein and Murray have showed successful solutions for the real time SLAM from an optimization-based perspective, which is closer to a Bundle Adjustment problem. In [Kaess et al., 2008], a reordering of the information matrix allows for an efficient resolution of SLAM posed as a least squares problem. In [Klein and Murray, 2007], authors get rid of the uncertainty matrix and its limitations by separating the camera tracking and the environment mapping. Camera motion relative to a fixed map is estimated on-line in one thread, while another thread updates this map periodically by applying Bundle Adjustment on selected keyframes.

Following the success of the new optimization-based SLAM approaches, in [Strasdat et al., 2010] Strasdat et al. compare the filtering vs optimization paradigms, and conclude that Bundle Adjustment optimisation generally outperforms filtering techniques. In [Kümmerle et al., 2011] Kümmerle et al. presented the g2o, a state-of-the-art C++ framework for graph optimization which aims to provide high scalability for several variants of SLAM and BA.

The probabilistic nature of SLAM and the incremental estimation of the camera motion, lead to an unavoidable error build-up. The accumulated error gives raise to a drift in the trajectory, which becomes evident when the sensor platform revisits a previous location. It is expressly in these situations when the so called loop closing techniques can be applied to correct the drift.

The loop closure process can be divided into three steps: loop detection, computation of the loop closing constraint and trajectory correction with the new constraints. The detection of loops is performed by attempting match the current frame against a database of keyframes. These keyframes are selected such that they are spatially far enough from temporally close keyframes but still share most of the captured environment. State of the Art approaches for loop detection [Cummins and Newman, 2011], [Gálvez-López and Tardos, 2012], use a technique based on Bag of Words. It consists in extracting local features in the keyframe and classifying these features into words using a vocabulary which has been previously constructed offline. Finally an histogram with the frequency in the keyframe of each vocabulary word is built. When querying the database of keyframes, those keyframes with histograms similar to the current frame are selected as keyframe candidates.

After a set of possible candidates has been selected, the final phase of loop detection comes which is coupled with the computation of the loop constraint. Given the correspondences between local features in both candidate and current frames, a putative motion estimate is

computed using algorithms for multiple view geometry for each candidate frame match. This is done within a RANSAC scheme, such that a loop is detected if there are enough correspondences supporting the best motion estimate. This estimate can be then further refined by non linear optimisation with the inlier matches.

The last step involves the correction of the trajectory and map by enforcing the new loop constraints. This is done by optimising a pose-graph where the final trajectory is the one which minimises the combined cost of violating the initial odometry constraints from the SLAM front-end and the new loop closure constraints.

One important limitation of visual SLAM techniques is that monocular vision systems are not able to provide depth measurements of the observed landmarks. One would need two images, provided that the baseline between the corresponding camera poses is large enough, to estimate both the camera translation and the depth of the observed landmarks up to an ambiguity in the scale factor.

In the context of monocular SLAM, this limitation translates in two closely related problems. The first one is a chicken-egg problem during initialisation by which neither camera translation nor the depth of the landmarks can be estimated until enough parallax is observed. This can be tackled by direct intervention of the user providing some initial landmarks whose 3D position is fully known [Davison, 2003], or manually selecting two initial keyframes with enough parallax [Klein and Murray, 2007]. However, an smarter automatic initialisation is possible using a specialised initialisation algorithms [Mur-Artal et al., 2015] or even without need of applying a specific algorithm, if we choose an appropriate parameterisation for the landmarks [Civera et al., 2008].

The second one is known as scale drift. During the initialisation of monocular SLAM, the scale is initially fixed to a real or an arbitrary value, depending of the used method of the mentioned before. However, far from keeping itself constant, the continuous lost of old landmarks and the initialisation of new ones gives raise to a change of the scale of the scene as the camera moves. This ends up by introducing a deformation in the final trajectory and map estimates which goes beyond a simple scale ambiguity. The deformation introduced by the drift in the scale can even persist after applying state-of-the-art loop closure techniques. To correct the scale drift one typically needs to track the scale using information from additional sensors or from prior information, though it is also possible to correct it at loop closure using similarity transformations to express the motion between camera poses [Strasdat et al., 2010].

In computer vision, the straightforward solution to the scale problem is using stereo vision systems [Nistér et al., 2006, Paz et al., 2008, Mei et al., 2010] where the known fixed baseline between two cameras allows to estimate the depth of the pixels in the image. However with respect to monocular systems they are much more expensive, bigger and more difficult to calibrate, and also they cannot accurately measure the depth of distant scene points or poorly textured areas.

For this reason the recent advent of new RGB-D sensors has aroused great interest in the development of visual odometry and SLAM systems. Their cheapness and their ability to provide dense depth measurements of the environment in contrast to traditional stereo cameras makes them quite appealing to address not only localisation and mapping but also many other problems for which monocular systems are typically used. The main limitation is their use being limited to indoor environments.

1.2. Goals and contributions

The main goals and contributions in this PhD Thesis can be grouped in the following related topics:

- **Monocular SLAM:** We first propose, an adaptation of a real-time monocular SLAM application from conventional to catadioptric systems. Secondly we propose a new algorithm for scale estimation and scale drift avoidance in monocular SLAM running on wearable cameras.
- **RGB-D Localisation and Dense Mapping:** We contribute with a dense RGB-D odometry method which as a novelty uses the inverse depth instead of depth to parametrise the geometric error. We also propose the pruning of high surface normal entropy in a dense RGB-D map in order to potentially improve place recognition with RGB-D cameras.
- **Pose-graph optimisation:** We propose a novel redefinition of the pose-graph constraints in order to allow for the resolution of pose-graph optimisation problems dropping the orientation part of the poses and optimising only the position.

1.2.1. Monocular SLAM

The first part of this thesis turns around visual SLAM with a single camera, or monocular SLAM. Much of the work in this part has been done over the MonoSLAM application developed initially in [Davison, 2003] and further improved in the works [Civera et al., 2008], [Civera et al., 2010]. Our contributions in the field of monocular SLAM are described in Chapter 2 and Chapter 3.

Generalization of Monocular SLAM for central projection systems: Chapter 2 describes the generalisation of a monocular SLAM initially designed only for conventional cameras, in order to make it work with every kind of central projection system, which include not only conventional cameras but also catadioptric systems which are those formed by a mirror and a camera. Since catadioptric systems offer a more extended field of view with respect to conventional ones, a visual SLAM system which is able to work with such cameras is likely to have better precision and produce richer reconstructions of the environment. The paper associated to this contribution is:

- [Gutiérrez et al., 2011] D. Gutiérrez, A. Rituerto, J.M.M. Montiel, J.J. Guerrero, "Adapting a Real-Time Monocular SLAM from Conventional to Omnidirectional Cameras", In *11th OMNIVIS Omnidirectional Vision, Camera Networks and Non-classical Cameras, IEEE International Conference on Computer Vision Workshops*, pp. 343-350, 2011.

Scaled Monocular SLAM for Wearable Systems: In monocular SLAM, the reconstruction of the 3D map and the trajectory of the camera can only be estimated up to one scale factor. Furthermore the scale factor is not unique for the whole reconstruction, being subject to a drift which ultimately produces severe deformations in the reconstruction. To address this problem in Chapter 3 we present a novel method to compute the scale for monocular SLAM running on wearable systems. Our method in essence obtains the walking speed of the user, from the step frequency, based on a biological model of people walking which is computed by analysing the vertical movement of the camera in a temporal window. Knowing the walking speed we are able to obtain a scale factor for the frames contained in that temporal

window. The experiments, carried out in outdoor and indoor environments and with different types of cameras, show that our method is reliable and robust to challenging situations like stops, changes in pace or stairs, and provides a significant improvement with respect to the initial unscaled estimate. The papers associated to this contribution are:

- [Gutiérrez-Gómez et al., 2012] D. Gutiérrez-Gómez, L. Puig, J.J. Guerrero. "Full Scaled 3D Visual Odometry from a Single Wearable Omnidirectional Camera", In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4276 - 4281, 2012.
- [Gutiérrez-Gómez and Guerrero, 2013] D. Gutiérrez-Gómez, J.J. Guerrero. "Scaled Monocular SLAM for Walking People", In *ACM International Symposium on Wearable Computers (ISWC)*, pp. 9 - 12, 2013.
- [Gutiérrez-Gómez and Guerrero, 2016] D. Gutiérrez-Gómez, J.J. Guerrero. "True Scaled 6 DoF Egocentric Localisation", *Journal paper under review*

1.2.2. RGB-D Localisation and Dense Mapping

In the second part of the thesis the work was focused towards RGB-D sensors.

Dense RGB-D visual odometry using inverse depth: With the advent of RGB-D sensors approaches for dense odometry estimation, *i.e.*, using all the pixels in the image rather than local features sparsely distributed along the image became increasingly popular. Though dense methods were previously applied on RGB sensors, the lack of depth measurements made the problem being solved ill-posed and thus requiring from costly and complex variational methods for the resolution. The availability of depth, greatly simplified the problem and dense visual odometry methods for RGB-D cameras started to proliferate. However, all of them have in common the usage of depth for the resolution of the problem, which in principle is not well suited to the noise model of the depth measurements in RGB-D sensors. Our contribution presented in Chapter 4 proposes a new visual odometry method computing the inverse depth, which fits better with the noise model of cameras than using standard depth. The method also summarises most of the best practices found in the literature, which makes the algorithm robust and non-dependent on heuristic or tunable parameters. We report a high accuracy of our method with superior results to other state of the art RGB-D odometry approaches in real world as well as in synthetic datasets. We have implemented this method within the scope of PCL (Point Cloud Library) as a branch of the code for large scale KinectFusion, where the original ICP system for odometry estimation has been completely substituted by our method. A PCL fork including the modified method is available for download .The papers associated to this contribution are:

- [Gutiérrez-Gómez et al., 2015a] D. Gutiérrez-Gómez, W. Mayol-Cuevas, J.J. Guerrero. "Inverse Depth for Accurate Photometric and Geometric Error Minimisation in RGB-D Dense Visual Odometry", In *International Conference on Robotics and Automation (ICRA)*, 2015. Finalist to the Best Robotic Vision paper Award and Best Conference Paper Award.
- [Gutiérrez-Gómez et al., 2016] D. Gutiérrez-Gómez, W. Mayol-Cuevas, J.J. Guerrero. "Dense RGB-D Visual Odometry using Inverse Depth".*Robotics and Autonomous Systems (RAS)*, 2016.

Segmentation in superjuts for robust place recognition: One open problem in the fields of place recognition and mapping is to be able to recognise a revisited place when its appearance and layout have changed between visits. In Chapter 5 we investigate this problem in the context of RGB-D mapping in indoor environments. We propose to segment the scene in *juts* (neighbourhood of 3D points with normals that stick out from the surroundings) and look at low-level features, like textureness or entropy of the normals. The key idea is that these kind of properties can help to differentiate zones of the scene which change or move along time from those that are likely to remain static. We also present a method which improves the matching between images of the same place taken at different times by pruning details basing on these features. We evaluate on a number of communal areas and also on some scenes captured 6 months apart. Experiments with our approach, show an increase up to 70% in inlier matching ratio at the cost of pruning only less than 20% of correct matches, without the need of performing geometric verification.The papers associated to this contribution are:

- [Gutiérrez-Gómez et al., 2015b] D. Gutiérrez-Gómez, W. Mayol-Cuevas, J.J. Guerrero. "What Should I Landmark? Entropy of Normals in Depth Juts for Place Recognition in Changing Environments Using RGB-D Data", *In International Conference on Robotics and Automation (ICRA)*, 2015.

Robust RGB-ID SLAM in Changing Environments: Chapter 6 brings together the contributions of the two previous chapters, to build a robust RGB-ID SLAM system, ID for Inverse Depth, which is able to relocate at scenes which have suffered changes between different sessions. Our system consists in 2 threads working in parallel. The first thread is a front-end operating at frame rate, which processes every incoming frame from the RGB-D sensor to compute the incremental odometry and integrate it in a keyframe which is changed periodically following a covisibility-based strategy. The second thread is a back-end which receives keyframes from the front-end. This thread is in charge of segmenting the keyframes based on their structure, describing them using Bags of Words, trying to find potential loop closures with previous keyframes, and in such case perform pose-graph optimisation for trajectory correction. The first experiments with our approach in the TUM RGB-D benchmark datasets show results superior in accuracy to the state-of-the-art in many of the sequences. These promising initial evaluation encourages us to further improve our method.

1.2.3. Pose-graph optimisation

The last part of this thesis is concerned with the pose-graph optimisation problem which in visual SLAM frequently arises when loops are closed. During loop closure, pose-graph optimisation is the final step where the camera trajectory and map are bended so that they satisfy relocalisation constraints of a camera in a previously visited area. Our contribution in this area is covered in Chapter 7.

Removal of orientation in pose-graph problems: We propose a reparametrisation of the pose-graph optimisation, where a graph containing only nodes with poses of a rigid body is optimised such that these poses satisfy new constraints. This kind of problem arises frequently when trying to close a loop in a trajectory. This trajectory, initially computed only from odometry constraints has to be bended such that it satisfies new loop constraints obtained when old zones of the environment are revisited. The usual practice is to establish these constraints both in the position and orientation of the body. We propose to reformulate the optimisation problem in order to drop the orientation part of the poses. Instead of treating

trajectory as a set of poses, we look at it as a curve in its pure mathematical meaning. We define an observation function which computes the estimate of one constraint in a local reference frame using only the robot positions. Our proposed method is compared against state-of-the-art pose graph optimisation algorithms in 2 and 3 dimensions.The main benefits of doing this is *i*) a reduction of the number of variables to be optimised and *ii*) avoid the mixture of errors of different magnitudes, which is specially important in the case that properly computed information matrices cannot be provided. The papers associated to this contribution are:

- [Gutiérrez-Gómez and Guerrero, 2014] D. Gutiérrez-Gómez, J.J. Guerrero. "Curve-Graph odometry: Removing the orientation in loop closure optimisation problems", In *International Conference on Intelligent Autonomous Systems (IAS)*, 2014. Best Student Paper Award in International Conference on Intelligent Autonomous Systems (IAS).
- [Gutiérrez-Gómez and Guerrero, 2015] D. Gutiérrez-Gómez, J.J. Guerrero. "Curve-graph odometry: Orientation-free error parametrisations for loop closure problems", *Robotics and Autonomous Systems (RAS)*, 2015.

I have also collaborated with other researchers of our group in the integration of developed algorithms in systems with a higher complexity, in the context of the VINEA research project:

- [Pérez-Yus et al., 2016] A. Pérez-Yus, D. Gutiérrez-Gómez, G. López-Nicolas, J. J. Guerrero "Stairs Detection with Odometry-aided Traversal From a Wearable RGB-D Camera". *Under Review Journal Paper.*
- [Guerrero et al., 2015] J. J. Guerrero, A. Pérez-Yus, D. Gutiérrez-Gómez, A. Rituerto, G. López-Nicolás. "Human navigation assistance with a RGB-D sensor". In *VI Congreso Internacional de Diseño, Redes de Investigación y Tecnología para todos (DRT4ALL)*, pp. 285-312, 2015.
- [Murillo et al., 2012] A. C. Murillo, D. Gutiérrez-Gómez, A. Rituerto, L. Puig and J. J. Guerrero. "Wearable Omnidirectional Vision System for Personal Localization and Guidance", In *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW) 2nd Workshop on Egocentric Vision*, pp. 8-14, 2012.

Chapter 2

Real-Time Generalized Monocular SLAM for Central Cameras

The SLAM (Simultaneous Localization and Mapping) problem is one of the essential challenges for the current robotics. The main objective in this chapter is to develop a real-time visual SLAM system using monocular omnidirectional vision. Our approach builds on a Visual SLAM system based on the Extended Kalman Filter (EKF). To obtain geometric information from the images, instead of the typical pin-hole camera model with distortion parameters, we use a generalised model which scopes not only conventional cameras but also any central projection system like the catadioptric omnidirectional cameras used in this work. This model is integrated in the EKF-based SLAM through the linearization of the direct and the inverse projections. We introduce an affine transformation to warp the descriptor patch for catadioptric omnidirectional cameras which aims to reach rotation and scale invariance. We perform experiments with omnidirectional images comparing this new approach with the conventional one. The experimentation confirms that our approach works better with omnidirectional cameras since features have a larger lifespan and constructed maps are bigger.

2.1. Introduction

Solving the SLAM problem implies building a map of the surrounding and localising an autonomous robot relative to this map using only partial measurements of the environment. SLAM is usually formulated in a probabilistic way, *i.e.*the estimate of the robot position and map are computed as a probability distribution. Two main approaches are used for the computation of the probability distribution: the extended Kalman filter (EKF) [Thrun et al., 2005] and the particle filter [Arulampalam et al., 2002].

In Visual SLAM applications, image projections of relevant points known as local features are used as measurements. To extract and store the features on the image an extractor and descriptor are used. The feature extractor process the image and detects the key-points on it. The image processing is a high time-consuming step, which is critical for a real time application like SLAM. Rosten *et al.* [Rosten et al., 2010] developed the feature extraction algorithm FAST (Features Accelerated Segment Test). They benchmark their FAST extractor with other widely used feature extractors showing that FAST outperforms them in computational cost and in repeatability when viewing the scene from different positions. The descriptor provides an identifier to an extracted point so that it can be recognised in future measurements. The

most basic descriptor is a patch of a certain size centred in the key-point, although there exists more kinds of descriptors like SIFT [Lowe, 2004], SURF [Bay et al., 2008], LBP [Heikkila et al., 2009], etc.

Since the seminal work of Davison [Davison, 2003], monocular SLAM has been a fertile research field. In this work we propose to combine state of the art robust EKF SLAM [Civera et al., 2010] with an omnidirectional sensor. We integrate the Spherical Camera Model in a Real Time application. The main differences with the work developed in [Rituerto et al., 2010] is that now we use image patches instead of SIFT descriptors and our solution includes robust detection of spurious, operating at video sampling rate. Besides that we develop an affine transformation for patch warping in catadioptric cameras which considers rotation and scale invariance in function of mirror parameters. To reach rotation invariance we base on the proposal of [Andreasson et al., 2007]. For scale invariance we develop a formulation of the scale factor in function of the mirror parameters which can be applied in any kind of central camera and, in particular, in a hyper-catadioptric system compound by a hyperbolic mirror coupled with a perspective camera.

2.2. Related Work

Visual SLAM using omnidirectional cameras has been proposed in [Corke et al., 2004], [Mei, 2007], [Tardif et al., 2008] and [Scaramuzza et al., 2009b]. The main advantage of using omnidirectional cameras in this context is that their large field of view, allows to track features during longer times than in the case of conventional cameras, specially when big camera rotations occur. The increased lifespan of the features on the image translates in a better estimation of the position of the features on the map, a lower need to initialise new features and a increased robustness.

However the omnidirectional images involve a more complex projection model, important image deformation, distortion and variable scale in the image. So, the feature descriptor should be modified for catadioptric cameras. In this way, [Svoboda and Pajdla, 2001] propose the use of patches with variable size and shape (active windows). Their experiments show that active windows provide best matching results than square windows. [Chang et al., 2009] propose the computation of patches of different angular apertures for the same feature to overcome the matching problems derived from the varying resolution of the camera. [Scaramuzza et al., 2007] take advantage of the projection of vertical lines of the world as radial lines on the image. They propose a method to extract and match vertical lines with rotation invariant descriptors and apply this method to an EKF-SLAM. [Andreasson et al., 2007] propose a modified SIFT feature with no scale invariance. To obtain rotation invariance they rotate each patch to the same global orientation. [Lu and Zheng, 2010] combine the rotation invariant patch by Andreasson with a FAST extractor and a CS-LBP descriptor and they compare it with the SIFT algorithm.

2.3. The Spherical Camera Model

Central catadioptric systems are those composed by a conventional camera and a planar or conic-shaped mirror, and have the characteristic of having a unique projection center. The projection of 3D points in systems with a unique projection center, which are not only catadioptric ones, but also conventional and fish-eye cameras can be described by a generalised projection model proposed by Geyer and Daniilidis [Geyer and Daniilidis, 2000] and extended Barreto and Araujo [Barreto and Araujo, 2001].

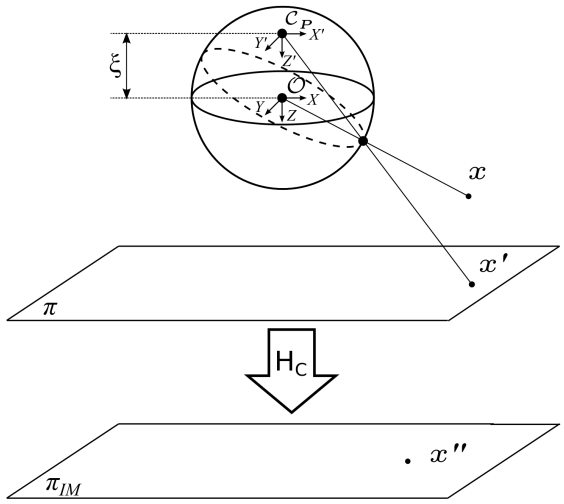


Figure 2.1: Scheme of the sphere projection model.

Taking a 3D point expressed in the camera reference frame with euclidean coordinates $\mathbf{x} = (x, y, z)$, its projection on the image is modeled by a following projection function $\mathbf{p} = \mathbf{\Pi}(\mathbf{x})$, which is invariant to the camera-point distance, *i.e.*, $\mathbf{\Pi}(\mathbf{x}) = \mathbf{\Pi}(k\mathbf{x}) \, \forall k \in \mathbb{R}+$. This allows to reinterpret $\mathbf{x}$ more generally as a projective ray where the projected 3D point lies. With this in mind, the projection can be divided in the following steps (see Fig. 2.1):

- The ray $\mathbf{x}$ is projected onto the unit sphere centred in the origin $\mathbf{O}$. The intersection point is projected to a virtual projection plane π through the virtual projection center $\mathbf{C_P} = (0, 0, -\xi)^\mathsf{T}$ yielding the point $\mathbf{x}'$. This step is coded by the non-linear function $\hbar$:

$$\mathbf{x}' = \hbar(\xi, \mathbf{x}) = \mathbf{x} + \begin{pmatrix} 0 \\ 0 \\ \xi||\mathbf{x}|| \end{pmatrix} \tag{2.1}$$

- The virtual plane π is transformed in the image plane π_{IM} through a homographic transformation $\mathbf{H_c}$

$$\mathbf{x}'' = \mathbf{H_c}\mathbf{x}' \tag{2.2}$$

$$\mathbf{H_c} = \mathbf{K_c}\mathbf{R_c}\mathbf{M_c} \tag{2.3}$$

where $\mathbf{K_c}$ includes the camera parameters, $\mathbf{M_c}$ includes the mirror parameters and $\mathbf{R_c}$ is the rotation matrix between camera and mirror. Assuming a pin-hole camera model and $\mathbf{R_c} = \mathbf{I}$, the transformation $\mathbf{H_c}$ yields:

$$\mathbf{H_c} = \begin{bmatrix} \eta f & 0 & u_0 \\ 0 & \eta f & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \gamma & 0 & u_0 \\ 0 & \gamma & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.4}$$

where $\gamma = \eta f$ is the generalized focal length of the camera-mirror system with η a mirror parameter and f the focal length of the camera.

- Finally the image coordinates are calculated by dividing $\mathbf{x}''$ by its z'' coordinate:

$$\mathbf{p} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \frac{\mathbf{x}''}{\mathbf{e_z}^T \mathbf{x}''}, \tag{2.5}$$

where $\mathbf{e_z}^T = (0,\ 0,\ 1)$.

The parameter of the model, ξ depends only on the kind of camera and the geometry of the mirror. For conventional cameras $\xi = 0$. $\xi = 1$ for catadioptric systems with parabolic mirror and orthographic camera, and $0 < \xi < 1$ with hyperbolic mirror and perspective camera.

It is also possible to estimate the 3D ray which is projected on the image, through the inverse of the projection model. Starting with the point in image coordinates $\mathbf{p} = (u,\ v,\ 1)^T$, we obtain the corresponding ray as follows:

$$\mathbf{x}' = {\mathbf{H_c}}^{-1}\mathbf{p} \tag{2.6}$$

$$\mathbf{x} = \hbar^{-1}(\xi, \mathbf{x}') = \mathbf{x}' - \begin{pmatrix} 0 \\ 0 \\ \frac{\xi||\mathbf{x}'||}{\xi \mathbf{e_z}^T \mathbf{x}' + \sqrt{(1-\xi^2)||\mathbf{x}'||^2 + \xi^2 (\mathbf{e_z}^T \mathbf{x}')^2}} \end{pmatrix}. \tag{2.7}$$

2.3.1. Jacobians of the Spherical Camera Model

In some cases, as occurs in optimisation, probabilistic filtering or uncertainty propagation, we need to compute the Jacobians of the projection model. In this section we provide the analytical expressions for those Jacobians, as obtained in [Rituerto et al., 2010].

For the projection model the Jacobian is given by:

$$\frac{\partial \mathbf{p}}{\partial \mathbf{x}} = \frac{\partial \mathbf{p}}{\partial \mathbf{x}''} \frac{\partial \mathbf{x}''}{\partial \mathbf{x}'} \frac{\partial \mathbf{x}'}{\partial \mathbf{x}} \tag{2.8}$$

$$\frac{\partial \mathbf{p}}{\partial \mathbf{x}''} = \begin{bmatrix} \frac{1}{z''} & 0 & -\frac{x''}{z''^2} \\ 0 & \frac{1}{z''} & -\frac{y''}{z''^2} \end{bmatrix} \tag{2.9}$$

$$\frac{\partial \mathbf{x}''}{\partial \mathbf{x}'} = \mathbf{H_c} \tag{2.10}$$

$$\frac{\partial \mathbf{x}'}{\partial \mathbf{x}} = \frac{\partial \hbar(\mathbf{x})}{\partial \mathbf{x}} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{\xi x}{\rho} & \frac{\xi y}{\rho} & 1 + \frac{\xi z}{\rho} \end{bmatrix}, \tag{2.11}$$

where $\rho = \sqrt{x^2 + y^2 + z^2}$

The Jacobian of the inverse projection is computed as:

$$\frac{\partial \mathbf{x}}{\partial \mathbf{p}} = \frac{\partial \mathbf{x}}{\partial \mathbf{x}'}\frac{\partial \mathbf{x}'}{\partial \mathbf{p}}, \tag{2.12}$$

$$\frac{\partial \mathbf{x}'}{\partial \mathbf{p}} = \mathbf{H}_{\mathbf{c}}^{-1}, \tag{2.13}$$

$$\frac{\partial \mathbf{x}}{\partial \mathbf{x}'} = \frac{\partial \bar{h}^{-1}(\mathbf{x}')}{\partial \mathbf{x}'} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{\xi x'}{\chi} & -\frac{\xi y'}{\chi} & 1 - \frac{\xi(z' - \xi\frac{x'^2+y'^2+z'^2}{\xi z'+\chi})}{\chi} \end{bmatrix}, \tag{2.14}$$

where $\chi = \sqrt{(1-\xi^2)(x'^2+y'^2)+z'^2}$

2.4. EKF-based Simultaneous Localisation And Mapping

The camera state as well as the position of the landmarks at time step i are encapsulated in a state vector $\mathbf{x}_i$

$$\mathbf{x}_i = (\underbrace{\mathbf{r}_{W,i}^C,\ {}_W\mathbf{R}_i^C,\ \mathbf{v}_{W,i}^C,\ \omega_{C,i}^C}_{\text{Camera state } \mathbf{x}_i^C},\ \underbrace{\mathbf{r}_{W,i}^{C(j)}, \theta_i^{(j)}, \phi_i^{(j)}, \rho_i^{(j)}, ...}_{\text{3D points (IDP) } \mathbf{y}_{IDP,W}^{(j)}}), \tag{2.15}$$

where $\mathbf{r}_{W,i}^C$ is the camera position, ${}_W\mathbf{R}_i^C$ is the rotation matrix representing its orientation and $\mathbf{v}_{W,i}^C$ and $\omega_{C,i}^C$ are its linear and angular velocities, respectively.

The 3D locations are parametrised in an anchored inverse depth parametrisation (IDP) [Civera et al., 2008], where $\mathbf{r}_{W,i}^{C(j)}$ is the anchor camera position where the landmark was first viewed, $\theta_i^{(j)}$, $\phi_i^{(j)}$ and $\rho_i^{(j)}$ are respectively the elevation angle, the azimuth angle and the inverse depth with respect to the anchor position.

Since the EKF produces a probabilistic estimate of the state we need to define also the covariance $\mathbf{P}_i$ corresponding to the state vector $\mathbf{x}_i$. We start by defining an error vector:

$$\boldsymbol{\delta}\mathbf{x}_i = (\underbrace{\boldsymbol{\delta}\mathbf{r}_{W,i}^C,\ \boldsymbol{\delta}\boldsymbol{\theta}_{W,i}^C,\ \boldsymbol{\delta}\mathbf{v}_{W,i}^C,\ \boldsymbol{\delta}\omega_{C,i}^C}_{\text{Camera state } \boldsymbol{\delta}\mathbf{x}_i^C},\ \underbrace{\boldsymbol{\delta}\mathbf{r}_{W,i}^{C(j)}, \delta\theta_i^{(j)}, \delta\phi_i^{(j)}, \delta\rho_i^{(j)}, ...}_{\text{3D points (IDP) } \boldsymbol{\delta}\mathbf{y}_{IDP,W}^{(j)}}), \tag{2.16}$$

from which we obtain the expression for the covariance

$$\mathbf{P}_i = \begin{bmatrix} \mathbf{P}_i^{CC} & \mathbf{P}_i^{CM} \\ \left(\mathbf{P}_i^{CM}\right)^T & \mathbf{P}_i^{MM} \end{bmatrix} = E\left[\boldsymbol{\delta}\mathbf{x}_i \boldsymbol{\delta}\mathbf{x}_i^T\right] \tag{2.17}$$

Generally the error for each component of the state vector is obtained by modeling it as an additive perturbation on the true quantity to be estimated ($\hat{x} = x + \delta x$). In the case of the rotation however a direct addition on a rotation matrix (${}_W\hat{\mathbf{R}}_i^C = {}_W\mathbf{R}_i^C + {}_W\boldsymbol{\delta}\mathbf{R}_i^C$) implies a direct violation of the constraints induced by the manifold of 3D rotations $SO(3)$. To preserve these constraints, errors on the true rotation must be modeled in the corresponding Lie Algebra, $\boldsymbol{\delta}\boldsymbol{\theta}_{W,i}^C \in so(3)$, using a minimal parametrisation with 3 DoF, then mapping it to the Lie Group through the exponential map, and applying the perturbation by the composition operation, *i.e.*,

$$ {}_W\hat{\mathbf{R}}_i^C = \exp([\boldsymbol{\delta\theta}_{W,i}^C]_\times)\, {}_W\mathbf{R}_i^C, \tag{2.18} $$

where the notation $[\mathbf{v}]_\times$ is used to denote the 3x3 skew-symmetric matrix obtained from a 3-dimensional vector $\mathbf{v}$.

2.4.1. Motion model

In an EKF based SLAM a motion and a measurement model must be provided. The motion model describes the change on the camera pose from time step $i-1$ to i and it is described by the following equation:

$$ \mathbf{x}_i^C = \mathbf{f}(\mathbf{x}_{i-1}^C, \mathbf{u}_i), \tag{2.19} $$

where $\mathbf{f}(\cdot)$ is the state transition function , $\mathbf{x}_{i-1}^C$ is the last estimated camera pose and $\mathbf{u}_i$ is the control input.

This model is used to propagate the camera state estimate $\mathbf{x}_i^C$ and its covariance $\mathbf{P}_i^{CC}$ in the EKF prediction step. Internal measurements from an odometer or an IMU can be integrated as control inputs $\mathbf{u}_i$. Alternatively, in absence of these measures, a constant velocity model is frequently used to propagate the state. Possible changes in velocity are dealt with by modelling the control input as linear and angular accelerations with zero mean and a variance which has to be tuned:

$$ \hat{\mathbf{u}}_i = \begin{pmatrix} \mathbf{n}_{a_C} \\ \mathbf{n}_{\alpha_C} \end{pmatrix}, \mathbf{n}_{a_C} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma_a}), \mathbf{n}_{\alpha_C} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma_\alpha}) \tag{2.20} $$

To obtain the transition function we need to integrate the differential equations governing the dynamics of a rigid body moving in the 3D space with random perturbations in acceleration:

$$ \dot{\mathbf{r}}_W^C = \mathbf{v}_W^C \tag{2.21} $$

$$ {}_W\dot{\mathbf{R}}^C = {}_W\mathbf{R}^C[\boldsymbol{\omega}_C^C]_\times \tag{2.22} $$

$$ \dot{\mathbf{v}}_W^C = {}_W\mathbf{R}^C\mathbf{n}_{a_C} \tag{2.23} $$

$$ \dot{\boldsymbol{\omega}}_C^C = \mathbf{n}_{\alpha_C} \tag{2.24} $$

Taking $\mathbf{n}_{a_C} = \mathbf{n}_{\alpha_C} = \mathbf{0}$, the transition function of the camera state is:

$$ \mathbf{r}_{W,i}^C = \mathbf{r}_{W,i-1}^C + \mathbf{v}_{W,i-1}^C \Delta t \tag{2.25} $$

$$ {}_W\mathbf{R}_i^C = {}_W\mathbf{R}_{i-1}^C \mathbf{R}(\boldsymbol{\omega}_{C,i-1}^C \Delta t) \tag{2.26} $$

$$ \mathbf{v}_{W,i}^C = \mathbf{v}_{W,i-1}^C \tag{2.27} $$

$$ \boldsymbol{\omega}_{C,i}^C = \boldsymbol{\omega}_{C,i-1}^C \tag{2.28} $$

To propagate the covariance we have to integrate the differential equations of the error model. The obtention and the integration of the continuous error model is detailed in Appendix A. At

the end we obtain a discrete time model for the error propagation from which we obtain the following expression for the propagation of the state covariance.

$$\mathbf{F}_i = \frac{\partial \delta \mathbf{x}_i^C}{\partial \delta \mathbf{x}_{i-1}^C} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{I}\Delta t & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & {}_W\mathbf{R}_{i-1}^C \mathbf{Q}(\boldsymbol{\omega}_{C,i}^C \Delta t)\Delta t \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{2.29}$$

$$\mathbf{G}_i = \frac{\partial \delta \mathbf{x}_i^C}{\partial \delta \mathbf{u}_{i-1}} = \begin{bmatrix} {}_W\mathbf{R}_{i-1}^C \mathbf{S}(\boldsymbol{\omega}_{C,i-1}^C \Delta t)\Delta t^2 & \mathbf{0} \\ 0 & {}_W\mathbf{R}_{i-1}^C \left(\mathbf{Q}(\boldsymbol{\omega}_{C,i-1}^C \Delta t) - \mathbf{S}(\boldsymbol{\omega}_{C,i-1}^C \Delta t)\right) \Delta t^2 \\ {}_W\mathbf{R}_{i-1}^C \mathbf{Q}(\boldsymbol{\omega}_{C,i-1}^C \Delta t)\Delta t & \mathbf{0} \\ \mathbf{0} & \mathbf{I}\Delta t \end{bmatrix} \tag{2.30}$$

$$\mathbf{P}_i^{CC} = \mathbf{F}_i \mathbf{P}_{i-1}^{CC} \mathbf{F}_i^T + \mathbf{G}_i \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{a}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{\boldsymbol{\alpha}} \end{bmatrix} \mathbf{G}_i^T \tag{2.31}$$

$$\mathbf{P}_i^{CM} = \mathbf{F}_i \mathbf{P}_{i-1}^{CM} \tag{2.32}$$

2.4.2. Measurement model

The measurement model is used to introduce the information from measurements of external sensors in the EKF update step. In monocular SLAM it is defined by an observation function which encapsulates a projectivity transformation. The observation function depends on the characteristics of the vision system. For central projection systems, i.e., planar (conventional), dioptric or catadioptric systems, we use the generalised model described in Section 2.3, taking $\mathbf{h}_{C,i}^{(j)}$ as the ray corresponding to the $j-th$ landmark, whose projection is predicted at the location $\mathbf{z}_i^{(j)}$ in the image plane. This is encoded in the following equations:

$$\mathbf{z}_i^{(j)} = \boldsymbol{\Pi}\left(\mathbf{h}_{C,i}^{(j)}\right) = \mathbf{H}_c \hbar\left(\xi, \mathbf{h}_{C,i}^{(j)}\right) \tag{2.33}$$

$$\mathbf{h}_{C,i}^{(j)} = \left({}_W\mathbf{R}_i^C\right)^T \left(\rho_i^{(j)} \left(\mathbf{r}_{W,i}^{C(j)} - \mathbf{r}_{W,i}^C\right) + \mathbf{m}\left(\phi_i^{(j)}, \theta_i^{(j)}\right)\right), \tag{2.34}$$

where ξ is a parameter encapsulating the geometric properties of the camera, $\hbar$ is a non-linear function, $\mathbf{K}_c$ is a conventional camera calibration matrix, and $\mathbf{m}(\cdot)$ the function which maps the elevation and azimuth angles to a unit vector. The Jacobian of the measurement function is computed as:

$$\begin{aligned} \mathbf{H}_i^{(j)} &= \frac{\partial \boldsymbol{\Pi}^{(j)}}{\partial \mathbf{h}_{C,i}^{(j)}} \left[\frac{\partial \mathbf{h}_{C,i}^{(j)}}{\partial \delta \mathbf{x}_i^C} \quad \cdots \quad \frac{\partial \mathbf{h}_{C,i}^{(j)}}{\partial \delta \mathbf{y}_{IDP,i}^{(j)}} \quad \cdots\right] \\ &= \frac{\partial \boldsymbol{\Pi}^{(j)}}{\partial \mathbf{h}_{C,i}^{(j)}} \left({}_W\mathbf{R}_i^C\right)^T \left[-\rho_i^{(j)} \quad [{}_W\mathbf{R}_i^C \mathbf{h}_{C,i}^{(j)}]_\times \quad \mathbf{0} \quad \mathbf{0} \cdots \rho_i^{(j)} \quad \frac{\partial \mathbf{m}}{\partial\left(\phi_i^{(j)},\ \theta_i^{(j)}\right)^T} \quad \left(\mathbf{r}_{W,i}^{(j)} - \mathbf{r}_{W,i}^C\right) \cdots\right], \end{aligned} \tag{2.35}$$

where $\frac{\partial \mathbf{\Pi}^{(j)}}{\partial \mathbf{h}_{C,i}^{(j)}}$ is the Jacobian of the sphere projection model from (2.8).

With this generalised model, the observations of the tracked features (those in the EKF state vector) are predicted, and then putative matches $\hat{\mathbf{z}}_i^{(j)}$ are obtained by active search in the uncertainty region given by the bound of the 95% confidence interval of the projection $\mathbf{S}_i^{(j)}$ of the state covariance:

$$\mathbf{S}_i^{(j)} = \mathbf{H}_i^{(j)} \mathbf{P}_i {\mathbf{H}_i^{(j)}}^{\mathsf{T}} + \mathbf{\Sigma}_{\mathbf{z}}^{(j)}, \tag{2.36}$$

where $\mathbf{\Sigma}_{\mathbf{z}}^{(j)}$ is the measurement noise in the image.

Spurious matches are rejected by RANSAC [Civera et al., 2010], and then correct matches are used to update the state both of the camera and the landmarks. For the update step we need to compute the innovation $\boldsymbol{\nu}_i$, the covariance of the measurements $\mathbf{S}_i$ and the Kalman gain $\mathbf{K}_i$:

$$\boldsymbol{\nu}_i = \begin{pmatrix} \hat{\mathbf{z}}_{i1} - \mathbf{\Pi}\left(\mathbf{x}_i^C, \mathbf{y}_{IDP,i}^{(1)}\right) \\ \vdots \\ \hat{\mathbf{z}}_{iM} - \mathbf{\Pi}\left(\mathbf{x}_i^C, \mathbf{y}_{IDP,i}^{(M)}\right) \end{pmatrix} \tag{2.37}$$

$$\mathbf{H}_i = \begin{bmatrix} \mathbf{H}_i^{(1)} \\ \vdots \\ \mathbf{H}_i^{(M)} \end{bmatrix} \tag{2.38}$$

$$\mathbf{S}_i = \mathbf{H}_i \mathbf{P}_i \mathbf{H}_i^T + \mathbf{\Sigma}_{\mathbf{z}}, \tag{2.39}$$

$$\mathbf{K}_i = \mathbf{P}_i \mathbf{H}_i^T \mathbf{S}_i^{-1}. \tag{2.40}$$

And the state update $\mathbf{\Delta x}_i = \left(\mathbf{\Delta r}_{W,i}^C,\ \mathbf{\Delta}\theta_{W,i}^C,\ \mathbf{\Delta v}_{W,i}^C,\ \mathbf{\Delta}\boldsymbol{\omega}_{C,i}^C,\ ...,\ \mathbf{\Delta y}_{IDP,i}^{(j)},\ ...\right)$ is computed as:

$$\mathbf{\Delta x}_i = \mathbf{K}_i \boldsymbol{\nu}_i. \tag{2.41}$$

For each variable in the state vector the update is applied by direct addition, $x \leftarrow x + \Delta x$, except for the rotation where the increment is applied by computing the exponential map and the update and the composition operation:

$${}_W\mathbf{R}_i^C \leftarrow \exp\left([\mathbf{\Delta}\theta_{W,i}^C]_\times\right) {}_W\mathbf{R}_i^C \tag{2.42}$$

The covariance of the state estimate has also to be updated due to the gain of information from the measurements in the last processed frame:

$$\mathbf{P}_i \leftarrow \left(\mathbf{I} - \mathbf{K}_i \mathbf{H}_i\right) \mathbf{P}_i. \tag{2.43}$$

2.4.3. Initialization of landmarks

After the EKF update, a keypoint extraction is performed in the image and new landmarks are initialised preferently in image areas with low density of features. New landmarks in inverse depth coordinates, $\mathbf{y}_{IDP,W}^{(j)} = \left(\mathbf{r}_W^{C(j)},\ \theta^{(j)},\ \phi^{(j)},\ \rho^{(j)}\right)^T$, are initialised from the current estimate of the camera state vector $\mathbf{x}_i^C$ and the corresponding keypoint location $\mathbf{z}^{(j)}$ in the image:

$$\mathbf{r}_W^{C(j)} = \mathbf{r}_{W,i}^C, \tag{2.44}$$

$$\begin{pmatrix} \theta^{(j)} \\ \phi^{(j)} \end{pmatrix} = \mathbf{m}^{-1}\left(\mathbf{h}_{W,i}^{(j)}\right), \tag{2.45}$$

$$\rho^{(j)} = \rho_0, \tag{2.46}$$

with $\mathbf{h}_{W,i}^{(j)} = {}_W\mathbf{R}_i^C \mathbf{\Pi}^{-1}\left(\mathbf{z}^{(j)}\right)$ where the function $\mathbf{m}^{-1}(\cdot)$ maps a projective ray to azimuth and elevation angles. Note that, since the inverse depth $\rho^{(j)}$ is not observable from just the first image measurement it is initialised to an arbitrary prior ρ_0 with a high standard deviation $\sigma_{\rho 0}$.

We have to estimate also de covariance of the new landmark. First we compute the Jacobians of the previous functions with respect to the state vector and the measures:

$$\mathbf{H}_{inv,\mathbf{x}_i^C}^{(j)} = \frac{\partial \mathbf{y}_{IDP,W}^{(j)}}{\partial \delta \mathbf{x}_i^C} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\frac{\partial \mathbf{m}^{-1}}{\mathbf{h}_{W,i}^{(j)}}[\mathbf{h}_{W,i}^{(j)}]_\times & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 \end{bmatrix} \tag{2.47}$$

$$\mathbf{H}_{inv,\mathbf{z}}^{(j)} = \begin{bmatrix} \frac{\partial \mathbf{y}_{IDP,W}^{(j)}}{\partial \mathbf{z}^{(j)}} & \frac{\partial \mathbf{y}_{IDP,W}^{(j)}}{\partial \rho^{(j)}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ -\frac{\partial \mathbf{m}^{-1}}{\mathbf{h}_{W,i}^{(j)}}\, {}_W\mathbf{R}_i^C \frac{\partial \mathbf{\Pi}^{-1}\left(\mathbf{z}^{(j)}\right)}{\partial \mathbf{z}^{(j)}} & \mathbf{0} \\ 0 & 1 \end{bmatrix}. \tag{2.48}$$

Then the covariance of a 3D point in inverse depth is computed as:

$$\mathbf{\Sigma}_{\mathbf{y}_{IDP}}^{(j)} = \mathbf{H}_{inv,\mathbf{x}_i^C}^{(j)} \mathbf{P}_i^{CC} \left(\mathbf{H}_{inv,\mathbf{x}_i^C}^{(j)}\right)^T + \mathbf{H}_{inv,\mathbf{z}}^{(j)} \begin{bmatrix} \mathbf{\Sigma}_{\mathbf{z}}^{(j)} & \mathbf{0} \\ 0 & \sigma_{\rho 0}^2 \end{bmatrix} \left(\mathbf{H}_{inv,\mathbf{z}}^{(j)}\right)^T, \tag{2.49}$$

and appended to the state covariance

$$\mathbf{P}_i \leftarrow \begin{bmatrix} \mathbf{P}_i & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{\mathbf{y}_{IDP}}^{(j)} \end{bmatrix}. \tag{2.50}$$

2.4.4. A remark on the EKF and least squares optimisation

The update of EKF can be interpreted also as the solution of the following non-linear least squares optimisation problem performing a single iteration.

$$\underset{\mathbf{x}_i}{\arg\min} \left\| \begin{pmatrix} \mathbf{f}\left(\hat{\mathbf{x}}_{i-1}^C, \mathbf{0}\right) \boxminus \mathbf{x}_i^C \\ \vdots \\ \hat{\mathbf{y}}_{IDP,i-1}^{(j)} - \mathbf{y}_{IDP,i}^{(j)} \\ \vdots \end{pmatrix} \right\|_{\mathbf{P}_i^{-1}}^2 + \sum_j \left\| \hat{\mathbf{z}}_{ij} - \mathbf{\Pi}\left(\mathbf{x}_i^C, \mathbf{y}_{IDP,i}^{(j)}\right) \right\|_{\mathbf{\Sigma}_{\mathbf{z}}^{(j)}}^2 \tag{2.51}$$

$$\mathbf{P}_i = \begin{bmatrix} \mathbf{F}_i & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{P}_{i-1} \begin{bmatrix} \mathbf{F}_i^T & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} + \begin{bmatrix} \mathbf{G}_i \mathbf{\Sigma}_{\mathbf{a},\boldsymbol{\alpha}} \mathbf{G}_i^T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{2.52}$$

where essentially the predicted camera pose and previous estimates of the 3D landmarks in the map are being treated as a multidimensional measurement with covariance $\mathbf{P}_i$, and new observations of the landmark on the current image are introduced as measurements $\hat{\mathbf{z}}_{ij}$ with covariance $\mathbf{\Sigma}_{\mathbf{z}}^{(j)}$.

By taking the predicted camera pose and previous estimates of the landmark positions as the seed for the non-linear optimisation

$$\mathbf{x}_i^C = \breve{\mathbf{x}}_i^C \boxplus \mathbf{\Delta x}_i^C = \mathbf{f}\left(\hat{\mathbf{x}}_{i-1}^C, \mathbf{0}\right) \boxplus \mathbf{\Delta x}_i^C, \tag{2.53}$$

$$\mathbf{y}_{IDP,i}^{(j)} = \breve{\mathbf{y}}_{IDP,i}^{(j)} + \mathbf{\Delta y}_{IDP,i}^{(j)} = \hat{\mathbf{y}}_{IDP,i-1}^{(j)} + \mathbf{\Delta y}_{IDP,i}^{(j)}, \tag{2.54}$$

and linearising the cost function around this seed we get:

$$\underset{\mathbf{\Delta x}_i}{\arg\min} \, ||\mathbf{\Delta x}_i||^2_{\mathbf{P}_i^{-1}} + ||\boldsymbol{\nu}_i - \mathbf{H}_i \mathbf{\Delta x}_i||^2_{\mathbf{\Sigma_z}} \tag{2.55}$$

where the solution $\Delta \mathbf{x}_i$ for is straightforward

$$\Delta \mathbf{x}_i = \left(\mathbf{P}_i^{-1} + \mathbf{H}_i^T {\mathbf{\Sigma_z}}^{-1} \mathbf{H}_i\right)^{-1} \mathbf{H}_i^T {\mathbf{\Sigma_z}}^{-1} \boldsymbol{\nu}_i \tag{2.56}$$

where it can be shown by using matrix identities [Bell and Cathey, 1993] that the expression $\left(\mathbf{P}_i^{-1} + \mathbf{H}_i^T {\mathbf{\Sigma_z}}^{-1} \mathbf{H}_i\right)^{-1} \mathbf{H}_i^T {\mathbf{\Sigma_z}}^{-1}$ is equivalent to the Kalman gain $\mathbf{K}_i$.

One interesting consequence of this view of the EKF, comes after noticing that, by means of the anchored inverse depth parametrisation of landmarks, rather than discarding old poses as normally occurs in filtering-based SLAM techniques, they are still tracked as landmark anchors. Then, the result of using this landmark parametrisation is that it can bring together two approaches, probabilistic filtering and Bundle Adjustment, which are usually depicted in the literature as opposite ways to solve the SLAM problem. Under the light of anchored inverse depth landmarks an EKF-SLAM approach is equivalent to the online resolution of a Bundle Adjustment optimisation.

2.5. Patch warping for catadioptric systems

An essential step in Monocular SLAM is the tracking of the landmarks of the map to compute the measurements of their 2D projections in the current frame. The tracking is performed through an active search in a region bounded by a prior uncertainty in the camera motion between consecutive frames. In this active search, a patch which is taken as the landmark descriptor during its initialisation is compared against the patches computed around every pixel in the search region, and the one with best correlation is picked as a putative match.

A prior warping of the patches predicting the change in appearance due to the change in viewpoint produced by the camera motion, considerably increases the chances of getting positive matches. This warping is specially important in catadioptric systems where the high distortion induced by the camera-mirror system leads to large changes in appearance under some camera motions. This section is then devoted to the computation of a simple affine transform which provides rotational and scale invariance in catadioptric systems.

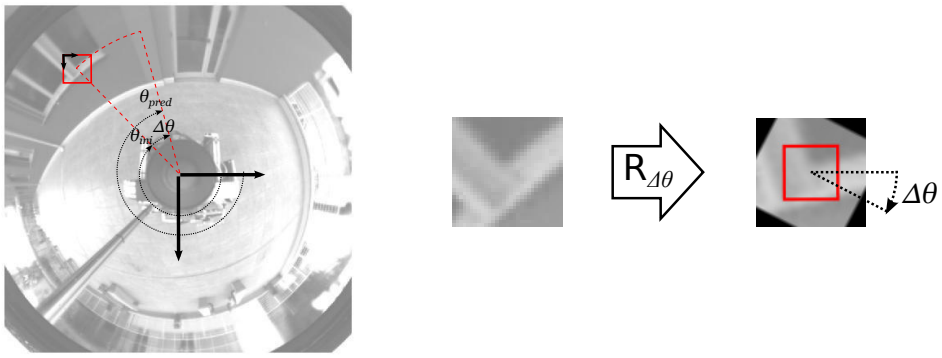


Figure 2.2: Rotation transformation computed from $\Delta\theta = \theta_{pred} - \theta_{ini}$ is applied to a big patch. New patch for correlation is extracted from the warped patch.

2.5.1. Rotation invariance

For the rotation invariance we inspire on the idea proposed in [Andreasson et al., 2007]. A squared oriented patch is extracted by bilinear interpolation in the radial direction from the principal point to the feature. This patch is then rotated to a fixed orientation and stored as descriptor.

However, in the used SLAM application matching is done by active search in a small region in the image. So, if we use Andreasson's approach each candidate patch inside the search region should be determined by bilinear interpolation in the non-natural radial and polar directions of the image, which would be time consuming.

To avoid this, we combine this idea with the implementation existing in the SLAM application [Molton et al., 2004]. A bigger patch is extracted during feature initialisation. Before the matching process, the big patch is warped by an homographic transformation [Hartley and Zisserman, 2000] to predict how the appearance patch varies depending of the variation of the position of the camera respect to the position in which the feature was initialised. A new patch for correlation is extracted from the center of the warped big patch. This way patches for correlation are always determined in the horizontal and vertical directions of the image and bilinear interpolation is only computed during the big patch warping.

For its use with omnidirectional cameras, instead of computing the homography, we transform the patch by a rotation transformation given by the variation of the polar angle ($\Delta\theta$) of the feature in the image between the current prediction of its projection and the position where it was first observed and initialised (Fig. 2.2).

2.5.2. Scale invariance

To reach scale invariance, we develop the simple idea of scaling the patch by a given scale factor. To consider the variable resolution in the catadioptric image, a theoretical formula was obtained as a function of the mirror parameters and the image position.

To obtain it, first we define a point in the 3D space in homogeneous coordinates at a depth D from the camera with an azimuth ϕ and an elevation of θ. Due to the rotational symmetry of the mirror and for the sake of simplicity an azimuth angle of $\phi = 0$ is taken without loss of generality. So the coordinates of the 3D point yield $\mathbf{X_0} = (D\cos\theta,\ 0,\ D\sin\theta,\ 1)^{\mathsf{T}}$ According to

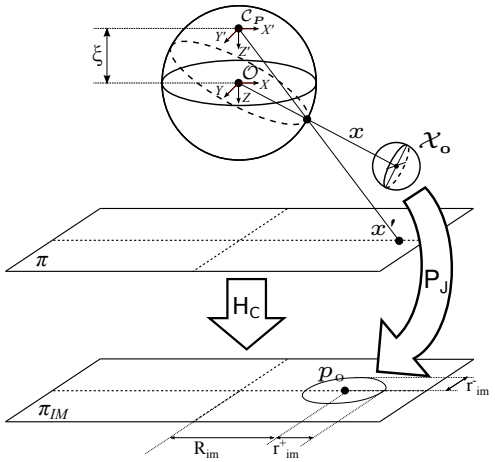


Figure 2.3: Projection of a sphere from the scene to the image plane by the jacobian computed on its centre $\mathbf{X_0}$

the spherical camera model, this point is projected on the image plane as $\mathbf{p_0} = (\frac{\gamma \cos\theta}{\xi + sin\theta},\ 0,\ 1)^\mathsf{T}$ taking the reference frame attached to the principal point ($u_0 = v_0 = 0$ in the matrix $\mathsf{H_C}$). The norm of the projected point is the distance from the principal point R_{im}:

$$R_{im} = ||\mathbf{p_0}|| = \frac{\gamma \cos\theta}{\xi + sin\theta}. \tag{2.57}$$

Projection of the points in the neighbourhood of $\mathbf{X_0}$ can be approximated by a linear mapping from the 3D scene to the image plane given by the projection jacobian of (2.8) computed in $\mathbf{X_0}$, which after some calculations and algebraic manipulation yields:

$$\mathsf{J}_{\mathbf{X}=\mathbf{X_o}} = \frac{\gamma}{D(\xi + S_\theta)^2} \begin{bmatrix} S_\theta(1+\xi S_\theta) & 0 & -C_\theta(1+\xi S_\theta) \\ 0 & \xi + S_\theta & 0 \end{bmatrix}, \tag{2.58}$$

where $S_\theta = \sin\theta$ and $C_\theta = \cos\theta$. This jacobian maps points from a 3D to a 2D euclidean space. To extend it to a projective transformation in the projective space we do:

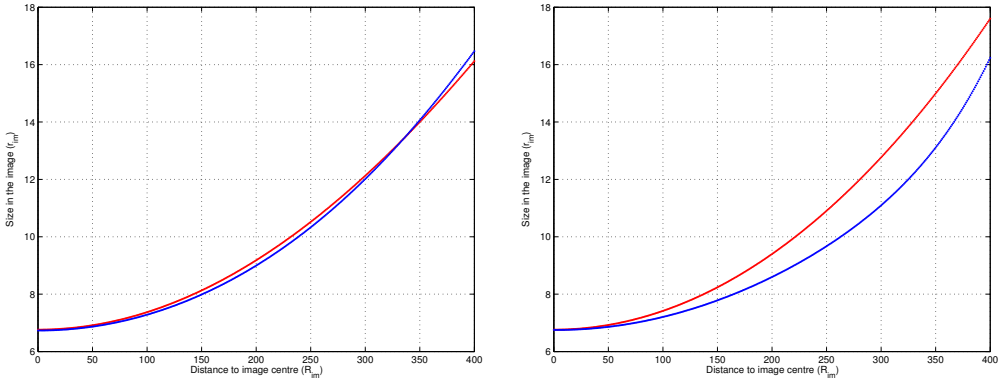
$$\mathbf{P_J}_{(3\times4)} = \begin{bmatrix} \mathbf{J}_{\mathbf{X}=\mathbf{X_o}} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}. \tag{2.59}$$

Now lets take a sphere of radius $r << D$ centred on $\mathbf{X_0}$. It is parametrised by a quadratic form with matrix:

$$\mathbf{Q}_{(4\times4)} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & -r^2 \end{bmatrix}. \tag{2.60}$$

And its projection is computed as follows (see Fig. 2.3):

$$\mathbf{C} = (\mathbf{P_J} \mathbf{Q}^{-1} \mathbf{P_J}^\mathsf{T})^{-1} = \begin{bmatrix} \frac{\gamma^2(1+\xi S_\theta)^2}{D^2(\xi+S_\theta)^4} & 0 & 0 \\ 0 & \frac{\gamma^2}{D^2(\xi+S_\theta)^2} & 0 \\ 0 & 0 & \frac{-1}{r^2} \end{bmatrix}, \tag{2.61}$$



(a) (b)

Figure 2.4: Comparison of the theoretical formulas to calculate the ellipse semi-axis in which the sphere is projected (red) with the results of a simulation using a camera model with distortion parameters (blue). Figure (a) for the minor semi-axis. Figure (b) for the major semi-axis

where $\mathbf{C}$ is the matrix which determines the quadratic form of a ellipse with major and minor semi-axis:

$$r_{im}^{+} = \gamma \frac{r}{D} \frac{1 + \xi \sin\theta}{(\xi + \sin\theta)^2} \tag{2.62}$$

$$r_{im}^{-} = \gamma \frac{r}{D} \frac{1}{\xi + \sin\theta} \tag{2.63}$$

From (2.57) for θ we can calculate $\sin\theta$ as a function of R_{im} and the parameters ξ and γ, which we call $f(\xi, \frac{R_{im}}{\gamma})$. After some substitutions and manipulation we get a second order equation with unknown $\sin\theta$. By solving the equation and selecting the solution with physical meaning, we obtain:

$$S_\theta = f(\xi, \frac{R_{im}}{\gamma}) = \frac{\sqrt{1 + (\frac{R_{im}}{\gamma})^2(1 - \xi^2)} - \xi(\frac{R_{im}}{\gamma})^2}{1 + (\frac{R_{im}}{\gamma})^2}. \tag{2.64}$$

According to the obtained formulas for the semi-axis it is deduced that the scale of a feature on the image depends on the following parameters:

- Real size of the feature (r).
- Distance of the feature to the camera (D).
- Camera-mirror parameters ξ and γ.
- Distance in the image to the principal point (R_{im}).

To compute the scale factor the real size of the feature is not relevant since it does not change between frames.

Concerning the contribution of R_{im} and the mirror parameters, to apply a uniform scale factor, one of the two formulas (2.62) and (2.63) must be selected. A sensibility test to the the

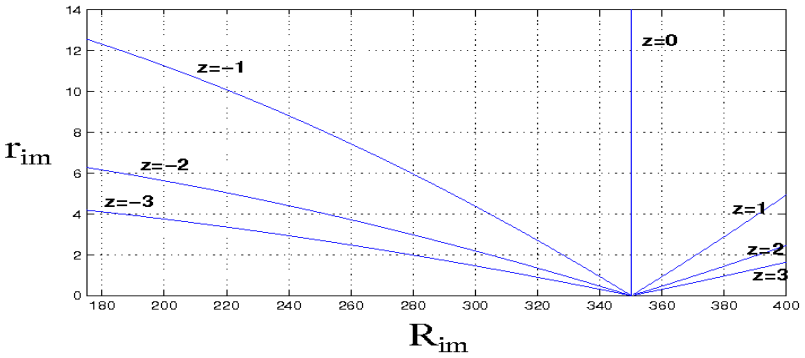


Figure 2.5: Scale of a feature in the image (r_{im}) as a function of the distance to the principal point (R_{im}) and the distance in meters (z) to the plane where the camera moves.

unmodelled image distortion and the linearisation error induced by the jacobian is lead. The test involves a simulation of the projection of a set of spheres with $D = 6\ m$, $r = 0.1\ m$ and a gradual shift on elevation angle using a real camera calibration with 5 distortion parameters [Mei and Rives, 2007]. From the simulation results, empirical data is obtained for the dependency on R_{im} of the semi-axis in the radial and tangential directions of the projected ellipses. These functions are compared with the functions for r_{im}^{+} and r_{im}^{-} respectively (Fig. 2.4). For the major semi-axis the maximum absolute and relative errors are 2 *pixels* and a 15% respectively, while for the minor semi-axis the maximum errors are 0.3 *pixels* and a 2%.

Therefore we select r_{im}^{-} to calculate the scale factor, which yields:

$$k = \frac{r_{im2}^{-}}{r_{im1}^{-}} = \frac{D_1}{D_2}\frac{\xi + f(\xi, \frac{R_{im1}}{\gamma})}{\xi + f(\xi, \frac{R_{im2}}{\gamma})}. \tag{2.65}$$

The depth of the feature in the scene D is the most problematic contribution since it is not observable in one image. As explained in Section 3, new detected features are initialized in IDP with an arbitrary depth value with high uncertainty, which is not reliable to calculate the scale factor. For this reason, the application of the whole scale factor can only be considered when features have been repeatedly observed and thus depth uncertainty has reasonably decreased.

For features with a high depth uncertainty, the application of a partial scaling dropping the depth terms can be considered. To evaluate it, we consider that the camera moves in a plane, so relative movement of the tracked features takes place in parallel planes. By making $D = \frac{z}{\sin\theta}$ in (2.63), we obtain the dependence of r_{im} on $\sin\theta$, and so on R_{im}, at different distances z from the plane where the camera moves (Fig. 2.5). For features below the camera ($z < 0$) their scale decreases until 0 in the line at infinity (given by the circumference with radius $R_{im} = R_\infty = \frac{\gamma}{\xi}$); while for features above the camera its scale increases from R_∞.

However, the contribution of the shape of the mirror always increases with R_{im} (Fig. 2.4). So, assuming movement in a plane, the use of a scale factor without depth estimate only makes sense when the tracked features are above the camera (*i.e.*$R_{im} > R_\infty$). As it only supposes a fraction of the image, the computation of the partial scale factor for landmarks with high depth uncertainty was eventually not considered.

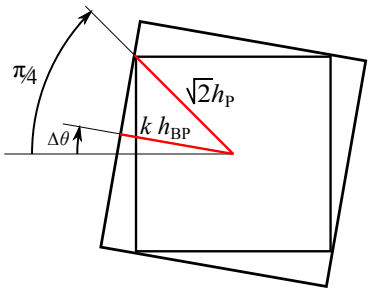


Figure 2.6: Limit for the minimum allowed scale factor ($kh_{BP} = \sqrt{2}h_P \cos(\frac{\pi}{4} - \mod(\Delta\theta, \frac{\pi}{2}))$)

2.5.3. Computation of patch transformation

Before computing the patch transformation, we must be check that the smaller descriptor patch will be fully contained in the zone of the warped big patch which contains some information. The limit situation arises when the borders of the warped patch are in contact with the corners of the descriptor patch (Fig. 2.6). In this case, the scale factor is:

$$k = \sqrt{2}\frac{h_P}{h_{BP}}\cos(\frac{\pi}{4} - \mod(\Delta\theta, \frac{\pi}{2})) \tag{2.66}$$

where h_{BP} and h_P are the half of the sizes of the big patch and the descriptor patch respectively and $\Delta\theta$ is the variation of the polar angle used to construct the transformation. If we add a security margin of 0.1 we obtain an expression for the limit of the scale factor:

$$k_{lim} = \sqrt{2}\frac{h_P}{h_{BP}}\cos(\frac{\pi}{4} - \mod(\Delta\theta, \frac{\pi}{2})) + 0.1 \tag{2.67}$$

With this previous consideration, the computation of the warping is done in the following steps:

1) Check the condition $k > k_{lim}$. If k does not fill this condition it is set to k_{lim}.

2) Calculation of the transformation matrix:

$$\mathbf{H} = \mathbf{H_{tr}}\mathbf{H_S}\mathbf{H_{tr}}^{-1} \tag{2.68}$$

$$\mathbf{H_S} = \begin{bmatrix} k\cos(\Delta\theta) & -k\sin(\Delta\theta) & 0 \\ k\sin(\Delta\theta) & k\cos(\Delta\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.69}$$

with $\mathbf{H_S}$ the transformation matrix which combines the rotation transformation $\mathbf{R_{\Delta\theta}}$ with the scale factor k and $\mathbf{H_{tr}}$ is a translation matrix to translate the patch coordinate frame to the center of the patch.

3) Computation of the warped patch by doing the inverse mapping to the original big patch and performing a bilinear interpolation.

$$\mathbf{X_{BP}} = \mathbf{H_S}^{-1}\mathbf{X_{WP}} \tag{2.70}$$

4) Extraction of the patch for correlation from the center of the warped patch.

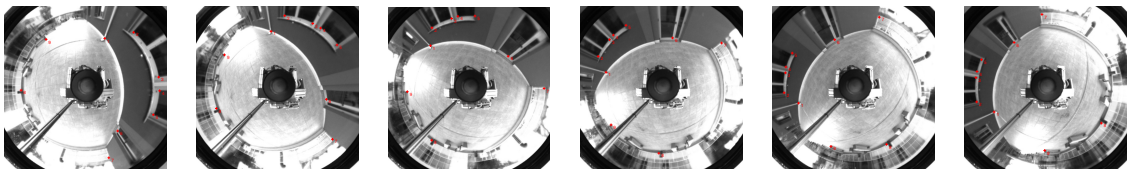


Figure 2.7: Image sequence taken for test 1 (180º rotation). Selected corners for matching are shown in red

2.6. Experiments

In this section experiments to evaluate the omnidirectional visual SLAM and the new patch are presented. The images used to lead the experiments were taken from one of the image databases provided by The Rawseeds Project[1]. This database consists of a sequence with more than 32000 frames acquired by a robot equipped with a hyper-catadioptric camera.

2.6.1. Experiment 1

We carried three tests out where we decoupled the matching process from the SLAM algorithm to make a preliminary evaluation of the rotation and the scale factor transformation. The set up of the three tests was quite similar. First, we extracted corners on the first frame with the FAST extractor. Among the extracted corners, we selected some features and stored their locations and their patches. Like the matching proccess has been decoupled from the SLAM algorithm, the true locations of the features were manually selected on each frame (Fig. 2.7) and the search region was fixed to a 50x50 pixels square. The matching in the selected frames is done by obtaining for each feature the best correlation inside the search region, as is done in the SLAM application.

For each feature and frame, we have defined the following variables to be measured:

- Correlation in true feature location.
- Best correlation in search region.
- Distance between true feature location and best correlation location.

Test 1: Rotated patch and 180º rotation

We evaluate the rotation of the patches in a sequence in which the robot rotates 180º. From this sequence we have extracted 6 frames spaced by 20 frames between them. We have selected 9 features to carry the test out (Fig. 2.7).

The results show that rotated patches provide a better correlation value on the true feature location, which confirms that they are more rotation invariant than the non rotated patches (Fig. 2.8). The rotated patch also provides by far better values for the best correlation inside the search region (all of them above 0.9) as well as a very low distance between true feature location and matched location.

[1] `HTTP://www.rawseeds.org`

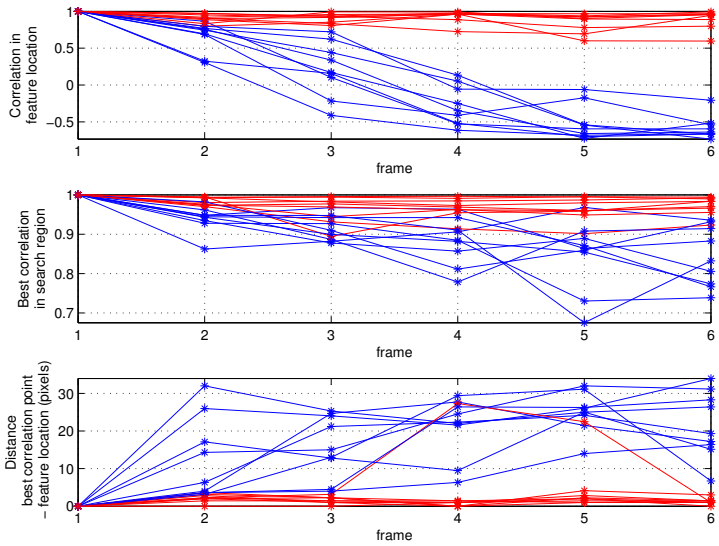


Figure 2.8: Matching results of test 1. In red, with oriented patch. In blue, with not oriented patch

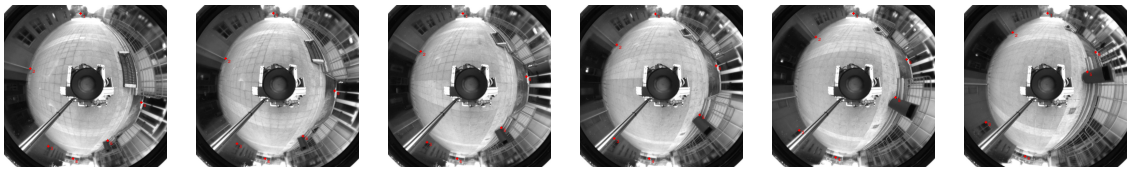


Figure 2.9: Image sequence taken for test 2 (translation). Selected corners for matching are shown in red

Test 2: Rotated patch and translation

Performance of rotated patches is evaluated in a sequence only containing camera translation. 6 frames were extracted with intervals of 20 frames between them and the number of selected features was 6 (Fig. 2.9).

The results (Fig. 2.10) reveal that slightly better correlation values are obtained using a rotated patch. This is due to the relative motion of the map features along lines which are projected as conic curves in a catadioptric image. So, the rotated patch initially intended to improve the matching results during camera rotations, can also deal with translations better than a non rotated patch. However as the translated distance increases, the matchings tend to be made in the wrong location for both patches. One possible reason is that as the robot translates, the features change their scale and their point of view and can become occluded by other scene objects.

Test 3: Scaled patch

We evaluate the performance of the matching process with respect to the scale changes. Due to the decoupling from SLAM, it is not possible to determine the depth of the patches extracted and the contribution of the depth to the scale factor is not considered. So, an image acquisition without depth changes in the features has been made following the next steps:

- Select a zone in the scene with potential patch richness and situated far enough from the camera so that $D \to \infty$.

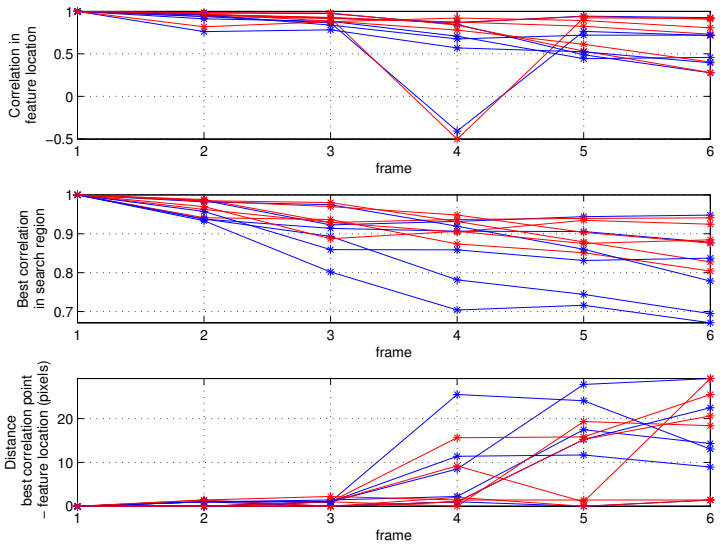


Figure 2.10: Matching results of test 2. In red, with oriented patch. In blue, with not oriented patch

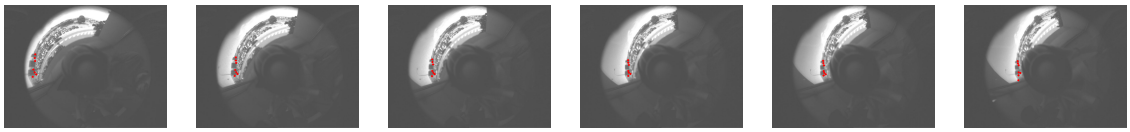


Figure 2.11: Image sequence taken for test 3 (scale change). Selected corners for matching are shown in red

- Capture images while camera rotates so that the selected zone moves only along the radial direction. As infinite distance has been assumed, little camera displacements during capture are not problematic.

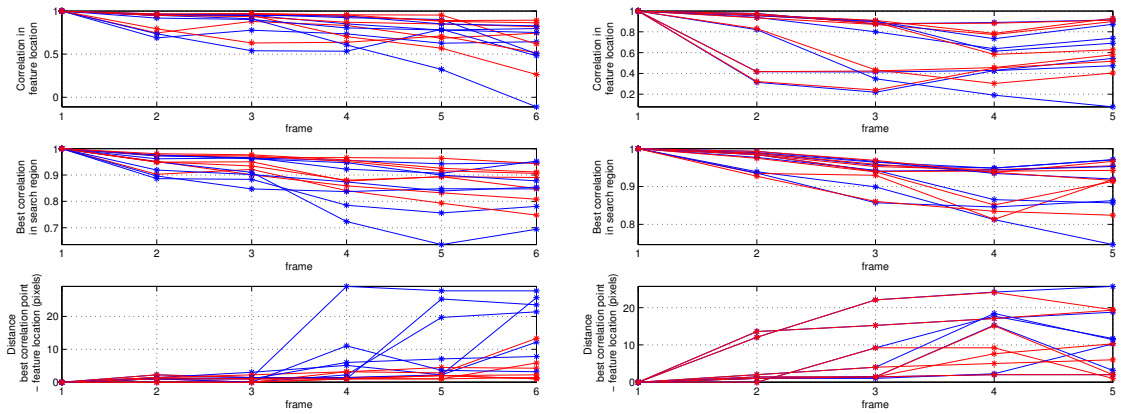
A sequence of 6 images was taken for the test. The number of selected features was 7. To evaluate the performance of the patches under scale change, the features were selected in a zone with no orientation change in the image (Fig. 2.11). Two cases have been carried out to prove the performance under scale decrease $k < 1$ and scale increase $k > 1$. In the scale decrease case, the extraction of the features was made in the image where the zone of extraction was the furthest from the image centre. For the scale increase case the order of the images has been inverted.

The results of the tests show that in scale decrease (Fig. 2.12(a)) the patch with scaling offers a better performance than a normal patch while in the case of scale increase(Fig.2.12(b)) both patches perform in a similar way, due to the impossibility of extracting new information by increasing the scale of an image.

2.6.2. Experiment 2

After testing the new transformations applied to the patch, we evaluated it integrated in our visual SLAM approach for omnidirectional cameras with the Real-Time application developed by Davison *et al.*. For the evaluation we selected a long outdoor sequence from the database provided by the Rawseeds Project.

To compare our warped patch with the normal patch we ran the sequence using both patches



(a) (b)

Figure 2.12: Matching results of test 3 for scale decrease (a) and scale increase (b). In red, with scaled patch. In blue, with unscaled patch

Table 2.1: Total number of initialised features (FI), Matchings per initialised feature (R_m) and features in map per initialised feature(R_f)

Correlation threshold	Warped patch			Normal patch		
	FI	R_m	R_f	FI	R_m	R_f
0.8	8648	22.31	0.1	8923	19.75	0.087
0.9	9834	19.38	0.062	10854	16.09	0.046
0.95	13189	13.31	0.027	14970	10.56	0.019

with different correlation thresholds for matching and we have measured three variables:

- Total number of features initialised (FI) .
- Matchings per feature ratio: ($R_m = \frac{Total\ matchings}{FI}$) .
- Features in map per feature initialised ratio: ($R_f = \frac{Final\ map\ size}{FI}$)

The results in Table 2.1 show that the warped patch for omnidirectional cameras performs better than the patch with no warping, as SLAM initialises less features and is obtains more information for SLAM per initialised feature. On the other, the correlation threshold may have more influence on the measured variables than the kind of patch, but at the expense of reducing the *a priori* precision of SLAM. In Fig. 2.13 the projections on the XY plane and the YZ plane of the trajectory obtained with the new patch and a correlation threshold of 0.8 are shown. Note that although the MonoSLAM application estimates 3D camera motion, being not bounded to a 2D plane, according to the obtained trajectory the camera is moving on the ground plane.

Finally we compared this trajectory with respect to the ground truth provided by the GPS data. As scale is not observable by one single camera, for the comparison we scaled the trajectory and aligned it with the trajectory obtained with the GPS (Fig. 2.14). To evaluate the accuracy of the SLAM trajectory numerically we have calculated the mean error of the distance between the corresponding points of both trajectories. The mean error is $\mu_{err} = 3.44\ m$ with a standard deviation of $\sigma = 1.93\ m$ and a maximum error of $max_{err} = 6.73\ m$. Dividing by the trajectory length, we obtain a relative mean error of 1%.

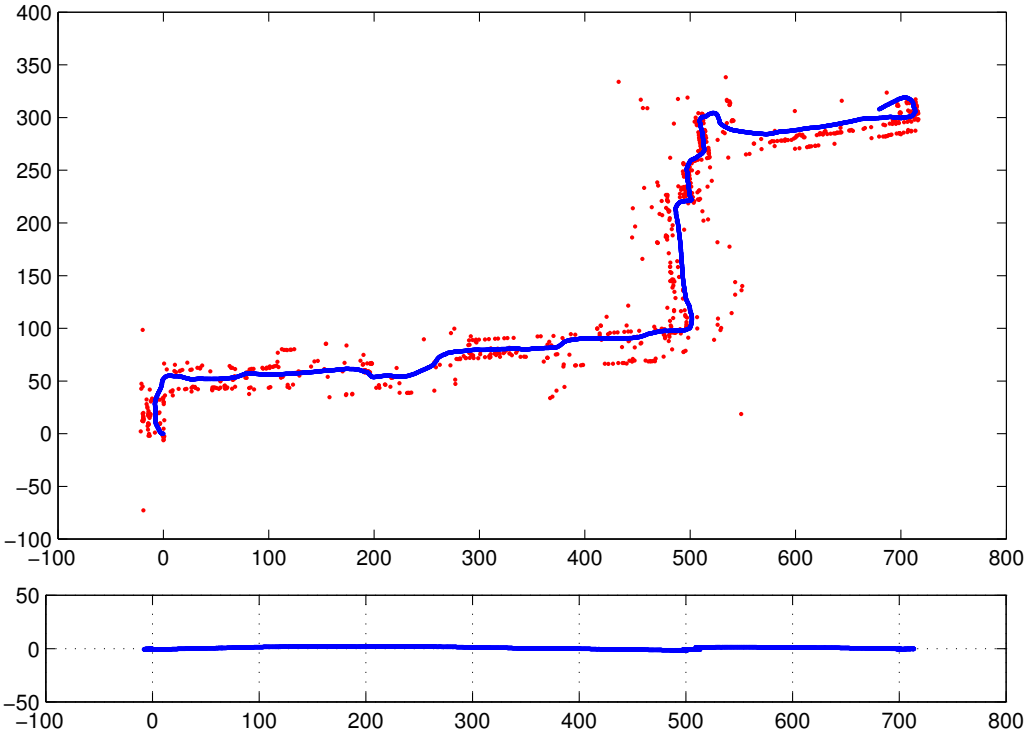


Figure 2.13: SLAM trajectory with correlation threshold 0.8 using the warped patch projected on the XY plane (up) and on the YZ plane (down) The red dots are the map features

2.7. Discussion

In this work we have developed a Visual SLAM for omnidirectional cameras building on state of the art EKF monocular SLAM [Civera et al., 2010] for conventional cameras. Two main modifications have been made: the implementation of the Spherical Camera Model for projection and the formulation of a new patch for omnidirectional cameras which aims to be rotation and scale invariant. Then we have lead experiments to compare the new patch with a conventional patch. First we have tested the matching process decoupled from the SLAM. Once the superiority of the new patch has been proven we have run in real time the SLAM algorithm in a 340 meters long trajectory. Results have shown that the obtained trajectory estimation is quite accurate, which encourages us to make experiments with longer trajectories in the future.

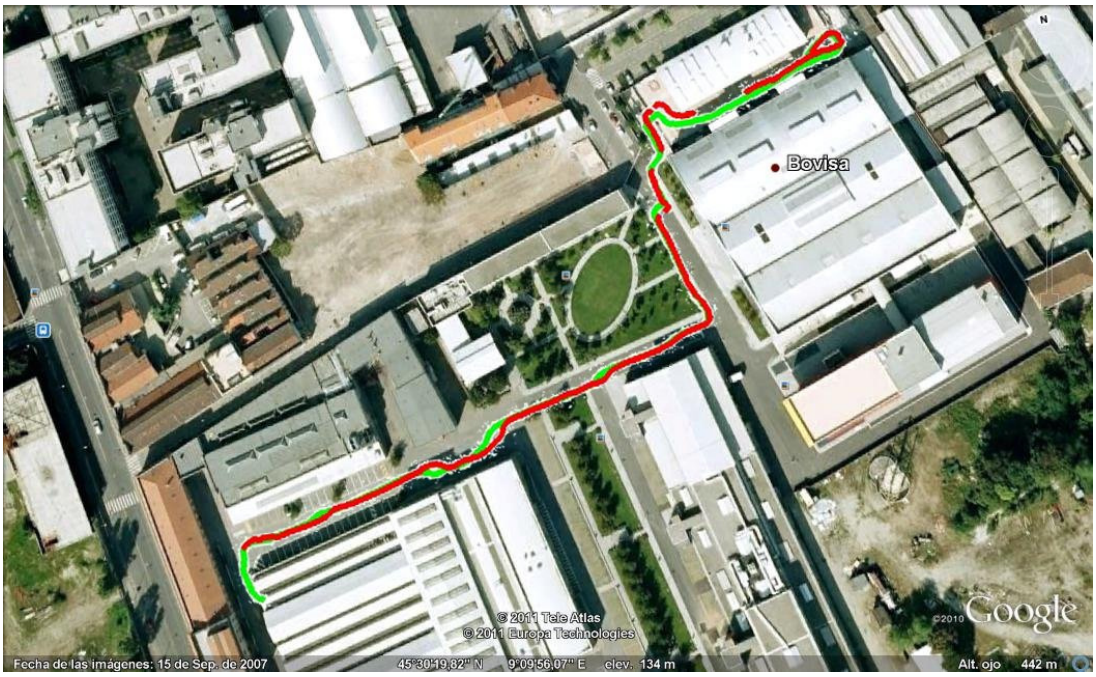


Figure 2.14: GPS trajectory (red) and SLAM trajectory (green) superposed on the satellite image of the Campus of Bovisa (Milan) where the sequences were acquired

Chapter 3

True Scaled 6 DoF Egocentric Localisation with Monocular Wearable Systems

After generalising a real-time monocular SLAM to work with any central projection system, in this chapter we present a novel approach to obtain scaled odometry and map estimates from monocular SLAM with wearable cameras of conventional or catadioptric types. Noticing that the oscillation of the body during walking can be observed in the odometric estimate from a monocular SLAM algorithm, we estimate the walking speed just from the step frequency using an empirical relation obtained in biomedical studies analysing human gait. This relation is supported by the tendency in humans to optimise the metabolic cost of the action of walking. A scale factor can be then dynamically computed to obtain a true scaled estimate of the map and visual odometry, avoiding scale drift on long term trajectories. Although the algorithm requires the person to be walking in order to estimate the scale, the experiments, carried out in outdoor and indoor environments and with different types of cameras, show that our method is reliable and robust to challenging situations like stops, changes in pace or stairs, and provides a significant improvement with respect to the initial unscaled estimate. It also outperforms state-of-the-art solutions to correct the scale drift in monocular SLAM, giving in addition the absolute scale of the trajectory and the 3D observed scene.

3.1. Introduction

In this chapter we propose a novel approach to compute dynamically the true scale from visual odometry estimates obtained with wearable single cameras (Fig. 3.1), avoiding scale drift problems in large environments. Our method is specially suited to be used in wearable systems since it takes advantage of the characteristic oscillatory movement of human body during walking to extract the scale information. The implementation is done within a state-of-the-art visual SLAM approach [Civera et al., 2010]. Nevertheless as it only needs the output corresponding to the trajectory estimate it can be used in any visual odometry system, provided that the accuracy and the time resolution of the SLAM algorithm are fine enough to capture the camera oscillation associated to walking. Also, to clearly perceive the walking oscillations some restrictions must be fulfilled. Firstly, the user must be walking on a relatively plain terrain, such that its roughness

Figure 3.1: Devices used in our experiments. On top, GoPro Hero 2 wide angle camera. On bottom, our helmet with catadioptric camera consisting on a mirror and a VS-C14U-80-ST catadioptric camera.

is lower than the amplitude of the walking oscillations. Secondly, the camera must be attached to a body part whose motion is mainly due to the action of walking, like head, shoulders or chest.

The method works as follows (Fig. 3.2): given an initial unscaled section of the trajectory composed by N camera poses, we extract the signal of the vertical component of the camera position and estimate the step frequency by computing its Discrete Fourier Transform (DFT). The real walking speed is computed from the step frequency using a biomedical relation [Grieve and Gear, 1966] which is supported by the natural tendency to optimise the metabolic cost of walking [Zarrugh et al., 1974,Kuo, 2001]. This speed measurement is fed into a computationally simple particle filter to compute a dynamic scale factor to scale each trajectory section.

Additionally, our method is made robust against bad scale factor estimates in two ways. First, the spectral power of the candidate step frequency is tested to be consistent with the amplitude of the walking oscillations. This test makes possible the detection of non-walking situations, e.g., when the user is stopped, and then allowing for a different update strategy of the scale factor. A second compatibility test is performed during particle filtering to reject big variations of the scale factor.

Our method has been thoroughly evaluated in experiments with GoPro-like wearable cameras as well as with a prototype consisting on a helmet with a catadioptric camera attached on top. We tested the performance of the method when camera is attached to different body parts (head and chest), and compared it to [Strasdat et al., 2010] to show the ability of our method to remove scale drift. We also measured the sensitivity of our method to a inaccurate estimate of user-dependent calibration parameters.

3.2. Related work

The main topics for related work in this chapter are wearable vision and the scale problem in monocular SLAM.

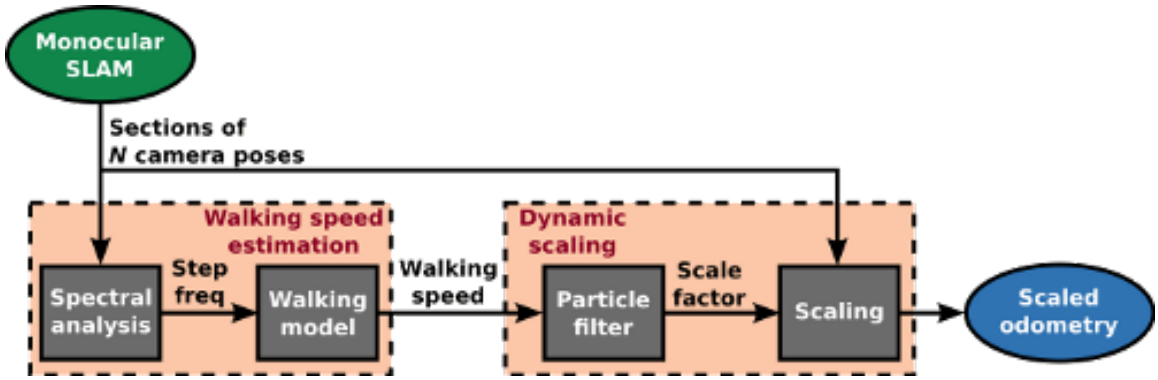


Figure 3.2: Scheme of the basic scaling method.

3.2.1. Wearable vision

The research on wearable cameras for personal aiding has widespread since the pioneering works of Mann [Mann, 1997]. Wearable cameras are normally placed in locations like the head, shoulders or chest, which, for a general purpose, allow for a wide field of view and resilience to body motion [Mayol-Cuevas et al., 2009].

Most works on wearable vision systems have developed towards recognition of human activities, where many approaches take advantage from a nice feature of chest-wearable cameras: the prior knowledge of the action or manipulation taking place at the centre of the image. Recognition problems where wearable systems have been applied are segmentation of handled objects [Ren and Gu, 2010, Fathi et al., 2011], recognition of activities and objects in the workspace [Pirsiavash and Ramanan, 2012], novelty detection in a daily routine [Aghazadeh et al., 2011], clustering of sport activities from video sequences [Kitani et al., 2011], human detection [Mitzel and Leibe, 2012] or analysis of human movements to infer social interactions [Fathi et al., 2012].

With respect to the use of wearable vision for localisation, some works [Kourogi et al., 2001, Aoki et al., 1999] propose appearance based localisation methods in indoor environments. Concerning odometric localisation, Mayol-Cuevas et al. [Mayol et al., 2003] presented a wearable active vision system which changes its heading direction and uses monocular SLAM for self-localisation. A similar system is presented by Castle et al. [Castle et al., 2010], using monocular SLAM and object recognition for augmented reality. In [Badino and Kanade, 2011] a head-wearable stereo system is used to estimate structure and motion. In [Alcantarilla et al., 2012] the authors propose a wearable stereo system which computes, together with SLAM, an estimate of the dense scene flow to segment moving objects in the scene. Also some works [Hesch and Roumeliotis, 2010, Baglietto et al., 2011] propose wearable platforms for human localisation and navigation without vision sensors, combining an Inertial Measurement Unit (IMU) and a laser scanner.

3.2.2. Monocular SLAM and the scale problem

The problem of estimating the true scale in monocular SLAM is addressed either by using additional proprioceptive sensors, like IMUs and odometers, which provide metric information; or by considering geometric priors or constraints, mainly in robotic platforms.

Among solutions using additional sensors, in [Lupton and Sukkarieh, 2008] the true map scale is made observable by integrating the visual data and the IMU data within an information

filter. In [Nützi et al., 2010], the scale is computed by fusing the position and velocity from the visual odometry estimate with the IMU data in an Extended Kalman Filter (EKF). In a similar approach [Weiss and Siegwart, 2011] includes not only the position and velocity, but also the orientation estimate in the EKF state vector, in order to detect failures of the visual system by checking the consistency with the IMU gyro measurements. In [Engel et al., 2012], the scale factor of a quadricopter visual odometry estimate is computed from the measurements of its on-board IMU and altimeter by using an optimisation scheme.

In [Cumani et al., 2004] the wheel odometry is used to provide a prior estimation of the true scaled motion between two consecutive frames which is refined by the update from camera measurements. [Eudes et al., 2010] takes the odometric measurement of distance between two camera poses to compute a scale factor which is applied to the displacement estimated with the camera. Similarly, in [Scaramuzza et al., 2009b], the measurement of the vehicle speed is used to compute the true distance between the last two frames and recover the 3D structure by triangulation of the common image points. In [Civera et al., 2010] a scaled visual odometry estimate is obtained by fusing the wheel odometry and visual information in a EKF-SLAM framework.

Concerning works not using additional sensors, [Lothe et al., 2010] uses the prior knowledge of the distance from the camera to the ground plane, which is obtained from the estimated 3D points of the scene, to compute the scale factor of the scene. In [Scaramuzza et al., 2009a] non-holonomic motion constraints of wheeled vehicles are exploited to resolve the scale when the vehicle turns. In [Botterill et al., 2012] the scale drift problem is solved by identifying previously learnt object classes of the environment and measuring the size of these objects to improve the scale estimate. In [Strasdat et al., 2010], the authors propose a loop closure technique in which camera poses are defined as 7 DoF similarity transformations (translation, rotation and scale) rather than the habitual 6 DoF rigid transformations (translation and rotation). This allows to correct the deformation produced by the scale drift, though it still remains a scale ambiguity in the final estimate.

The method proposed in this chapter fits in the second category, since we get the scale only using the visual information and a prior given by a walking model which depends on the height and specific user parameters. The strength of our method is that it is explicitly thought to operate on human wearable systems, where unlike to ground vehicles, odometric information from encoders is not available to recover the scale.

On the other hand, faced to scale estimation methods not using odometric measurements, all of them have their own drawbacks. Using the IMU, besides involving an additional sensor, to obtain scale information we need to know the initial speed, which, if not known *a priori*, can only be estimated by performing a joint IMU-camera initialisation. The estimation of the scale by the identification of the ground plane can be problematic in environments with a poor textured ground and containing other dominant planes like walls. The identification of known objects of the scene relies in the detection of a set of previously learnt object categories which could be affected by false positives. The correction of the scale drift at loop closures is only possible when one region of the environment is revisited.

Thus, given that all the approaches have their own limitations, an alternative way to compute the scale as proposed in this chapter has always a beneficial effect.

3.3. Monocular SLAM

Monocular SLAM algorithms aim to estimate a visual odometry and at the same time build a map of landmarks using only the measurements from a single camera. The algorithm we use

in this work is based in the monocular EKF-SLAM proposed in [Civera et al., 2010] and its adaptation to omnidirectional cameras as it was described in Chapter 2. For completeness we include a brief description in the following lines.

The camera state as well as the position of the landmarks at time step i are encapsulated in a state vector $\mathbf{x}_i$

$$\mathbf{x}_i = (\underbrace{\mathbf{r}^C_{W,i},\ {}_W\mathbf{R}^C_i,\ \mathbf{v}^C_{W,i},\ \omega^C_{C,i}}_{\text{Camera state } \mathbf{x}^C_i},\ \underbrace{\mathbf{r}^{C(j)}_{W,i}, \theta^{(j)}_i, \phi^{(j)}_i, \rho^{(j)}_i, ...}_{\text{3D points (IDP) } \mathbf{y}^{(j)}_W}), \tag{3.1}$$

where $\mathbf{r}^C_{W,i}$ is the camera position, ${}_W\mathbf{q}^C_i$ is the quaternion of its orientation and $\mathbf{v}^C_{W,i}$ and $\omega^C_{C,i}$ are its linear and angular velocities, respectively.

The 3D locations are parametrised in an anchored inverse depth parametrisation (IDP) [Civera et al., 2008], where $\mathbf{r}^{C(j)}_{W,i}$ is the anchor camera position where the landmark was first viewed, $\theta^{(j)}_i$, $\phi^{(j)}_i$ and $\rho^{(j)}_i$ are respectively the elevation angle, the azimuth angle and the inverse depth with respect to the anchor position.

In an EKF based SLAM a motion and a measurement model must be provided. The motion model describes the change on the camera pose from time step $i-1$ to i and it is described by the following equation:

$$\mathbf{x}^C_i = \mathbf{f}(\mathbf{x}^C_{i-1}, \mathbf{u}_i), \tag{3.2}$$

where $\mathbf{f}(\cdot)$ is the state transition function , $\mathbf{x}^C_{i-1}$ is the past camera pose and $\mathbf{u}_i$ is the control input. This model is used to propagate the state estimate $\mathbf{x}_i$ and its covariance $\mathbf{P}_i$ in the EKF prediction step. Internal measurements from an odometer or an IMU can be integrated as control inputs $\mathbf{u}_i$. Alternatively it is often used a constant velocity model, where the control input $\mathbf{u}_i$ represents the acceleration which is modeled by zero mean random Gaussian noise. Possible changes in velocity are taken care of by setting the variance of the acceleration noise.

The measurement model is used to introduce the information from measurements of external sensors in the EKF update step. In monocular SLAM it is defined by an observation function which encapsulates a projectivity transformation. The observation function depends on the characteristics of the vision system. For central projection systems, i.e., planar (conventional), dioptric or catadioptric systems, it can be divided in these two steps: one projection onto a unit sphere independent from camera parameters and a non-linear mapping from the sphere to the image plane which accounts for the camera geometry and calibration parameters [Geyer and Daniilidis, 2000]. This is synthesised in the following equations:

$$\mathbf{z}^{(j)}_i = \mathbf{\Pi}\left(\mathbf{x}^C_i, \mathbf{y}^{(j)}_W\right) = \mathbf{K}_c \hbar\left(\xi, \frac{\mathbf{y}^{(j)}_{C,i}}{\left|\left|\mathbf{y}^{(j)}_{C,i}\right|\right|}\right) \tag{3.3}$$

$$\mathbf{y}^{(j)}_{C,i} = \left({}_W\mathbf{R}^C_i\right)^T \left(\mathbf{r}^{C(j)}_{W,i} - \mathbf{r}^C_{W,i} + \frac{1}{\rho^{(j)}_i}\mathbf{m}\left(\phi^{(j)}_i, \theta^{(j)}_i\right)\right), \tag{3.4}$$

where ξ is a parameter encapsulating the geometric properties of the camera, $\hbar$ is a non-linear function, $\mathbf{K}_c$ is a conventional camera calibration matrix, and $\mathbf{m}(\cdot)$ the function which maps the elevation and azimuth angles to a unit vector.

With this generalised model, the observations of the tracked features (those in the EKF state vector) are predicted, and then putative matches $\mathbf{z}_i^{(j)}$ are obtained by active search in the uncertainty region given by the bound of the 95% confidence interval of the projection $\mathbf{S}_i^{(j)}$ of the state covariance:

$$\mathbf{S}_i^{(j)} = \mathbf{H}_i^{(j)} \mathbf{P}_i {\mathbf{H}_i^{(j)}}^{\mathsf{T}} + \mathbf{\Sigma}_{\mathbf{z}}^{(j)}, \tag{3.5}$$

where $\mathbf{H}_i^{(j)}$ is the jacobian of the measurement function and $\mathbf{\Sigma}_{\mathbf{z}}^{(j)}$ the measurement noise in the image.

Spurious matches are rejected by RANSAC, and then correct matches are used to update the state both of the camera and the landmarks.

3.3.1. Map Management

To keep the computational cost low, only a few features, referred to as *tracked features* $\mathcal{T}_i$, are kept within the EKF state. Tracked features are divided in *point* (low depth uncertainty) and *ray* (high depth uncertainty) features. At each step, features which failed to be matched in the last frames are removed from the EKF. The set of removed features at step i is denoted by $\mathcal{E}_i = [\mathcal{E}_i^r, \mathcal{E}_i^p]$

The removed point features are added to an independent fixed map $\mathcal{M}_{E,i}$ which is not updated by the EKF:

$$\mathcal{M}_{E,i} = [\mathcal{M}_{E,i-1}, \mathcal{E}_i^p]\,, \tag{3.6}$$

being the global map composed by the point tracked features and the features in $\mathcal{M}_{E,i}$

$$\mathcal{M}_i = [\mathcal{T}_i^p, \mathcal{M}_{E,i}]\,. \tag{3.7}$$

3.3.2. Loop closure

To make possible a fair evaluation of our method using the ground truth trajectory, we correct the noticeable odometry drift which inevitably arises in large experiments by closing the loop at revisited areas. Closing the loops makes also possible the comparison of our proposal with a state-of-the-art method to correct the scale drift [Strasdat et al., 2010].

To compute the relative pose between loop frames we use the libraries OpenCV [Bradski, 2000] for feature extraction and matching and OpenGV [Kneip and Furgale, 2014] for geometric algorithms. Given a loop detected between a recent frame j and an older frame i , we must estimate the rigid body motion between them expressed by the transformation:

$${}_j\mathbf{T}^i = \begin{pmatrix} {}_j\mathbf{R}^i & \mathbf{r}_j^i \\ 0 & 1 \end{pmatrix} \in SE(3). \tag{3.8}$$

To do so, first we establish 2D-2D correspondences between frame j and a close frame j_{aux} with enough parallax; and we robustly estimate with RANSAC the relative pose, rescaling the translational part by taking the norm of the translation between frames from the visual odometry. Secondly, correct correspondences between frames j and j_{aux} are triangulated, and finally ${}_j\mathbf{T}^i$ is obtained from 3D-2D correspondences between the triangulated 3D points referenced at frame j and the keypoints extracted in frame i.

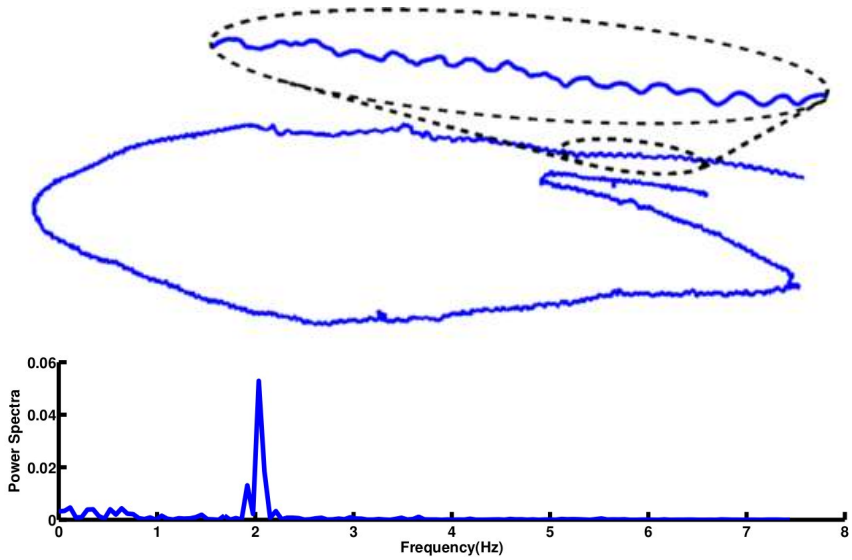


Figure 3.3: Top: Trajectory estimate of Visual SLAM from a head-mounted catadioptric camera including a partial zoom. Bottom: Power spectra of the vertical component

For the comparison with [Strasdat et al., 2010] we need to use similarity transforms:

$$ {}_j\mathbf{S}^i = \begin{pmatrix} {}_j s^i \, {}_j\mathbf{R}^i & \mathbf{r}_j^i \\ 0 & 1 \end{pmatrix} \in Sim(3), \tag{3.9} $$

where we require an additional parameter ${}_j s^i$ to compute the loop constraint. To estimate it, first we compute ${}_i\mathbf{T}^j$ in the same way as with ${}_j\mathbf{T}^i$ by interchanging the roles of i and j. Note that in presence of scale drift ${}_j\mathbf{T}^i \, {}_i\mathbf{T}^j \neq \mathbf{I}$ since translation of each transform has a different scale propagated from the visual odometry at the frame taken as reference for the triangulated 3D points. However, introducing the extra scale parameter in the transform we can apply ${}_j\mathbf{S}^i \, {}_i\mathbf{S}^j = \mathbf{I}$ and thus we get:

$$ {}_j s^i \, {}_j\mathbf{R}^i \mathbf{r}_i^j + \mathbf{r}_j^i = \mathbf{0} \implies {}_j s^i = \frac{\left|\left|\mathbf{r}_j^i\right|\right|}{\left|\left|\mathbf{r}_i^j\right|\right|}. \tag{3.10} $$

3.4. Walking speed estimation

Our scheme for the estimation of the real walking speed is based in two hypotheses. First, the oscillation of the body during walking can be observed in the visual odometry (Fig. 3.3). The second hypothesis is the existence of a tight correlation between the step frequency and the stride length which allows to estimate the walking speed without knowing the later.

3.4.1. Walking speed - step frequency relation

The estimation of the walking speed is based on its close correlation with the step frequency, which was shown in the work by Grieve et al. [Grieve and Gear, 1966]. This correlation is encapsulated in a power law where the walking speed (V_{walk}) normalised with height (H) is presented as a function of the step frequency (f_{step}):

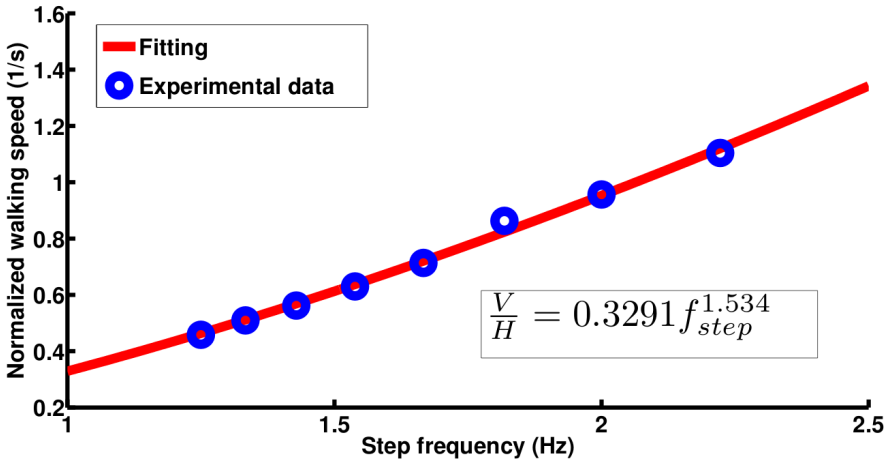


Figure 3.4: Power fitting of the experimental data to compute the relation between walking speed and step frequency ($\mu_{err} = 0.018$, $max_{err} = 0.04$).

$$V_{walk} = \alpha f_{step}^{\beta} H, \tag{3.11}$$

where V_{walk} is in m/s, f_{step} in Hz, H in m, and α and β are characteristic parameters which differ from one individual to another. The mean values provided in [Grieve and Gear, 1966] for these parameters are, after converting to I.S. units, $\alpha = 0.2896$ and $\beta = 1.7544$. Noting that:

$$V_{walk} = f_{step} L_{stride}, \tag{3.12}$$

and substituting in (3.11) we can observe, that though not explicitly shown, our method in essence is taking the stride length L_{stride} as the geometric prior needed to get the absolute scale, and computes it as a function of the step frequency and specific user parameters (α, β and height).

By measuring the oxygen consumptions of different subjects under forced and free gaits for a set of speeds, in [Zarrugh et al., 1974] it was shown that the relation in (3.11) is the result of a human tendency to choose a step frequency which minimises the metabolic cost of walking. [Kuo, 2001] proposed a metabolic cost function modelling the combined actions of pushing off and swinging the leg, whose minimisation predicts the preferred relationship presented in [Grieve and Gear, 1966].

Although walking speed can be estimated using the mean values provided in [Grieve and Gear, 1966], for higher accuracy we use our own α and β parameters explicitly computed for the camera operator. We measured the time t_i it took the operator to walk a distance $s = 100$ m at the times per step ΔT_i given by a metronome ranging from 0.45 to 0.80 seconds in intervals of 0.05 seconds. The height of the operator is $H = 1.88$ m. During the experimentation, it was noticed that measurements above and below the considered interval of times per step were not possible due to the impossibility to consistently synchronise with the metronome beat. Thus, we have established a range of feasible step frequencies between $f_{st}^{-} = 1$ Hz and $f_{st}^{+} = 3$ Hz. Normalised walking speeds V_i' and step frequencies f_i were computed from the raw experimental data. Then a power fitting was applied to obtain the values of $\alpha = 0.329$ and $\beta = 1.534$ (Fig. 3.4).

3.4.2. Estimation of the step frequency

Given the previous biomedical relation, the problem of estimating the walking speed, can be translated into estimating the step frequency.

To correct scale drift, the walking speed must be computed periodically in sections or windows of N camera poses. If we let i be the SLAM time step index, each camera pose $\mathbf{x}_i^C$ can be equivalently notated as $\mathbf{x}_k^C(n)$ assigning a section k and an index n within that section as follows:

$$k = \text{Int}\left(\frac{i-1}{N}\right), \tag{3.13}$$

$$n = i - (k-1)N. \tag{3.14}$$

The world reference frame for the visual odometry is fixed by the initial camera pose. In the experimental setups for the different cameras, the camera frame is oriented so that one of its axis is approximately aligned with the normal to the ground plane. This is a reasonable assumption for a camera worn on the head or some part of the trunk. Also, since a fixed transformation known *a priori* can be applied so that the z-axis of the world frame is aligned with the ground normal, without loss of generality we assume that the body oscillation is observed in the z-component of the trajectory.

The step frequency is estimated by extracting the spectral composition of the oscillatory component of the trajectory estimate using the DFT. When computing the Fourier Transform of discrete signals, the gap between the first and the last elements of the signal, the variations or the drift of the ground plane, and the slight miss-alignment of the reference z-axis with the ground normal can introduce low frequency harmonics. To counteract these effects the data sequence is preprocessed by subtracting the first element $z_k(1)$ from all the elements and applying a second order high pass digital filter with a cut-off frequency of $f_c = 0.7$ Hz, which provides a good attenuation of the low frequency harmonics, without affecting the harmonics in the range of feasible step frequencies. (Fig. 3.5).

After filtering the data the spectral composition is obtained by the DFT:

$$\Gamma_{d,k}(f_m) = \frac{1}{F_s N}\left|\left|\sum_{n=1}^{N} z_k(n)\text{exp}\left(-j\frac{2\pi f_m(n-1)}{F_s}\right)\right|\right|^2 \tag{3.15}$$

$$f_m = \frac{mF_s}{N} \qquad m = -\frac{N}{2}, ..., -1, 0, 1, ..., \frac{N}{2}, \tag{3.16}$$

where Γ_d power spectral density function, F_s is the sampling frequency, which in our case is the number of frames per second (fps) of the camera, and f_m are the frequencies for which the spectrogram is sampled.

The sampling frequency must be high enough to avoid aliasing when the maximum feasible step frequency f_{st}^+ occurs. Also, the number of poses taken must be high enough to provide a good resolution of the spectrogram, with its lower limit given by the minimum number of samples needed to observe at least one oscillation in the case of the minimum feasible step frequency f_{st}^- .

For the computation of the DFT, N has to be as high as possible, but from a global point of view, a high N involve a less frequent update of the scale factor and a reduced ability to detect changes in the walking speed. This can result in a decreasing accuracy in the scaled trajectory estimate. Moreover, if interested in real time operation, the time delay to update the scaled

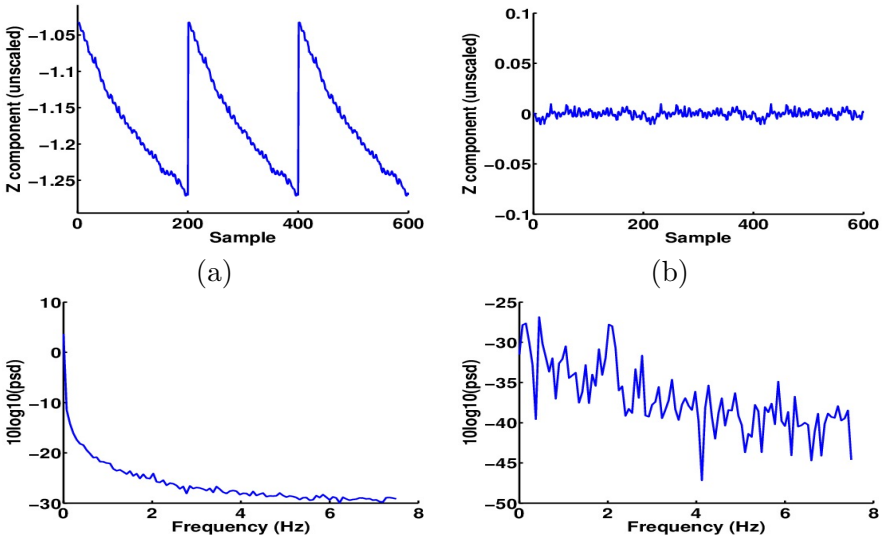


Figure 3.5: Z-component signal segment (top) and corresponding power spectra in logarithmic scale (bottom) of two instances from the same visual odometry section: (a,c) without preprocessing the input signal and (b,d) with offset elimination and filtering of the input signal. Note how in (b) the power peak at the step frequency (2 Hz) is observable and the highest in the interval of feasible step frequencies. Signal segments have been copied three times to make visible the difference in the discontinuity between the two instances.

trajectory grows linearly with N, since before scaling one section we need to get the new N unscaled camera poses from the SLAM algorithm. Thus, in summary, for the choice of N we must reach a trade-off between the accuracy of the DFT and the frequency with which the scale factor is updated.

Given the spectrogram $\Gamma_{d,k}(f_m)$, the step frequency $f_{st,k}$ is estimated as:

$$f_{st,k} = \underset{f_m \in [f_{st}^-, f_{st}^+]}{\arg\max} \; \Gamma_{d,k}(f_m). \tag{3.17}$$

3.4.3. Detection of non-walking situations

At this point the method to estimate the step frequency would return an estimate regardless of whether the user is walking or not. To discard erroneous estimates when the user is not walking we check that the spectral power $\bar{P}\left(f_{st,k}\right)$ of the computed step frequency to be consistent with a range of feasible oscillation amplitudes during walking bounded by a A_z^+ and A_z^- . Applying the Parseval's theorem, which states that the energy of a signal is preserved in the frequency domain, we can approximate the energy $\bar{P}$ of the signal corresponding to the body oscillation as:

$$\bar{P}\left(f_{st,k}\right) = 2\frac{F_s}{N} \sum_{m=m^-}^{m^+} \Gamma_d\left(f_{m,k}\right), \tag{3.18}$$

with $m^- = \text{round}\left(N\frac{f_{st}-\Delta f}{F_s}\right)$, $m^+ = \text{round}\left(N\frac{f_{st}+\Delta f}{F_s}\right)$, and where $f_{st,k}$ is the estimated step frequency, Γ_d is the discretised spectrogram of the z-component of the camera poses, F_s is the

sampling frequency of the camera, N the camera poses in the analysed section and Δf the frequency interval centred at $f_{st,k}$ along which the energy is spread along.

Since the power spectral density is computed for the unscaled z-component of the visual odometry, the computed power must be scaled by multiplying it by the square of the current mean scale factor $\bar{d}_k$. Thus, assuming that the body oscillation is sinusoidal, the condition for the spectral power consistency of the step frequency yields:

$$\frac{1}{2}A_z^{-2} \leq \bar{d}_k^2 \bar{P}\left(f_{st,k}\right) \leq \frac{1}{2}A_z^{+2}. \tag{3.19}$$

If this condition is not fulfilled we should choose another strategy. For example if $d_k^2 \bar{P}\left(f_{st,k}\right) \leq \frac{1}{2}A_z^{-2}$ we may assume that the person is stopped and then avoid updating the scale factor.

3.5. Dynamic scale update

Having a walking speed estimate and assuming that it is affected by Gaussian noise, the maximum likelihood scale factor for section k could be straightforwardly computed by $d_k = \frac{V_{walk,k}}{\mu_{V,k}}$, where $\mu_{V,k}$ is the average dimensionless speed of the camera poses in section k. However, given the empirical method for the walking speed estimation and the possible variability of the SLAM velocity along N frames, we propose to use a probabilistic filter for the computation of the scale factor. The purpose is to provide robustness against spurious estimations of the walking speed.

After computing the scale factor, the poses and the features measured in current section are scaled using a recursive approach to take care of the scale drift with respect to previous sections.

3.5.1. Particle Filter for scale factor tracking

For the design of the probabilistic filter we consider a dynamic system whose state $\mathbf{s}_k$ is composed by the magnitude of the SLAM velocity $V_{SLAM,k}$ and the decimal logarithm of the scale factor $\lambda_k = \log_{10}(d_k)$.

$$\mathbf{s}_k = \begin{bmatrix} V_{SLAM,k} \\ \lambda_k \end{bmatrix}. \tag{3.20}$$

Taking the logarithm has the advantage to naturally restrict the scale factor to positive values allowing to model its uncertainty and noise with additive Gaussian distributions. As a consequence all the non-linearities of the model will be encapsulated in the measurement function.

To track the scale factor, a particle filter with Sampling Importance Resampling is designed [Gordon et al., 1993]. The use of the particle filter is encouraged over an EKF due to its ability to deal with high uncertainty priors of the scale factor which would involve a large linearisation error of the measurement function. Also, since the system state is composed only by 2 variables, the major drawback of the exponential growth of the number of required particles with the number of state variables is negligible.

Hence the state of the system in each section k is approximated by a set of particles:

$$\mathcal{S}_k = \left\{ (\mathbf{s}_k^{(L)}, \omega_k^{(L)}) \mid L = 1, 2, ..., P \right\}, \tag{3.21}$$

where P is the number of particles and $\mathbf{s}_k^{(L)}$ and $\omega_k^{(L)}$ are respectively the state vector and the resampling weight of particle L.

The particles are initialised such that the initial values of $\lambda_0^{(L)}$ are drawn from a Gaussian distribution $\lambda_0 \sim \mathcal{N}(0, \sigma_0)$, where σ_0 is a parameter related to the orders of magnitude in the scale being scoped out.

In the predictive part of the particle filter, particles are sampled down by a proposal distribution $p(\mathbf{s}_k|\mathbf{s}_{k-1})$:

$$\mathbf{s}_k^{(L)} \sim p(\mathbf{s}_k|\mathbf{s}_{k-1}^{(L)}). \tag{3.22}$$

In our system the sampling of the proposal distribution includes both the update of the non-dimensional speed, which is taken as a control input coming from the visual odometry, and the possible drift in the scale. This is encoded in the following equations:

$$V_{SLAM,k}^{(L)} = \mu_{V,k} + n_v^{(L)} \tag{3.23}$$

$$\lambda_k^{(L)} = \lambda_{k-1}^{(L)} + n_\lambda^{(L)}, \tag{3.24}$$

with $n_v^{(L)} \sim \mathcal{N}(0, \sigma_{V,k})$ and $n_\lambda^{(L)} \sim \mathcal{N}(0, \sigma_{drift})$, and where $\mu_{V,k}$ and $\sigma_{V,k}$ are the averaged speed and the corresponding standard deviation of the last set of N SLAM camera poses used for spectral analysis, and σ_{drift} is a prior for the standard deviation of the scale drift between two consecutive sections, which is modelled as Gaussian noise.

To incorporate the walking speed information into the state estimate, first we need to define a measurement function $h_V(\mathbf{s}_k^{(L)})$ which predicts the walking speed measurement for each particle L.

$$h_V(\mathbf{s}_k^{(L)}) = V_{SLAM,k}^{(L)} 10^{\lambda_k^{(L)}}. \tag{3.25}$$

To prevent from later accepting spurious estimates of the walking speed, first we take the 95% confidence interval of the histogram of the predicted measurements by eliminating the samples at the tails, thus reducing the number of particles from P to P'.

The remaining samples are weighted by a probability density function $p(V_{walk,k}|\ \mathbf{s}_k^{(L)})$, which captures the statistics of the speed estimate. Assuming that it is affected by Gaussian noise of zero mean and a standard deviation σ_{Vwalk} to be set up empirically, weights are computed as:

$$\omega_k^{(L)} = p(V_{walk,k}|\ \mathbf{s}_k^{(L)}) = \phi\left(\frac{V_{walk,k} - h_V(\mathbf{s}_k^{(L)})}{\sigma_{Vwalk}}\right), \tag{3.26}$$

where $\phi(z)$ is the probability density function of the standard normal distribution.

Then a consistency test is carried out to verify that the weighted particles lie in the 95% confidence interval of the probability distribution of the measurement model, i.e.:

$$0.95 = P\left(-1.96 \leq \frac{V_{walk,k} - h_V(\mathbf{s}_k^{(L)})}{\sigma_{Vwalk}} \leq 1.96\right). \tag{3.27}$$

If every particle fails this test, the particle filter iteration ends here. Otherwise the weights of the particles are normalised:

$$\hat{\omega}_k^{(L)} = \frac{\omega_k^{(L)}}{\sum_{p=1}^{P'} \omega_k^{(p)}}, \tag{3.28}$$

and the set of particles $\mathcal{S}_k$ is resampled by drawing P particles from a multinomial distribution $Mult(P, \hat{\omega}^{(1)}, ..., \hat{\omega}^{(P')})$.

Then for a given section k, the scale factor is obtained by computing the geometric mean of the scale values in the particle set $\mathcal{S}_k$. That is:

$$\bar{\lambda}_k = \frac{\sum_{i=1}^{P} \lambda_k^{(i)}}{P} \tag{3.29}$$

$$\bar{d}_k = 10^{\bar{\lambda}_k}. \tag{3.30}$$

3.5.2. Scaling of the trajectory

This scale factor must be applied to the state variables with length dimensions, position and velocity. These are encapsulated in a vector:

$$\mathbf{x}_k^{C_d}(n) = \begin{bmatrix} \mathbf{r}_{W,k}^{C}(n) \\ \mathbf{v}_{W,k}^{C}(n) \end{bmatrix}. \tag{3.31}$$

To ensure the continuity in position and velocity, each vector $\mathbf{x}_k^{C_d}(n)$ contained in the current section k is scaled using the following recursive formula:

$$\breve{\mathbf{x}}_k^{C_d}(n) = \breve{\mathbf{x}}_{k-1}^{C_d}(N) + \bar{d}_k \left[\mathbf{x}_k^{C_d}(n) - \mathbf{x}_{k-1}^{C_d}(N)\right] \qquad k = 2, 3, ... \tag{3.32}$$

$$\breve{\mathbf{x}}_1^{C_d}(n) = \bar{d}_1 \mathbf{x}_1^{C_d}(n), \tag{3.33}$$

where $\breve{\mathbf{x}}_k^{C_d}(n)$ contains the scaled camera position and velocity estimates.

3.5.3. Scaling of landmarks

The set of landmarks to be updated or initialised in the scaled map estimate is given by all the point landmarks marginalised from the EKF during current section and the landmarks matched in the EKF update of the last pose of the section:

$$\mathcal{M}_{\mathcal{S},k} = \left[\mathcal{E}_{(k-1)N+1}^{p}, \cdots, \mathcal{E}_{kN}^{p}, \mathcal{T}_{kN}^{p}\right]. \tag{3.34}$$

The scale of each landmark is set to the one of its anchor pose in IDP:

$$\breve{\mathbf{y}}_W^{(j)} = \breve{\mathbf{r}}_W^{C(j)} + \bar{d}_{\kappa(j)} \left(\mathbf{y}_W^{(j)} - \mathbf{r}_W^{C(j)}\right) \tag{3.35}$$

where the function $\kappa : \mathbb{N} \to \mathbb{N}$ which establishes a surjective mapping from a landmark index j to the index of the section containing the anchor pose $\mathbf{r}_W^{C(j)}$ of the landmark.

3.5.4. Real time implementation and reduction of the update delay

The complete scaling algorithm is implemented in a new thread within the real-time monoSLAM C++ application [Davison et al., 2007], working in parallel with the main SLAM thread. After each EKF iteration, the main thread stores the last camera pose and the corresponding landmarks in two shared buffers. When the buffer with the camera poses is fully updated with N poses, the main thread triggers the scaling thread. After executing the scaling algorithm, the scaled trajectory is updated by adding the recently scaled camera poses.

In spite of the real-time operation, the update of the scaled odometry estimate is delayed due to the time t_{DFT} it takes to fill the buffer of camera poses with the N states needed to perform the DFT. One way to reduce this delay is to reduce N, at the expense of reducing the accuracy of the DFT to compute the step frequency.

Alternatively we propose to use a sliding window, updating only one fraction N_f of the buffer of camera poses at a time. Thus the number of camera poses used for the spectral analysis remains N by reusing poses from previous sections, while the amount of scaled camera poses per section is reduced to N_f. The time required to update the scaled trajectory with the new N_f poses is notated as t_{upd}. The complete scaling method is described in Algorithm 1.

Algorithm 1 Complete Visual Odometry Scaling algorithm

Require: $\mathbf{x}_k^C(1..N)$, $\mathcal{S}_{k-1}$
Ensure: $\breve{\mathbf{x}}_k^C(1..N_f)$, $\mathcal{S}_k$
  //Notation
  $\mathbf{x}_k^C(n) =$ n^{th} unscaled camera state of section k
  $\breve{\mathbf{x}}_k^C(n) =$ n^{th} scaled camera state of section k
  $N =$ # input camera states
  $N_f =$ # output/new camera states
  $\mathcal{S}_k =$ Set of particles for the particle filter
  //Algorithm
  $k = 0$; $[\mathcal{S}_0]$ = Initialise particles ()
  while Not end of sequence **do**
    $k = k + 1$
    Wait for new $\breve{\mathbf{x}}_k^C(1..N)$ from monoSLAM
    $[z_k(1..N),\ \mu_{V,k},\ \sigma_{V,k}]$ = Extract z-comp & mean speed $\left(\mathbf{x}_k^C(1..N)\right)$
    $[z_k(1..N)]$ = High Pass Filter $(z_k(1..N))$
    $[f_m,\ \Gamma_{d,k}]$ = Spectrogram $(z_k(1..N))$
    $[f_{st,k},\ \Gamma_{d,k}\,(f_{st,k})]$ = Estimate Step Frequency $(f_m,\ \Gamma_{d,k})$
    if Step freq power is consistent $(d_{k-1},\ \Gamma_{d,k}\,(f_{st,k}))$ **then**
      $[V_{walk,k}]$ = Walking speed model $(f_{st,k})$
      $[\mathcal{S}_k]$ = Sample Proposal Distribution $(\mathcal{S}_{k-1},\ \mu_{V,k}, \sigma_{V,k})$
      $[\mathcal{S}_k]$ = Weighting and Resampling $(\mathcal{S}_k,\ V_{walk,k})$
      $[d_k]$ = Compute mean scale factor $(\mathcal{S}_k)$
    else
      $d_k = d_{k-1}$; $\mathcal{S}_k = \mathcal{S}_{k-1}$
    end if
    if k=1 **then**
      $\left[\breve{\mathbf{x}}_1^C(1..N)\right]$ = Scale Section $\left(d_1,\ \mathbf{x}_1^C(1..N)\right)$
    else
      $\left[\breve{\mathbf{x}}_k^C(1..N_f)\right]$ = Scale Section $\left(d_k,\ \mathbf{x}_k^C((N-N_f+1)..N)\right)$
    end if
  end while

3.6. Experiments

For the validation of our proposal we use three different cameras in our experiments, each one with varying geometries, resolution and frame rates.

The first experiments have been carried out with a catadioptric omnidirectional camera VS-C14U-80-ST model which consists on a mirror and a Sentech UltraSmall STC-MC83USB camera with a resolution of 1024x768, at frame rate of 15 fps, which is mounted on a helmet to be carried by a human operator.

In the last experiments we used the wearable GoPro Hero and Sony Action Cam, in order to verify the applicability of our method also to conventional cameras, whose narrower field of view is likely to provide less accurate motion estimates than omnidirectional cameras [Rituerto et al., 2010], which could affect the perception of the body oscillations. Also, their ability to be attached to several body parts, allows us also to evaluate our method when the camera is worn on the chest, which, as in the case of the head, also shows the characteristic oscillatory motion while walking.

Trajectory estimates in the experiments have been refined by applying loop closure optimisation when possible, where the loop constraints were determined as explained in 3.3.2. Frames for loop closure were selected manually, since the loop detection problem is out of the scope of this work. Also this avoids uncertainties due to possible errors in automatic loop detection, allowing for a fair comparison of the scale drift removal capabilities between our method and the proposed in [Strasdat et al., 2010].

The Ground Truth for the experiments has been obtained from the Google Maps satellite view using the distance measurement tool. To compare numerically the Ground Truth and the scaled odometry estimates, we parametrise both curves by the normalised arc length or accumulated distance ξ which spans from 0 (start) to 1 (end). Then, given the Ground Truth trajectory $\mathbf{t}_{GT}$, the error for a given pose is computed as follows:

$$err^{(i)} = \left|\left|\mathbf{t}_{VO}^{(i)} - \mathbf{t}_{GT}(\xi(\mathbf{t}_{VO}^{(i)}))\right|\right|. \tag{3.36}$$

3.6.1. Parameter tuning

Our algorithm presents a series of parameters that have to be tuned previously. The number of particles used in the particle filter is set to $P = 5000$ for all the experiments. The standard deviation of the distribution for the initial logarithmic scale factor is set to $\sigma_0 = 1$, which allows us to cover an uncertainty interval for the scale factor between 10^{-2} and 10^{2} with a 95% of confidence. The setup of the uncertainty parameters for the scale drift σ_{drift} and $\sigma_{V_{walk}}$ is heuristic, trying to reach a trade-off between robustness and rapid response to sharp scale factor changes. We use $\sigma_{drift} = 0.1$ and $\sigma_{V_{walk}} = 0.2$ m/s. This last value lies within the interval of standard deviations for the feasible step frequencies range, as it can be proved by computing the derivative of (3.11):

$$\sigma_{V_{walk}} = \alpha\beta f^{\beta-1} H \sigma_f, \tag{3.37}$$

with $\sigma_f = \frac{1}{t_{DFT}}$; and taking $t_{DFT} = 3$ s and evaluating at the limits of the feasible step frequencies. Note that though for larger t_{DFT}, $\sigma_{V_{walk}}$ should be theoretically lower, keeping it constant overcompensates the fact of loosing accuracy due to a lower robustness to changes in speed when using larger windows.

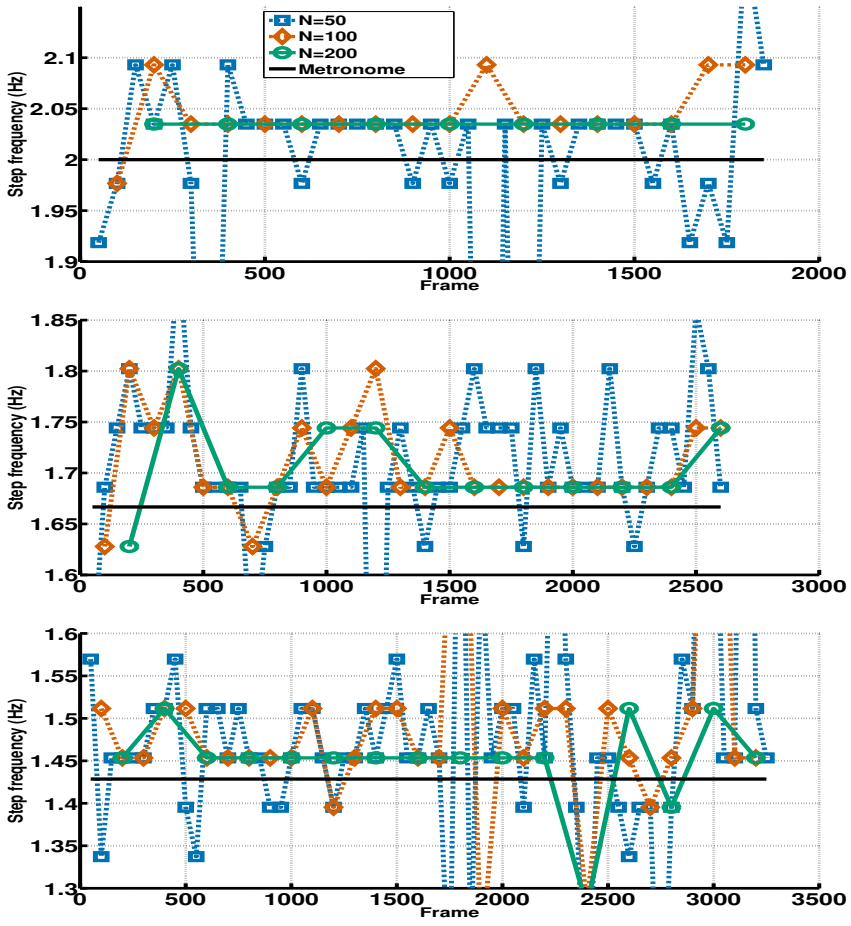


Figure 3.6: Spectral analysis along the same path at the three step frequencies of 2 (top), 1.67 (centre) and 1.43 Hz (bottom) with different section sizes. A higher section size implies a more reliable frequency estimate.

3.6.2. Testing the ability to measure the step frequency

For the first experiment we acquired three image sequences with the catadioptric camera walking along the same path of 230 meters with three different step frequencies, which were set up by a metronome with 0.01 seconds of resolution. The user's pace was fixed to 0.50, 0.60 and 0.70 seconds per beat for each sequence, which translates in step frequencies of 2, 1.67 and 1.43 Hz respectively. The purpose of this experiment is testing the ability of the method described in Section 3.4.2 to estimate the step frequency just from the visual odometry signal.

We select different section dimensions of $N = 50$, $N = 100$ and $N = 200$ camera poses. To compute the DFT we use the FFTW (Fast Fourier Transform West) C library [Frigo and Johnson, 2005].

In Fig. 3.6 it is shown that the dominant frequency obtained by spectral analysis closely approximates the step frequency fixed by the metronome. Among the considered setups, $N = 200$ results in better accuracy and less outliers in the estimation of the step frequency. However, as argued in Section 3.4.2, a window of this length might not be the optimal for the performance of our method as a whole. This issue is addressed in the next experiments.

Table 3.1: Estimation error for combinations of t_{DFT} and t_{upd} for the experiment with changes in pace.

t_{DFT} [s]	t_{upd} [s]	Mean error[m]	Maximum error[m]	Relative mean error
12	12	12.54	28.83	1.42%
12	3	10.43	18.65	1.18%
3	3	9.26	21.24	1.05%

3.6.3. System robustness under changes of pace

The sequence for this experiment was acquired with the catadioptric camera along a path of 886 m containing a variety of challenging situations like changes of pace, stops, stairs and walking along a narrow corridor. Transitions between these situations have been ticked accordingly in the frame when they take place.

The objective is evaluating the robustness of our method under these conditions, and select the optimal number of poses to be taken for scaling at each iteration. Instead of using the number of poses N and N_f to define the window sizes in the experiments, from this point we will take their temporal length t_{DFT} and t_{upd}, as it generalises better for different frame rates.

In this experiment we evaluate the use of a dynamic window approach to scale the trajectory sections. We have tested 3 different alternatives. In two alternatives t_{DFT} is set to 12 seconds and t_{upd} is varied to compare the performance of static ($t_{upd} = 12$ seconds) and dynamic ($t_{upd} = 3$ seconds) windows. The third alternative consists in using a static window with $t_{DFT} = t_{upd} = 3$ seconds.

Fig. 3.7 (top) reveals how the step frequency computed from the raw unscaled visual odometry varies accordingly with the pace of the walker. Note that for lower t_{DFT} the step frequency estimate is less accurate and oscillates more, though the global tendency in the pace is still captured. In Fig. 3.7 (bottom), during the long stop, the spectral power shows a violation of the consistency condition which leads to ignore the erratic estimation of the step frequency. During the short stop, the violation of the consistency is not observed with $t_{DFT} = 12$ s, due to the masking effect of the poses corresponding to a walking state. In the case of going upstairs the power peak is noticeable but not as clear as in the stop to establish a clear upper limit for the consistency condition.

Fig. 3.8 (top) shows the dynamic estimation of the scale factor. It can be noticed that scale drift can occur in two different ways. On the one hand it can be a gradual drift as occurs at the start of the sequence or during the corridor. Drift at the start can be due to the still highly uncertain depth estimate of the tracked points during initialisation, while on the corridor it is the gradual substitution of points in spacious areas by points on the walls of the narrow corridor what causes the drift. On the other hand drift can occur sharply if landmarks which act as an anchor for the scale are suddenly lost. This might be caused for example by occlusions by dynamic elements or sudden camera accelerations, as occurs when restarting the walk after a stop.

Fig. 3.9a shows a comparison between the scaled trajectories obtained for each considered setup. Loops have been closed at the end and at the middle of the path. Numerical comparison with the Ground Truth is detailed in Table 3.1. It is shown that taking $t_{DFT} = 3$ s offers an slightly better scaled estimate of the visual odometry with a mean relative error of 1.05% over the trajectory length. This demonstrates the convenience of losing a bit of accuracy in the DFT using short windows (but large enough to capture the body oscillations), in order to get a higher

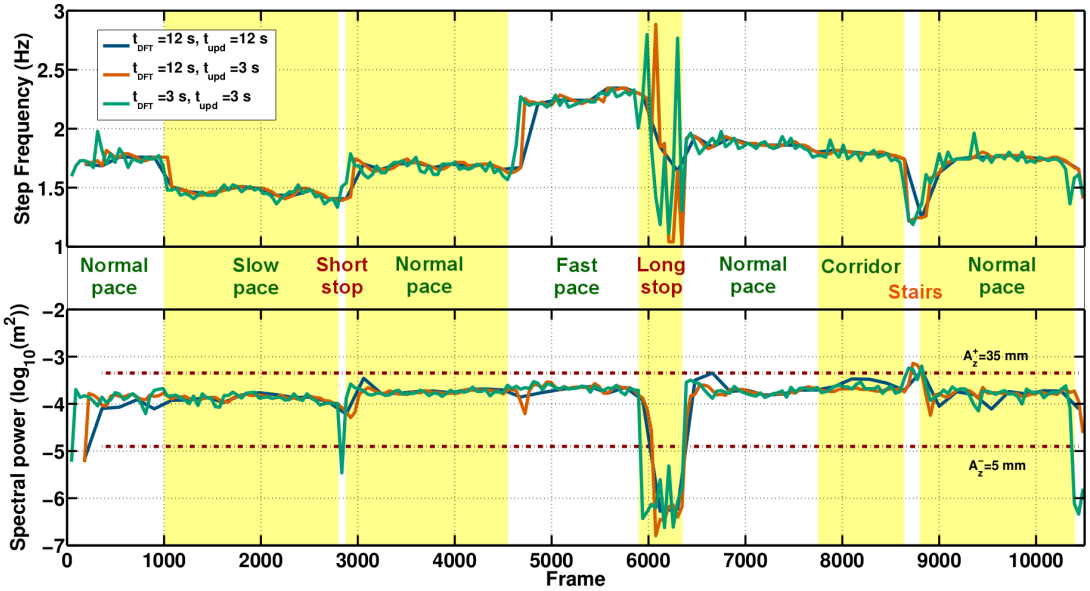


Figure 3.7: Evolution of the step frequency estimate (top) and its corresponding spectral power (bottom) in the changing pace experiment. Consistency bounds are violated when estimate does not correspond to a walking step frequency.

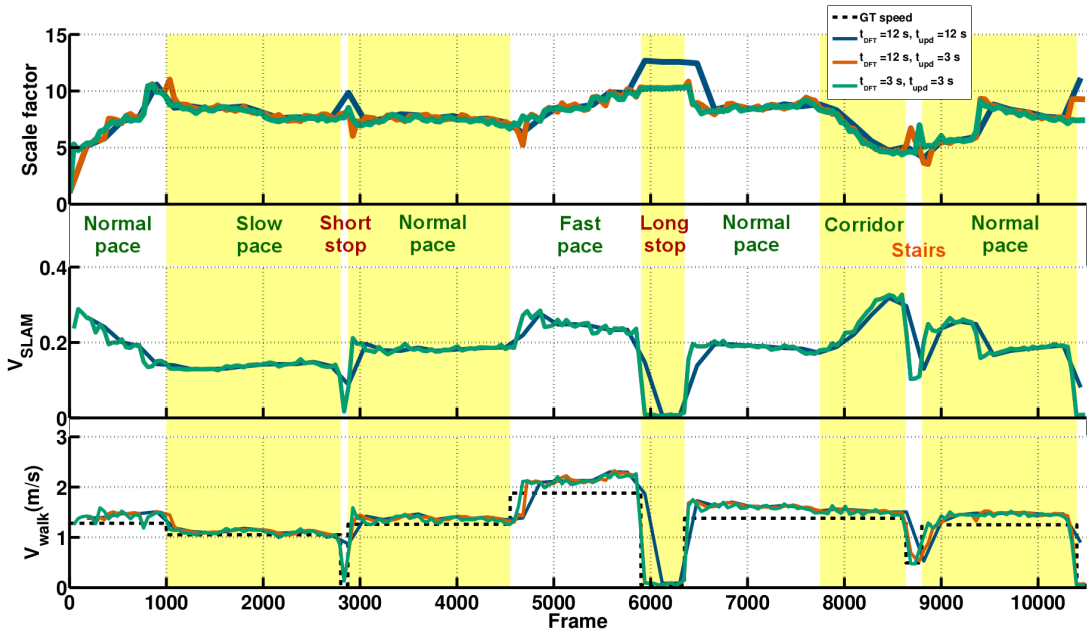


Figure 3.8: Evolution of (top) the scale factor, (middle) the non-dimensional speed and (bottom) the estimated real walking speed in the changing pace experiment. Ground Truth speed was assumed constant for a given pace and computed from Google Maps Ground Truth and the time difference between frames.

update rate of the scale and a faster detection of stops or changes in pace (Fig. 3.7).

The great improvement of all the considered cases with respect to the raw odometry estimate shows the ability of our approach not only to compute the scale but also to remove the scale drift, which is reflected on the deformation of the raw estimate.

Comparison with state-of-the-art scaling approach

Once we pick the configuration with $t_{DFT} = t_{upd} = 3$ s we compare our method with the approach described in [Strasdat et al., 2010]. This approach aims to remove the scale drift in the trajectory during loop closure optimisation by expressing the camera poses and both the odometry and loop closure constraints as similarity transforms (Sim(3)) instead of as conventional rigid body motions (SE(3)). Note that this approach tackles the scale drift problem but it does not compute the absolute scale of the odometric estimate. Thus, for a fair comparison, each of the finally obtained trajectories by the different evaluated methods is rescaled so that its total distance is the same as that of the Ground Truth. We evaluate 4 different solutions: standard loop closure optimisation in SE(3) of the raw estimate (unscaled), scale drift correction with loop closure optimisation in Sim(3) (optimSim3 [Strasdat et al., 2010]), standard loop closure optimisation in SE(3) of our scaled estimate (dynScale) and loop closure optimisation in Sim(3) of our scaled estimate (dynScale + optimSim3 [Strasdat et al., 2010]). Qualitative and numerical results shown in Fig. 3.9b and Table 3.2 respectively. It can be observed that both our method and [Strasdat et al., 2010] clearly improve the raw estimate, though our method provides better accuracy. Note also that the combination of both methods slightly improves our raw proposal providing the most accurate estimate.

This result is expected since while our approach estimates the scale every 3 seconds, in [Strasdat et al., 2010] it is only possible to observe it at loop closures. However it can be noted that, as both approaches obtain scale drift information from different sources, they can combine well providing a more accurate estimate together than both alone. A complete view of the best trajectory estimate and the points of the scene is shown in Fig. 3.9c.

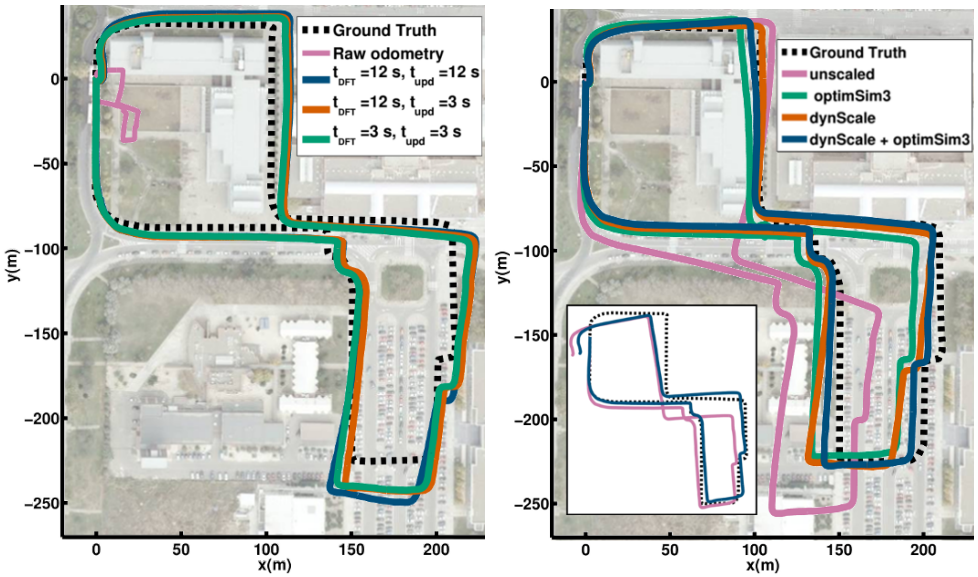
3.6.4. Indoor experiment

In the second experiment we test our approach with the catadioptric camera in an indoor environment [Murillo et al., 2012] along a path of 464 m under a freely chosen gait, keeping the tuning used in previous experiment. Our approach is able to correct a significant scale drift of 200% taking place at the start of the trajectory (Fig. 3.10b), providing a final odometry estimate (Fig. 3.13a) with a mean error of 4.69 m (1.01% over the trajectory length). Note in Fig. 3.10a the peak in the spectral power occurring at the two stair parts in the trajectory.

The comparison with [Strasdat et al., 2010] shown in Fig. 3.13 and Table 3.3 is similar to the previous experiment. Our approach alone outperforms [Strasdat et al., 2010] alone, and both together provide a better estimate.

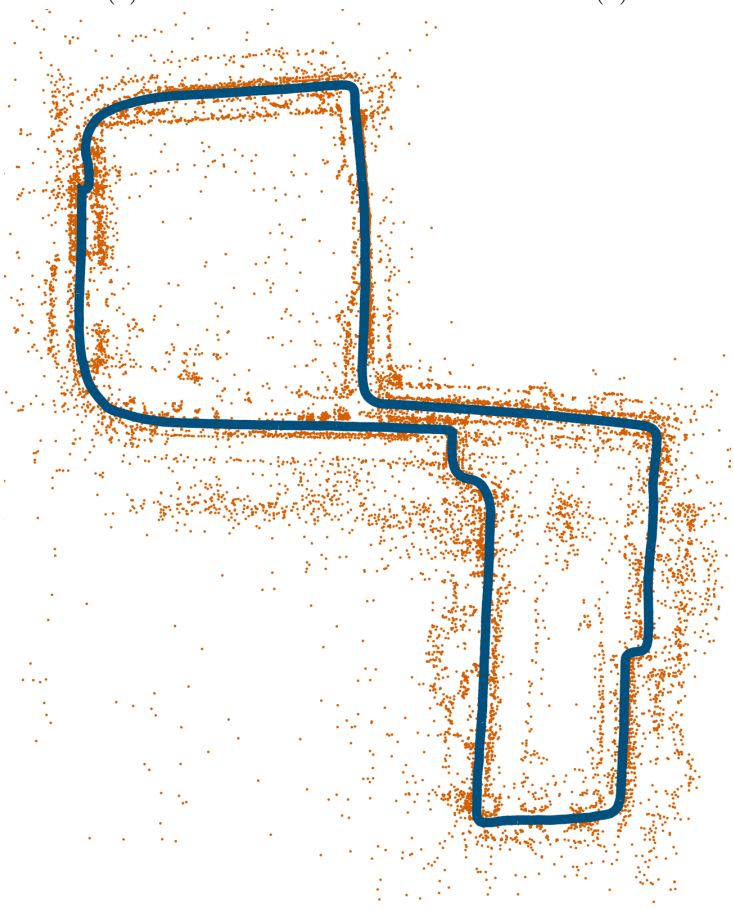
3.6.5. GoPro experiment

For this experiment we acquired an image sequence with a GoPro camera attached to the user's head. The user walked at steady pace along a path of 410 m. the camera resolution and frame rate were set to 1280x720 and 60 fps respectively. The tuning of our scaling algorithm is the same as the one used in the previous experiments with the omnidirectional camera, taking again $t_{DFT} = t_{upd} = 3$ s.



(a)

(b)

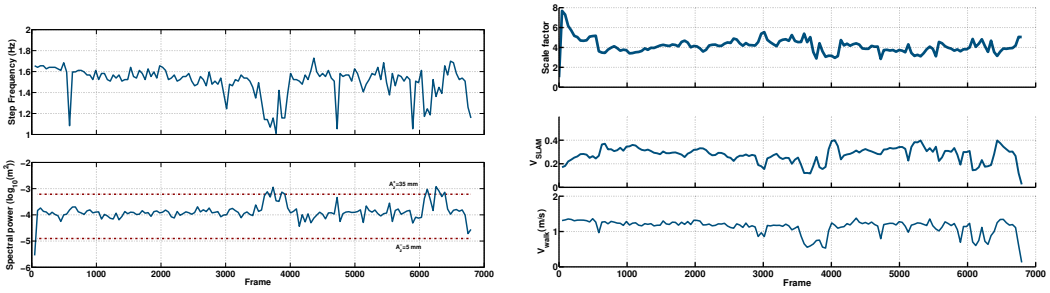


(c)

Figure 3.9: Changing pace experiment. (a)Scaled trajectory estimates for different setups of t_{DFT} and t_{upd}. (b) Result of using different approaches for scale drift removal, with all trajectories rescaled to the Ground Truth's scale. Estimates of our scaled and raw trajectory estimates prior to loop closure are shown in the small view, in which the raw estimate has been rescaled for better visualisation. (c) Scaled trajectory and scene points obtained with the most accurate approach.

Table 3.2: Estimation error for different scaling and optimisation combinations for the experiment with changes in pace.

Method	Mean error[m]	Maximum error[m]	Relative mean error
unscaled	30.47	71.10	3.43%
optimSim3 [Strasdat et al., 2010]	11.29	30.02	1.27%
dynScale (ours)	5.35	19.34	0.60%
dynScale (ours) + optimSim3 [Strasdat et al., 2010]	4.04	9.68	0.46%



(a) (b)

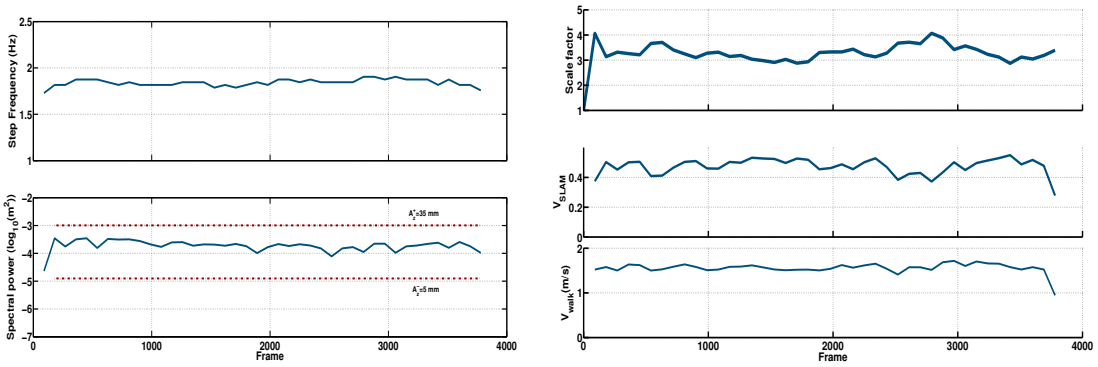
Figure 3.10: Indoor experiment with catadioptric camera. (a) Evolution of the step frequency estimate (top) and its corresponding spectral power (bottom), (b) Evolution of (top) the scale factor, (middle) the non-dimensional speed and (bottom) the estimated real walking speed.

Table 3.3: Estimation error for different scaling and optimisation combinations for the indoor experiment with the catadioptric camera.

Method	Mean error[m]	Maximum error[m]	Relative mean error
unscaled	10.82	22.90	2.33%
optimSim3 [Strasdat et al., 2010]	6.47	12.82	1.39%
dynScale (ours)	3.77	10.37	0.81%
dynScale (ours) + optimSim3 [Strasdat et al., 2010]	3.21	9.37	0.69%

In Fig.3.14a it is shown both the evolution of the step frequency and the spectral power of the head oscillation. Note that there exists a peak in the spectral power above the higher limit for oscillation amplitude, which corresponds to the stairs part of the trajectory. An outlier can be observed in the estimation of the step frequency, though, as it can be noted in Fig. 3.14b, the particle filter successfully rejects this measure and both scale factor and walking speed estimates are not affected. As in the previous experiment the scale smoothly drifts as we enter in the narrow corridor (between frames 2200 and 3200).

The final odometric estimate in Fig. 3.15a shows once again the competitiveness of our method, with a mean error of 1.79 m (0.44% over the total trajectory length). Note that even prior to loop closure, scale drift correction can be observed in the small view as the start and end points of the trajectory are almost coincident. In this experiment the loop was closed at the end of the sequence. As the camera view direction of the scene differs between the start and the end poses, and a camera with limited field of view is used, correspondences for loop closure can only be obtained in a small portion of the matched images, which result in a loss of precision of



(a) (b)

Figure 3.11: Camera-on-chest experiment. (a) Evolution of the step frequency estimate (top) and its corresponding spectral power (bottom), (b) Evolution of (top) the scale factor, (middle) the non-dimensional speed and (bottom) the estimated real walking speed.

the loop constraint.

As we can observe in Fig. 3.15b and Table 3.4, our approach outperforms alone clearly all the other options. The lack of improvement when combining optimSim3 with our method may be due to the limited common field of view between the loop closing frames. The final odometry and map view obtained with our approach is shown in Fig. 3.15c.

3.6.6. Sony Action Cam attached to the chest

In this experiment, we test the ability of our method for performing scale correction when the camera is not worn on the head. We used a Sony Action Cam with a resolution of 960x540 at 30 fps attached to the user's chest. The sequence was acquired in an indoor loop with a length of 187 m.

Fig. 3.11 show that the step frequency and the scale factor can be successfully estimated when the camera is worn on the chest, due to showing only a oscillatory motion during locomotion. The final reconstruction after loop closure (Fig. 3.12) shows the correction of a slight drift in the scale and that the estimate with the absolute scale is indeed computed.

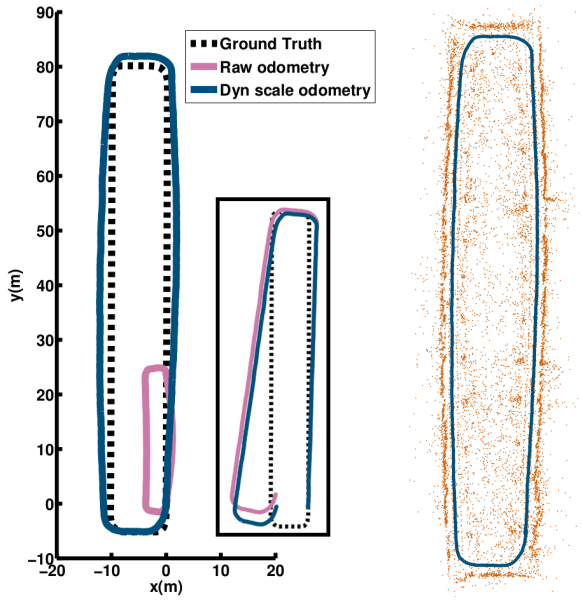
3.6.7. Analysis of the change in the walking model parameters

In this section we analyse how the scaled estimate of some of the previous experiments is affected by the variation of the walking model parameters α and β from their nominal values explicitly fitted for the user. We evaluate the effect both on the absolute scale and on the scale drift. A prior theoretical analysis is lead by naively assuming that:

$$d(t) = \frac{V_{walk}(t)}{V_{SLAM}(t)}. \tag{3.38}$$

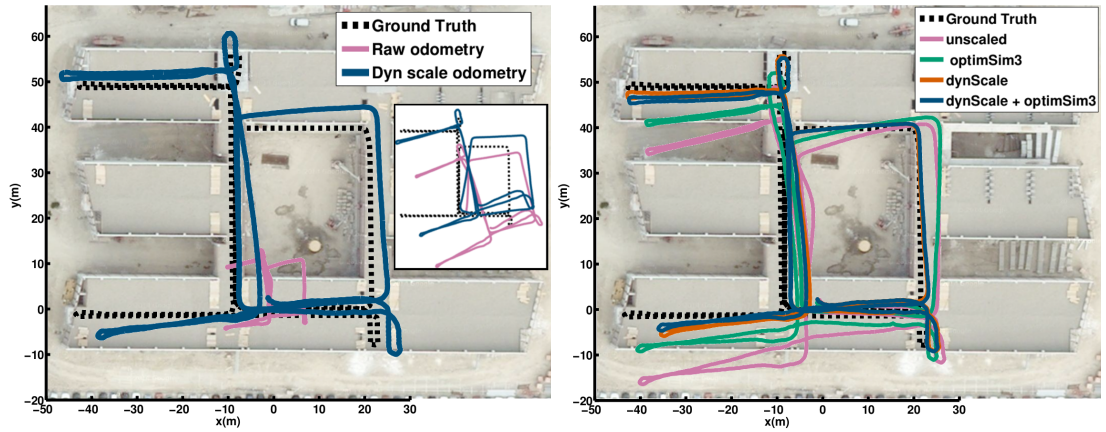
The change of the model parameters produce a variation of the estimated V_{walk}, which leads to a change in d. The change in the absolute scale is expressed by the following ratio:

$$r(t) = \frac{d(t)}{\hat{d}(t)} = \frac{V_{walk}(t)}{\hat{V}_{walk}(t)}, \tag{3.39}$$

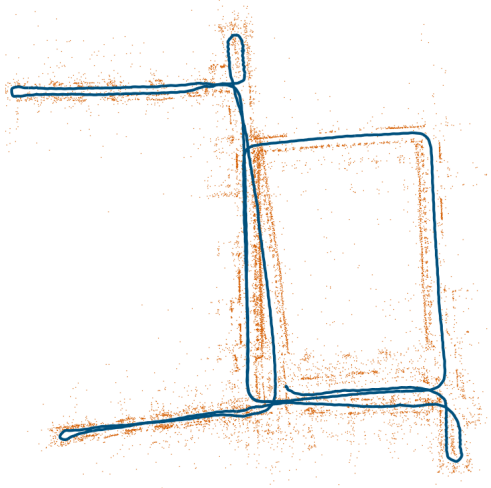


(a) (b)

Figure 3.12: Camera-on-chest experiment. (a)Trajectory estimate after loop closure with and without our scaling algorithm. Estimates prior to loop closure are shown in the small view, in which the raw estimate has been rescaled for better visualisation. (b) Scaled trajectory and scene points obtained with our approach.



(a) (b)

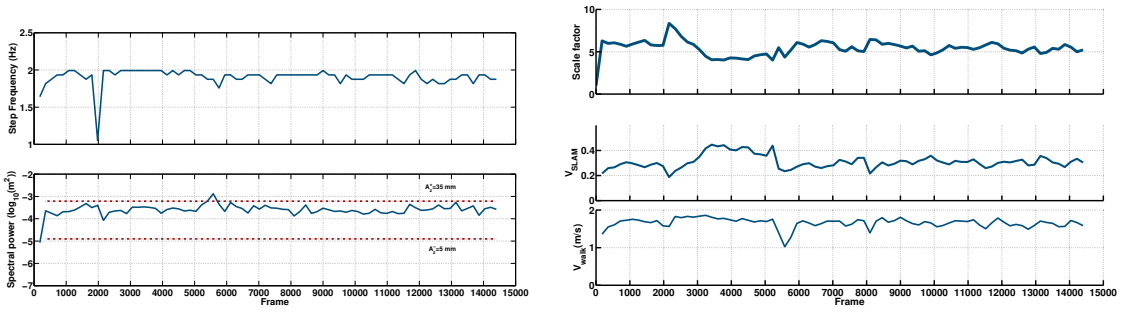


(c)

Figure 3.13: Indoor experiment with catadioptric camera. (a)Trajectory estimates after loop closure in our scaled estimate and the raw estimate. Estimates prior to loop closure are shown in the small view, in which the raw estimate has been rescaled for better visualisation. (b) Result of using different approaches for scale drift removal. Absolute scale of each estimate is fitted to the Ground Truth's. (c) Scaled trajectory and scene points obtained with our approach.

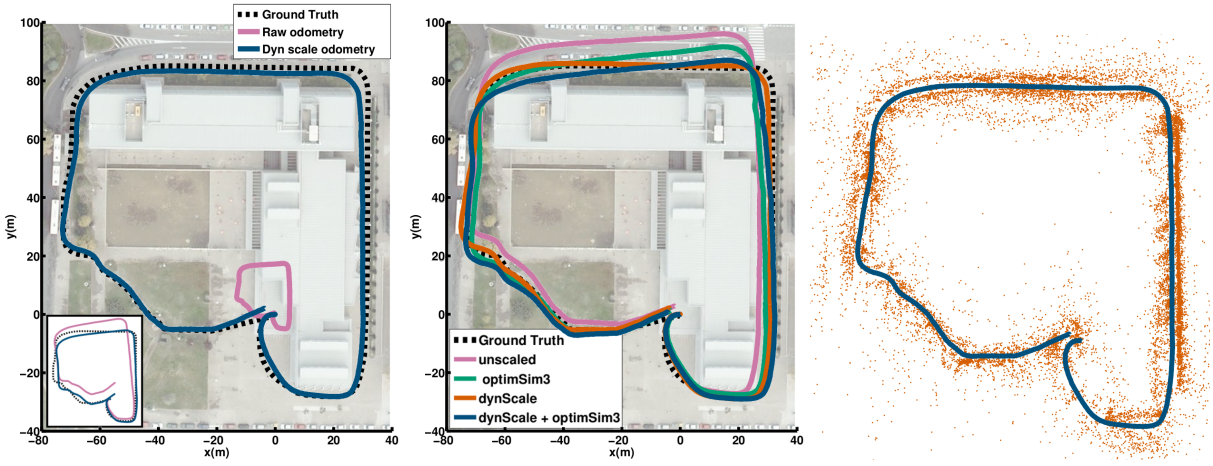
Table 3.4: Estimation error for different scaling and optimisation combinations for the GoPro sequence.

Method	Mean error[m]	Maximum error[m]	Relative mean error
unscaled	5.83	11.94	1.42%
optimSim3 [Strasdat et al., 2010]	2.81	7.40	0.69%
dynScale (ours)	1.54	4.46	0.37%
dynScale (ours) + optimSim3 [Strasdat et al., 2010]	2.53	5.89	0.62%



(a) (b)

Figure 3.14: GoPro experiment. (a) Evolution of the step frequency estimate (top) and its corresponding spectral power (bottom), (b) Evolution of (top) the scale factor, (middle) the non-dimensional speed and (bottom) the estimated real walking speed.



(a) (b) (c)

Figure 3.15: GoPro experiment. (a)Trajectory estimates after loop closure in our scaled estimate and the raw estimate. Estimates prior to loop closure are shown in the small view, in which the raw estimate has been rescaled for better visualisation. (b) Result of using different approaches for scale drift removal. Absolute scale of each estimate is fitted to the Ground Truth's. (c) Scaled trajectory and scene points obtained with our approach.

while the scale drift with respect to the nominal model is given by its first order time derivative $r'(t)$.

Let us first analyse how the variation of parameter α affects both $r(t)$ and $r'(t)$. Recalling (3.11) we get that:

$$r(t) = \frac{\alpha}{\hat{\alpha}} \qquad (3.40) \qquad\qquad r'(t) = 0. \qquad (3.41)$$

which means that a proportional change in the scale is to be expected if we vary α. Since α is a constant parameter, no drift is expected with respect to the nominal scaled estimate.

In the case of varying β we have:

$$r(t) = \frac{f^{\beta}}{f^{\hat{\beta}}} = f^{\Delta\beta} \qquad (3.42) \qquad\qquad r'(t) = \Delta\beta f^{\Delta\beta-1}\frac{\partial f}{\partial t}, \qquad (3.43)$$

which means first, that the change in the absolute scale will be higher with increasing $\Delta\beta$ and higher step frequencies and, secondly, that if there are changes in the user's pace, we might expect a drift in the scale higher for larger $\Delta\beta$.

For the experimental evaluation we used the changing pace and GoPro sequences. Each time we run our algorithm, we set one parameter to the nominal value fitted to our camera operator, while varying the other between our nominal value and 3 other options computed from average, lower limit and upper limit values of the parameters for the model proposed in [Grieve and Gear, 1966].

Results in Fig. 3.16 show that an *ad hoc* calibration for each user is crucial to get the absolute scale of the estimate. However, for just scale drift correction, which is more important in practice, it is shown that an accurate calibration of the walking model parameters is not critical, as long as extreme pace changes are avoided during walking. Note that the experimental observations confirm the predicted behaviour by the theoretical analysis.

3.7. Discussion

In this work we have presented a novel approach to provide estimates with the absolute scale of wearable odometric localisation systems using a single camera. Our proposal makes these hypothesis: first, the camera must be attached to a body part whose motion is mainly caused by the action of walking; secondly, the initial unscaled visual odometry estimate must be accurate enough to register the oscillations which take place during walking, and thirdly the roughness of the terrain on which the user moves is low enough not to mask the amplitude of the walking oscillations.

Our method has been thoroughly evaluated in a rich set of video sequences, combining indoor and outdoor environments and using many kinds of cameras, attached either to the head or to the chest of the user. In spite of this high variety in the conditions which the system has been tested on, our algorithm shows a good performance without requiring to be retuned for each experiment with the same user. Also we show that if the pace of the user does not change a lot during the path, the calibration of our system for a specific user is not critical in terms of scale drift correction.

We have compared our algorithm in our most challenging sequences against the scale drift correction method proposed in [Strasdat et al., 2010], where ours shows a better performance.

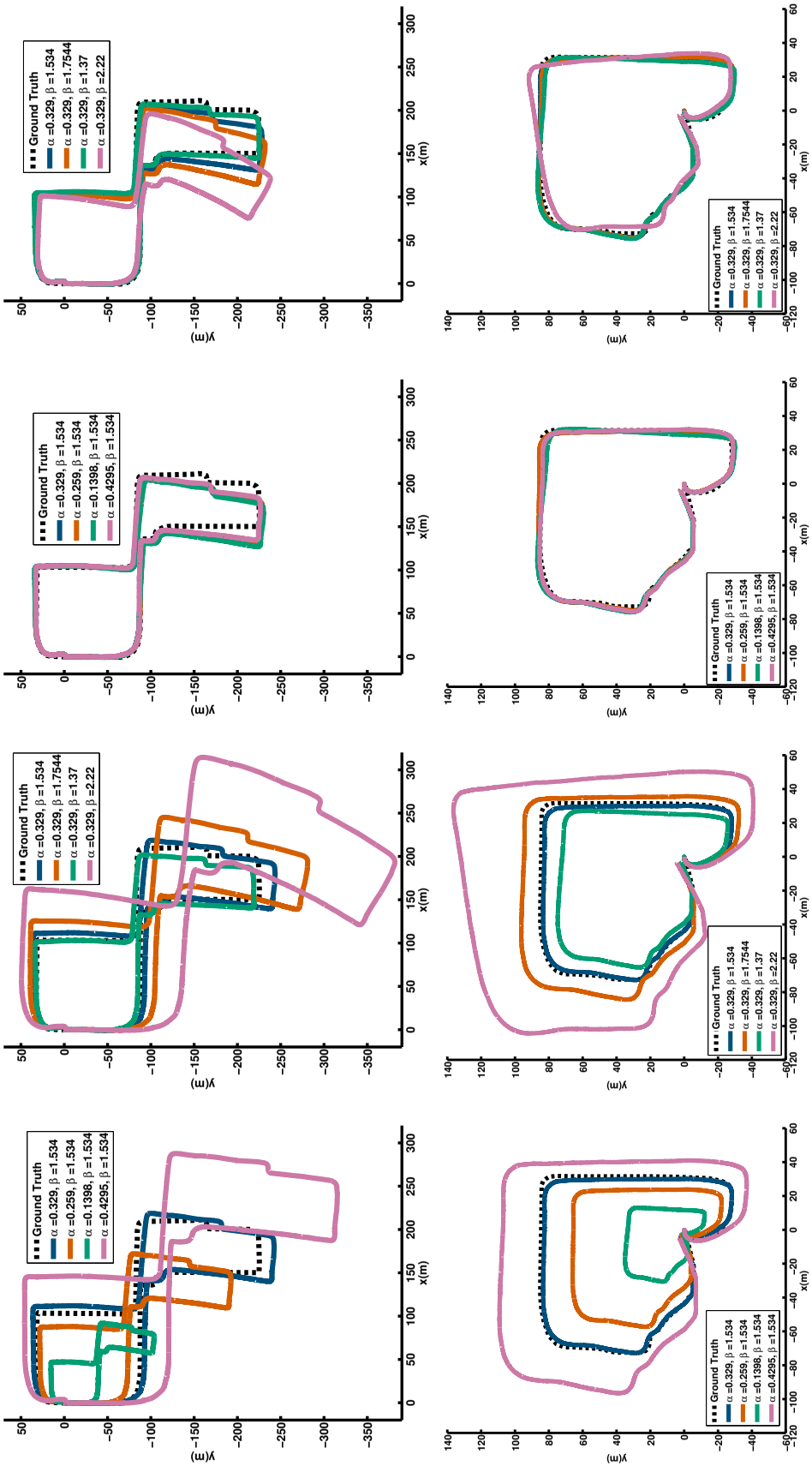


Figure 3.16: Variation of (a) the absolute scale with α, (b) the absolute scale with β, (c) the scale drift with α, (d) the scale drift with β. Results for the pace change sequence are shown in the first row. Results for the GoPro sequence are shown in the second row row. Note that a wrong α has no negative effect on scale drift correction, while a wrong β is only harmful if there are high changes in pace.

This gain in performance is due to the fact that our method corrects the scale dynamically every few seconds, instead of having to wait for a loop detection to obtain scale information as done in [Strasdat et al., 2010]. Note also, that as both methods extract the scale from different sources, they are not mutually exclusive. Indeed we have shown in the experiments, that he combination of both provides the better performance. In this sense, we expect also that the proposal presented in this work can also combine well not only with [Strasdat et al., 2010], but also some of other present and future methods for scale computation, in order to get more robust information about the real camera trajectory and 3D observed scene.

Chapter 4

Dense RGB-D Visual Odometry using Inverse Depth

In this chapter we present a dense visual odometry system for RGB-D cameras performing both photometric and geometric error minimisation to estimate the camera motion between frames. Contrary to most works in the literature, we parametrise the geometric error by the inverse depth instead of the depth, which translates into a better fit of the distribution of the geometric error to the used robust cost functions. To improve the accuracy we propose to use a keyframe switching strategy based on a visibility criteria between frames. For the comparison of our approach with state-of-the-art approaches we use the popular datasets from the TUM for RGB-D benchmarking as well as two synthetic datasets. Our approach shows to be competitive with state-of-the-art methods in terms of drift in meters per second, even compared to methods performing loop closure too. When comparing to approaches performing pure odometry like ours, our method outperforms them in the majority of the tested datasets. Additionally we show that our approach is able to work in real time and we provide a qualitative evaluation on our own sequences showing a low drift in the 3D reconstructions. We have implemented this method within the scope of PCL (Point Cloud Library) as a branch of the code for large scale KinectFusion, where the original ICP system for odometry estimation has been completely substituted by our method. A PCL fork including the modified method is available for download .

4.1. Introduction

The recent advent of new RGB-D sensors has aroused great interest in the development of visual odometry and SLAM systems. Their cheapness and their ability to provide dense depth measurements of the environment in contrast to traditional stereo cameras makes them quite appealing to address not only localisation and mapping but also many other problems for which monocular systems are typically used. The main limitation is their use being limited to indoor environments.

Some of the first systems using RGB-D sensors for SLAM [Henry et al., 2010], [Huang et al., 2011], [Endres et al., 2012] tended to adapt the sparse feature based approaches from monocular vision, using the depth information to straightforwardly lift the features to 3D points and occasionally to apply the iterative closest point (ICP) algorithm for refinement of the pose estimate. Preference for systems using sparse features for localisation might be caused by the fact

that in monocular systems, direct odometry estimation from raw frames, *i.e.*, without extracting sparse features, inherently forced to estimate the dense depth or optical flow map between frames simultaneously or prior to the camera motion. These are ill posed problems with more unknowns than constraints and requiring from the use of regularisers and variational methods for their resolution [Newcombe et al., 2011b]. In addition to this, dense methods require high computational power for real time computation of pixel-wise operations. In this sense, advent of new generation CPUs and high performance GPUs almost simultaneously to RGB-D sensors allowed for a significant cost reduction of dense algorithms due to new programming paradigms which allowed for high paralellisation of per pixel operations.

One of the first and maybe most known approaches for direct odometry estimation is KinectFusion [Newcombe et al., 2011a], which using only the depth channel, is able to estimate the odometry and a dense map by ICP alignment. Almost alongside with KinectFusion came more direct approaches for odometry estimation either minimising the pixel-wise photometric error [Steinbrücker et al., 2011] or both photometric and geometric errors [Tykkala et al., 2011] between pairs of close enough frames.

We present a direct visual odometry method minimising both types of error. Our main contribution is the novelty of using the inverse depth to parametrise the geometric error instead of the depth as most works in the literature do. In monocular vision, the ability of inverse depth (if measured along the camera optical axis) or inverse distance (if measured along the projected ray) to easily deal with points at long distances [Civera et al., 2008] has been shown to lead to an improvement in performance [Solà et al., 2012]. In depth range systems, though there is no need to deal with points at distances greater than the maximum camera range, inverse depth has still the theoretical benefit of fitting better to the depth error model of a RGB-D camera. This potential benefit of using the inverse depth is experimentally validated in the Technische Universität München (TUM) real world benchmarking datasets [Sturm et al., 2012] as well as in synthetic datasets [Handa et al., 2014], showing the better performance of the geometric error based on inverse depth. As additional contribution, though equivalent in essence, the problem formulation is slightly varied with respect to related works, first linearising the flow equations, obtaining generic linear 3D flow equations and then applying the assumption of small rigid scene motion between frames to get the linear constraints just on camera motion parameters. We also evaluate the performance under different robust cost functions (Huber, Tukey biweight and Student's t distribution-based estimator) and two different methods in the state-of-the-art to compute the uncertainty-based scaling parameters of an error distribution: one with the Median Absolute Deviation (MAD), and one with its Maximum Likelihood (ML) estimator given the cost function used for robust optimisation. Also, as an alternative to alignment of consecutive frames, we consider the alignment against a reference frame, using a frame switching strategy based on covisibility between frames. We have implemented this method within the scope of PCL (Point Cloud Library) [Rusu and Cousins, 2011] as a branch of the code for large scale KinectFusion [Bondarev et al., 2013], where the original ICP system for odometry estimation has been completely substituted by our method, while algorithms for dense volumetric mapping and volume shifting have been kept unchanged. A fork of PCL including our modification is available for download [1].

4.2. Related Work

We classify the related work in RGB-D visual odometry into two categories: methods which rely at some point on the extraction and matching of sparse RGB features and those

[1] `http://webdiis.unizar.es/~danielgg/code.html`

which are completely dense performing pixel-wise minimisation of photometric and/or geometric constraints from the intensity and depth maps.

4.2.1. Sparse feature-based methods

One of the first SLAM methods with RGB-D cameras was presented by Henry *et al.* [Henry et al., 2010]. They perform the visual odometry estimation between two frames in two steps. In the first step SIFT features are matched and lifted to 3D points using the available depth information. Then RANSAC-based 3D alignment between the features in both frames is applied to find the initial estimate of the relative rigid transformation. In the second step the motion estimate is refined by joint minimisation of the euclidean 3D distance between inlier correspondences from previous step and the point to plane distance from the ICP alignment between point clouds. Both error contributions have to be weighted by a parameter which is empirically estimated. This method is further developed in [Henry et al., 2012], where main changes in the visual odometry estimation process are the substitution of SIFT features by FAST features, and the substitution of the 3D euclidean error between features by the image reprojection error.

In [Endres et al., 2012], Endres *et al.* propose a RGB-D SLAM system, which similarly to [Henry et al., 2010] estimate the initial transformation by RANSAC-based 3D alignment of sparse features. However, in the refinement step they only minimise the point cloud alignment error from the ICP algorithm. The method is improved in [Endres et al., 2014], including an Environment Measurement Model to prune wrong motion estimates which passed undetected in the RANSAC and ICP steps.

Maybe one weakness of both methods is that they seem to rely on the detection of loop closures to provide accurate map and trajectories estimates. The precision of the methods for visual odometry, *i.e.*, computing the motion estimate only between temporally close frames, can not be assessed since quantitative evaluation is performed after the loop closure step. Also they tend to directly extend the algorithm from visual odometry estimation for loop closure, rather than making first an appearance-based selection of loop candidates. In [Henry et al., 2012] authors attempt to align each new created keyframe with all the previous keyframes by RANSAC, which would lead to prohibitive computational costs for large datasets. In [Endres et al., 2014] in turn authors attempt to close loops in a random sample from a set of keyframes and each frame-to-frame motion calculation between candidates for loop closure is parallelised, thus keeping bounded the computational cost and close to the frame rate of the camera (between 5 and 15 Hz). This has proven to be a successful approach in the RGB-D TUM datasets, usually comprising short datasets with frequent loop closures. However in the case for large datasets with few loop closures, it could be argued that as the pool of eligible keyframes steadily grew, the chances of randomly picking keyframes with successful loop closures would be increasingly reduced.

In [Dong et al., 2014], Dong *et al.* propose the combination of 3D RANSAC alignment and large scale KinectFusion for RGB-D dense mapping.

4.2.2. Direct methods

In the last years, approaches which estimate the camera motion directly from the images without a previous extraction and matching of sparse features have become more and more popular. The main characteristic of these methods is that they compute the motion estimate between frames from pixel-wise constraints instead of sparse feature correspondences. Since direct methods make use of the whole dense information contained in the image, they are likely

to offer better precision for camera tracking than methods based on sparse features. The main restriction of direct methods is that inter-frame motion must be small, producing disparities of a few pixels. Though this restriction can be relaxed up to some degree with coarse-to-fine approaches by image downsampling; the main consequence is that though direct methods perform successfully in video sequences, where the temporal proximity between frames generally guarantees small pixel disparities, for temporally unrelated pairs of frames, as occurs in loop closure, sparse feature methods would be more robust.

Maybe one of the first dense approaches with RGB-D cameras is KinectFusion by Newcombe *et al.* [Newcombe et al., 2011a]. KinectFusion is composed by two different modules, one for camera tracking and one for dense volumetric mapping. For each new frame first the motion is estimated by frame-to-model ICP alignment of depth maps, *i.e.*, current depth map is aligned with the depth map raycasted from a voxelized 3D model. Then, current depth map is integrated in the 3D model using a truncated signed distance function. The main limitation of KinectFusion is its limitation to small workspaces, which was nevertheless solved in latter works by using a cyclical buffer to shift the volume as the camera explores the environment [Bondarev et al., 2013], [Whelan et al., 2013a].

Bylow *et al.* [Bylow et al., 2013] proposed a method which, as KinectFusion, uses only the depth fusion, but rather than raycasting a depth map from the model for posterior ICP-alignment, the camera is tracked by directly minimising the signed distance function between the current warped depth map and the model surface defined in the voxelised volume. This results in a better accuracy and similar real-time computational performance compared to KinectFusion.

In contrast to direct approaches using only geometric information, Steinbrücke *et al.* [Steinbrücker et al., 2011] presented a method for visual odometry estimation based on the pixel-wise minimisation of the photometric error between consecutive frames, showing that if interframe motion is small enough their approach is more accurate and computationally efficient than ICP alignment. Audras *et al.* [Audras et al., 2011] propose a similar method, but instead of standard least squares they propose the Huber robust cost function in order to gain robustness to outliers, *e.g.*, moving objects or occlusions. An information selection scheme is used to prune pixels in homogeneous regions and gain computational performance. Motion is estimated by aligning the current and a reference frame, switching the reference frame when the Median Absolute Deviation (MAD) of the error between the aligned frames is above a given threshold.

In [Kerl et al., 2013b], Kerl *et al.* extend the method described [Steinbrücker et al., 2011] by modelling the photometric error by a Student's t-distribution. This leads to a cost function which shows to be robust to outliers and performs better than other widely used estimators like Huber's or Tukey's.

Klose *et al.* [Klose et al., 2013] optimise the Tukey-robustified photometric error by performing Efficient Second Order Minimisation (ESM) between a reference and current frame, using the accumulated camera motion, to decide when to switch the reference frame. Also, to gain robustness to ilumination changes they include parameters modelling the variation in contrast and brightness in the optimised variables.

Following the paradigm of working on 3D models [Newcombe et al., 2011a], Stuckler and Behnke propose in [Stuckler and Behnke, 2012] and [Stückler and Behnke, 2014] converting the RGB and depth images into multi-resolution surfel maps by using a voxel octree representation. Each surfel maintains a shape-texture descriptor, which guide data association between surfels in different maps during camera pose estimation. To alleviate the odometry drift they register the current frame with respect to the latest keyframe. A new keyframe is inserted when camera motion w.r.t. last keyframe is large enough. They also propose a loop closure technique where loop closure candidates are randomly sampled from a probability density function which

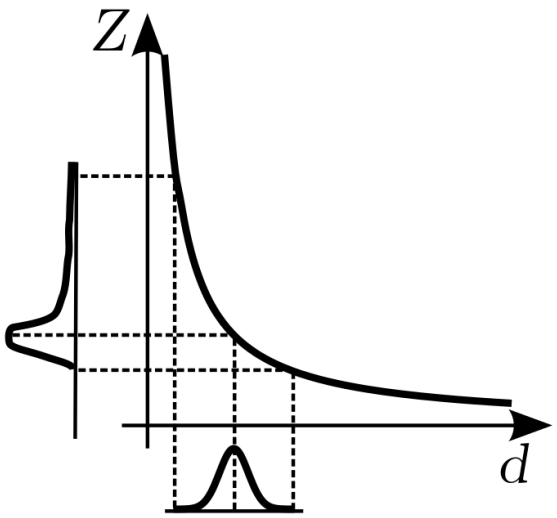


Figure 4.1: Assuming that the disparity (d) error follows a Gaussian or, more generally, a symmetric distribution, the depth ($Z \propto \frac{1}{d}$) error distribution is not Gaussian, not even symmetric. The asymmetry is more pronounced for higher Z.

positively weights the selection of spatially closer keyframes.

Direct motion estimation minimising simultaneously the geometric and photometric residuals was proposed first by Tykkala *et al.* [Tykkala et al., 2011]. To solve the problematic of mixing residuals in different magnitudes they propose the heuristic of weighting the depth residuals by the quotient of the medians of the intensity and depth maps. In [Damen et al., 2012], Damen *et al.* propose an ESM approach to minimise both residuals, weighting their contributions by an empirically set parameter. In [Whelan et al., 2013b], Whelan *et al.* propose to compute the visual odometry by mixing the costs functions from [Newcombe et al., 2011a] and [Steinbrücker et al., 2011], also weighting them by an empirically set parameter. Based on this work Whelan et al. proposed in [Whelan et al., 2014] a RGB-D SLAM system with volumetric fusion performing appearance-based loop closing to improve the system accuracy. The use of heuristics to weight both error contributions can be risky, since a tuning which works well for some dataset could not do so in other ones. For this reason it is advisable to reduce their use as much as possible. In this sense Kerl *et al.* [Kerl et al., 2013a] propose the computation of an automatic scaling matrix based on the covariance of the photometric and geometric pixel residuals, which produces a rigorous normalisation of both residuals. In addition to this automatic scaling they propose the estimation of the camera motion with respect to keyframes, which are switched following an entropy-based criteria, and the inclusion of a simple but effective loop closure method based on keyframes spatial proximity to further refine the final odometry estimation.

In all of the described approaches the use of a constant scaling parameter, either heuristic or automatic, for all the geometric residuals is prone to be a source of inaccuracies in the estimation process due to the quadratic grow of the depth uncertainty in RGB-D sensors [Khoshelham and Elberink, 2012]. Meilland *et al.* [Meilland and Comport, 2013b] take this fact into account, weighting the depth residuals by the inverse squared depth, but still use additional heuristic parameters to weight photometric and depth residuals. Also, under the frequent assumption of a symmetric, generally Gaussian, distribution of the disparity error; the depth which is inversely proportional to the image disparity, is not symmetric (Fig. 4.1), and thus inaccurately modeled by the robust cost functions frequently employed in the literature.

Inverse depth, in turn, depends linearly on the disparity, which means that it follows the same error distribution and thus its uncertainty is constant for any depth. In spite of this, to the best of our knowledge, the use of inverse depth for dense visual odometry with RGB-D cameras has been only proposed in by Lui *et al.* [Lui et al., 2012], by extending the ICP algorithm from KinectFusion. However its performance is only tested on short sequences, lacking from a thorough evaluation on larger RGB-D sequences and a comparison against state-of-the-art methods for dense RGB-D odometry estimation.

4.3. Linear Visual Odometry Constraints from Optical Flow

In this section we derive the visual odometry pixel-wise constraints through the flow equations obtained from the photometric and geometric constraints between two camera positions.

4.3.1. Optical flow equations

Let us denote two camera frames as A and B, at instants t and $t + \Delta t$ respectively. Given the intensity images $\mathcal{I}_A$ and $\mathcal{I}_B$, and inverse depth maps $\mathcal{W}_A$ and $\mathcal{W}_B$ defined over the image domain $\Omega \subset \mathbb{P}^2$, for an image point $\mathbf{p} = (u\ v\ 1)^T \in \Omega$ in frame A, the following constraints hold:

$$\mathcal{I}_B(\mathbf{p} + \boldsymbol{\Delta}\mathbf{p}) = \mathcal{I}_A(\mathbf{p}) \tag{4.1}$$

$$\mathcal{W}_B(\mathbf{p} + \boldsymbol{\Delta}\mathbf{p}) = \frac{1}{\mathbf{e}_\mathbf{z}^T \mathbf{X}_B}, \tag{4.2}$$

where $\mathbf{X}_B$ is the 3D point lifted from pixel $\mathbf{p} + \boldsymbol{\Delta}\mathbf{p}$ in frame B, $\boldsymbol{\Delta}\mathbf{p} = (\Delta u\ \Delta v\ 0)^T$ is the displacement of one point from frame A to B, and $\mathbf{e}_\mathbf{z}^T = (0\ \ 0\ \ 1)$. The constraint in intensity assumes constant illumination of one scene point. The second constraint is the measurement model of the depth sensor at frame B.

Assuming small pixel displacements between frames we compute the flow equations from (4.1) and (4.2):

$$\nabla\mathcal{I}_A(\mathbf{p})\boldsymbol{\Delta}\mathbf{p} + \mathcal{I}_B(\mathbf{p}) = \mathcal{I}_A(\mathbf{p}) \tag{4.3}$$

$$\nabla\mathcal{W}_A(\mathbf{p})\boldsymbol{\Delta}\mathbf{p} + \mathcal{W}_B(\mathbf{p}) = \frac{1}{\mathbf{e}_\mathbf{z}^T \mathbf{X}_B}, \tag{4.4}$$

where the gradient operators $\nabla\mathcal{I} = \left(\frac{\partial\mathcal{I}}{\partial u}\ \ \frac{\partial\mathcal{I}}{\partial v}\ \ 0\right)$ and $\nabla\mathcal{W} = \left(\frac{\partial\mathcal{W}}{\partial u}\ \ \frac{\partial\mathcal{W}}{\partial v}\ \ 0\right)$.

4.3.2. Projection model

A world point $\mathbf{X}$ is projected in the image point $\mathbf{p}$ by:

$$\mathbf{p} = \pi\left(\mathbf{X}\right) = \mathbf{K}\frac{\mathbf{X}}{\mathbf{e}_\mathbf{z}^T\mathbf{X}} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \frac{\mathbf{X}}{\mathbf{e}_\mathbf{z}^T\mathbf{X}}\ , \tag{4.5}$$

where $\mathbf{K}$ is the conventional calibration matrix, including the camera intrinsic parameters.

Inverse depth measurements $\mathcal{W}(\mathbf{p}) = \frac{1}{\mathbf{e}_\mathbf{z}^T\mathbf{X}}$ allow to lift 2D points from the image to 3D coordinates by the inverse projection function:

$$\mathbf{X} = \pi^{-1}(\mathbf{p}) = \frac{1}{\mathcal{W}(\mathbf{p})}\mathbf{K}^{-1}\mathbf{p}. \tag{4.6}$$

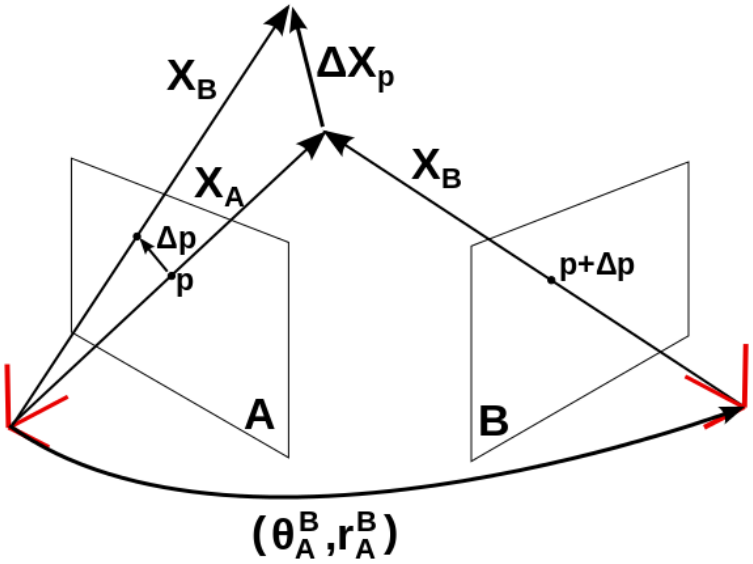


Figure 4.2: Schematic representation of optical and scene flow between two frames A and B.

4.3.3. 3D flow equations

Flow constraints (4.3) and (4.4), can be manipulated to get constraints on the 3D flow at one pixel, which is denoted as $\mathbf{\Delta X_p} \doteq \mathbf{X}_B - \mathbf{X}_A$ (see Fig. 4.2). Using this relation, first we compute the first order Taylor expansion of the inverse depth of one point at frame B:

$$\frac{1}{\mathbf{e_z}^T\mathbf{X}_B} = \frac{1}{\mathbf{e_z}^T\mathbf{X}_A} - \frac{1}{(\mathbf{e_z}^T\mathbf{X}_A)^2}\mathbf{e_z}^T\mathbf{\Delta X_p} + \mathcal{O}\left(\left|\left|\mathbf{e_z}^T\mathbf{\Delta X_p}\right|\right|^2\right) \tag{4.7}$$

$$\approx \mathcal{W}_A(\mathbf{p}) - \mathcal{W}_A^2(\mathbf{p})\mathbf{e_z}^T\mathbf{\Delta X_p}. \tag{4.8}$$

Using this relation and the camera projection model we get also:

$$\mathbf{\Delta p} = \mathbf{K}\frac{\mathbf{X}_B}{\mathbf{e_z}^T\mathbf{X}_B} - \mathbf{K}\frac{\mathbf{X}_A}{\mathbf{e_z}^T\mathbf{X}_A} \tag{4.9}$$

$$\overset{(4.8)}{=} \mathbf{K}\mathbf{X}_B\left(\mathcal{W}_A(\mathbf{p}) - \mathcal{W}_A^2(\mathbf{p})\mathbf{e_z}^T\mathbf{\Delta X_p}\right) - \mathbf{K}\mathbf{X}_A\mathcal{W}_A(\mathbf{p}) \tag{4.10}$$

$$\overset{(4.5)}{=} \mathcal{W}_A(\mathbf{p})\left(\mathbf{K} - \mathbf{p}\mathbf{e_z}^T\right)\mathbf{\Delta X_p} - \mathbf{K}\mathcal{W}_A^2(\mathbf{p})\mathbf{\Delta X_p}\mathbf{e_z}^T\mathbf{\Delta X_p} \tag{4.11}$$

$$\approx \mathcal{W}_A(\mathbf{p})\left(\mathbf{K} - \mathbf{p}\mathbf{e_z}^T\right)\mathbf{\Delta X_p}. \tag{4.12}$$

And substituting in (4.3) and (4.4) we get:

$$\mathcal{W}_A(\mathbf{p})\nabla\mathcal{I}_A(\mathbf{p})\left(\mathbf{K} - \mathbf{p}\mathbf{e_z}^T\right)\mathbf{\Delta X_p} + \mathcal{I}_B(\mathbf{p}) - \mathcal{I}_A(\mathbf{p}) = 0 \tag{4.13}$$

$$\mathcal{W}_A(\mathbf{p})\left(\nabla\mathcal{W}_A(\mathbf{p})\left(\mathbf{K} - \mathbf{p}\mathbf{e_z}^T\right) + \mathcal{W}_A(\mathbf{p})\mathbf{e_z}^T\right)\mathbf{\Delta X_p} + \\ + \mathcal{W}_B(\mathbf{p}) - \mathcal{W}_A(\mathbf{p}) = 0. \tag{4.14}$$

4.3.4. Rigid motion

We have obtained general flow equations taking small pixel displacement as the only assumption. Only (4.3) presents a dense optical or 2D flow estimation problem [Horn and

Schunck, 1981], while (4.13) and (4.14) involve a dense scene or 3D flow problem [Herbst et al., 2013]. Both are ill posed problems which require regularisation and variational methods to reach a solution.

We focus instead on RGB-D visual odometry estimation. This implies the assumption of a rigid scene, i.e., the displacements $\mathbf{\Delta X_p}$ of each of the $W_{im}H_{im}$ points projected on the image frame are due only to the motion of the camera, which has 6 DoF. Assuming a small motion described by the rotation and translation pair $({}_A\mathbf{R}^B, \mathbf{r}_B^A) \in \mathbb{SE}(3)$ we have:

$$\begin{aligned}\mathbf{\Delta X_p} &= {}_B\mathbf{R}^A\mathbf{X}_A + \mathbf{r}_B^A - \mathbf{X}_A \\ &= \left(\mathbf{I} + [\boldsymbol{\theta}_B^A]_\times\right)\mathbf{X}_A + \mathbf{r}_B^A - \mathbf{X}_A + \mathcal{O}\left(\left\|[\boldsymbol{\theta}_B^A]_\times^2\mathbf{X}_A\right\|\right) \\ &\approx \mathbf{r}_B^A - [\pi^{-1}(\mathbf{p})]_\times\boldsymbol{\theta}_B^A = \mathbf{M}(\mathbf{p})\boldsymbol{\xi}_B^A. \end{aligned} \tag{4.15}$$

where $[\cdot]_\times$ denotes the antisymmetric matrix from a vector and $\boldsymbol{\theta}_B^A$ is the logarithmic map of ${}_A\mathbf{R}^B$. Note that $\boldsymbol{\xi}_B^A = (\mathbf{r}_B^A;\ \boldsymbol{\theta}_B^A)$ is not a twist, *i.e.*, $\boldsymbol{\xi}_B^A \notin \mathfrak{se}(3)$, since $\mathbf{r}_B^A$ is yet the translation part of the rigid motion. Eq. (4.15) leads to a well-posed problem with 6 unknowns for nearly $W_{im}H_{im}$ constraints, excluding pixels without depth measurements, with the following residuals:

$$\begin{aligned} r_{\mathcal{I}}(\mathbf{p},\boldsymbol{\xi}) = \mathcal{W}_A(\mathbf{p})\nabla\mathcal{I}_A(\mathbf{p})(\mathbf{K}-\mathbf{p}\mathbf{e}_\mathbf{z}^T)\mathbf{M}(\mathbf{p})\boldsymbol{\xi}+ \\ + \mathcal{I}_B(\mathbf{p}) - \mathcal{I}_A(\mathbf{p}) \end{aligned} \tag{4.16}$$

$$\begin{aligned} r_{\mathcal{W}}(\mathbf{p},\boldsymbol{\xi}) = \mathcal{W}_A(\mathbf{p})\big(\nabla\mathcal{W}_A(\mathbf{p})(\mathbf{K}-\mathbf{p}\mathbf{e}_\mathbf{z}^T)+\mathcal{W}_A(\mathbf{p})\mathbf{e}_\mathbf{z}^T\big)\mathbf{M}(\mathbf{p})\boldsymbol{\xi}+ \\ + \mathcal{W}_B(\mathbf{p}) - \mathcal{W}_A(\mathbf{p}), \end{aligned} \tag{4.17}$$

which can be straightforwardly minimised by standard Gauss-Newton least squares.

Note that in the monocular RGB case, no depth is provided and only the constraint (4.16) would be used. Thus $\mathcal{W}_A(\mathbf{p})$ becomes an unknown yielding an ill-posed problem with $W_{im}H_{im}$ constraints for $W_{im}H_{im} + 6$ unknowns, which is solved using optical flow variational methods [Newcombe et al., 2011b], or by performing variable baseline stereo matching [Engel et al., 2013].

4.4. Visual Odometry Estimation by Iterative Optimisation

With the proposed residuals, $\boldsymbol{\xi}_B^A$ is computed as the solution to the following optimisation problem:

$$\boldsymbol{\xi}_B^A = \underset{\boldsymbol{\xi}}{\operatorname{argmin}} \sum_{\mathbf{p}\in\Omega} \rho\left(\frac{r_{\mathcal{I}}(\mathbf{p},\boldsymbol{\xi})}{\sigma_{r_{\mathcal{I}}}}\right) + \rho\left(\frac{r_{\mathcal{W}}(\mathbf{p},\boldsymbol{\xi})}{\sigma_{r_{\mathcal{W}}}}\right), \tag{4.18}$$

where $\rho(x)$ is a generic cost function which must be symmetric, definite positive and $\rho(0) = 0$. $\sigma_{r_{\mathcal{I}}}$ and $\sigma_{r_{\mathcal{W}}}$ are scaling parameters which capture the uncertainty in intensity and inverse depth residuals, and allow for normalisation of residuals in different magnitudes. The choice $\rho(x) = \frac{x^2}{2}$ results in standard least-squares linear optimisation. Nevertheless to gain robustness against outliers, *e.g.*, pixels belonging to non-static elements, robust M-estimators are usually employed. Optimisation with robust cost functions is addressed by the Iteratively Reweighted Least Squares algorithm (IRLS) [Holland and Welsch, 1977] (see Appendix B), which results in a linear least-squares problem to be solved at each iteration:

$$\boldsymbol{\xi}_B^A = \underset{\boldsymbol{\xi}}{\operatorname{argmin}} \sum_{\mathbf{p}\in\Omega} \omega\left(\frac{\check{r}_{\mathcal{I}}(\mathbf{p})}{\sigma_{r_{\mathcal{I}}}}\right)\frac{r_{\mathcal{I}}^2(\mathbf{p},\boldsymbol{\xi})}{\sigma_{r_{\mathcal{I}}}^2} + \omega\left(\frac{\check{r}_{\mathcal{W}}(\mathbf{p})}{\sigma_{r_{\mathcal{W}}}}\right)\frac{r_{\mathcal{W}}^2(\mathbf{p},\boldsymbol{\xi})}{\sigma_{r_{\mathcal{W}}}^2}, \tag{4.19}$$

where $\check{r}_{\mathcal{I}}(\mathbf{p})$ and $\check{r}_{\mathcal{W}}(\mathbf{p})$ denote the initial residuals computed after updating the camera motion at previous iteration, and the weighting function $\omega(x)$ depends on the used M-estimator. For more details on robust minimisation with M-estimators we refer the reader to Appendix B.

Rigid motion between frames is computed in a coarse-to-fine manner using image pyramids, performing a number of iterations at each pyramid level. Let us have the intensity and inverse depth image pairs $\{\mathcal{I}_k,\ \mathcal{W}_k\}$ and $\{\mathcal{I}_{k+1},\ \mathcal{W}_{k+1}\}$, between consecutive frames k and $k+1$. At the start and every time we step down to the next pyramid level, we set $\{\mathcal{I}_A,\ \mathcal{W}_A\} = \left\{\mathcal{I}_k^{(pyr)},\ \mathcal{W}_k^{(pyr)}\right\}$, and compute $\{\nabla\mathcal{I}_A,\ \nabla\mathcal{W}_A\}$. Initial camera motion, expressed by the transform ${}_k\mathbf{T}_{(0)}^{k+1}$ is initialised assuming a constant velocity, *i.e.*, ${}_k\mathbf{T}_{(0)}^{k+1} = {}_{k-1}\mathbf{T}^k$.

After initialisation, the following steps are performed at each iteration γ: image warping, scaling parameters computation, optimisation and pose composition.

4.4.1. Image Warping

Image warping is performed at the start of every iteration in order to reset the incremental motion estimate to $\boldsymbol{\xi}_B^{A(\gamma)} = 0$, instead of accumulating it. This is done to avoid unrealistic intensity or inverse depth estimates in frame B beyond the sensor measurement limits, as it can be verified if we take a look back to the left members of (4.3) and (4.4). At each iteration, $\{\mathcal{I}_{k+1},\ \mathcal{W}_{k+1}\}$ are warped towards frame k using the current motion estimate ${}_k\mathbf{T}_{(\gamma)}^{k+1}$, resulting in the warped images $\{\mathcal{I}_B^{(\gamma)},\ \mathcal{W}_B^{(\gamma)}\}$. This is done by reverse warping in the following steps:

- Given a pixel $\mathbf{p}$ in the destination warped image, the corresponding pixel $\mathbf{p}_{k+1}^{(\gamma)}$ in the source image is obtained as:

$$\mathbf{X}_{k+1}^{(\gamma)} = {}_{k+1}\mathbf{R}_{(\gamma)}^k\mathbf{K}^{-1}\mathbf{p}\frac{1}{\mathcal{W}_k(\mathbf{p})} + \mathbf{r}_{k+1}^{k(\gamma)}, \tag{4.20}$$

$$\mathbf{p}_{k+1}^{(\gamma)} = \mathbf{p} + \boldsymbol{\Delta}\mathbf{p}^{(\gamma)} = \mathbf{K}\frac{\mathbf{X}_{k+1}^{(\gamma)}}{\mathbf{e}_{\mathbf{z}}^T\mathbf{X}_{k+1}^{(\gamma)}}, \tag{4.21}$$

- By using (4.1) and (4.2) and resetting $\mathbf{X}_{k+1}^{(\gamma)} = {}_{k+1}\mathbf{R}_{(\gamma)}^k\mathbf{K}^{-1}\mathbf{p}\frac{1}{\mathcal{W}_B^{(\gamma)}(\mathbf{p})} + \mathbf{r}_{k+1}^{k(\gamma)}$, compute the warped intensity and inverse depth maps, $\mathcal{I}_B^{(\gamma)}$ and $\mathcal{W}_B^{(\gamma)}$ as:

$$\mathcal{I}_B^{(\gamma)}(\mathbf{p}) = \mathcal{I}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)}) \tag{4.22}$$

$$\mathcal{W}_B^{(\gamma)}(\mathbf{p}) = \frac{\mathbf{e}_{\mathbf{z}}^T\,{}_{k+1}\mathbf{R}_{(\gamma)}^k\mathbf{K}^{-1}\mathbf{p}}{1 - \frac{\mathbf{e}_{\mathbf{z}}^T\mathbf{r}_{k+1}^{k(\gamma)}}{\mathcal{W}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)})}}\mathcal{W}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)}), \tag{4.23}$$

  where $\mathcal{I}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)})$ and $\mathcal{W}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)})$ are obtained by bilinear interpolation, which is efficiently computed with CUDA capable NVIDIA GPUs using *texture memory*.

  Warping is performed at the level with highest resolution. Once the warping is done, $\{\mathcal{I}_B^{(\gamma)},\ \mathcal{W}_B^{(\gamma)}\}$ are downsampled to the pyramid level where current optimisation step is taking place.

4.4.2. Scaling parameters

In a proper minimisation problem, specially when mixing residuals in different magnitudes, the scaling parameters related with the covariance of the residuals need to be provided. In some cases these scaling parameters are known or can be estimated before the optimisation and thus they can be introduced as constants. However in other cases they are difficult to know and they have to be computed prior to every optimisation step from the current estimates of the residuals. In principle we assume that these parameters are not known and to obtain them we first compute the initial residuals at $\boldsymbol{\xi}_B^{A(\gamma)} = 0$:

$$\breve{r}_{\{\mathcal{I},\mathcal{W}\}}(\mathbf{p}) = \{\mathcal{I}_B^{(\gamma)}(\mathbf{p}),\ \mathcal{W}_B^{(\gamma)}(\mathbf{p})\} - \{\mathcal{I}_A(\mathbf{p}),\ \mathcal{W}_A(\mathbf{p})\}. \tag{4.24}$$

Scaling parameters $\sigma_{r_\mathcal{I}}$ and $\sigma_{r_\mathcal{W}}$ can then be computed by the Median Absolute Deviation (MAD):

$$\sigma^{MAD}_{r_{\{\mathcal{I},\mathcal{W}\}}} = 1.4286 \operatorname*{med}_{\mathbf{p}} |\breve{r}_{\{\mathcal{I},\mathcal{W}\}}(\mathbf{p}) - \operatorname*{med}_{\mathbf{p}} \left(\breve{r}_{\{\mathcal{I},\mathcal{W}\}}(\mathbf{p})\right)|, \tag{4.25}$$

or alternatively they can be computed by their Maximum Likelihood (ML) estimator, noticing that for any cost function $\rho(\frac{r-\mu}{\sigma})$ we can obtain the associated likelihood function as:

$$f_\rho(r|\mu,\sigma) = \frac{K_\rho}{\sigma} \exp\left(-\rho\left(\frac{r-\mu}{\sigma}\right)\right), \tag{4.26}$$

where K_ρ is a scaling constant for $\int_{-\infty}^{\infty} f_\rho(r|\mu,\sigma)dr = 1$, and μ and σ the location and the scaling parameter of the residuals respectively. The scaling parameters would be computed by iteratively solving:

$$\left[\mu^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}, \sigma^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}\right] = \operatorname*{argmin}_{\mu,\ \sigma} \sum_{\mathbf{p}\in\Omega} \log\sigma + \rho\left(\frac{\breve{r}_{\{\mathcal{I},\mathcal{W}\}}(\mathbf{p}) - \mu}{\sigma}\right), \tag{4.27}$$

taking $\rho(x) = \frac{x^2}{2}$ in the first iteration to compute the initial seed and then switching to our selected cost function. Though not explicitly required, location parameters $\mu^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}$ are also calculated since the scaling parameters depend on their estimate.

Computing the scaling parameters using all the pixels in the image can involve a high computational burden. As an alternative, we propose computing the scale parameter taking only a sample from all the pixel residuals. Following a similar reasoning to the typically followed to compute the minimum number of iterations in a RANSAC scheme, we can determine statistically the minimum sample size N to reach a relative precision ϵ in the scale parameter estimation with a confidence $1-\alpha$. Assuming that the sum of the weighted squared normalised residuals follows a chi squared distribution, we have that:

$$\frac{(N-1)\hat{\sigma}^2}{\sigma^2} \sim \chi^2_{N-1}, \tag{4.28}$$

where σ is the true scaling parameter and $\hat{\sigma}$ its estimate. We can define then a confidence interval with the desired relative precision in our estimate.

$$\begin{aligned} &P\left((1-\epsilon)\sigma^2 < \hat{\sigma}^2 < (1+\epsilon)\sigma^2\right) = \\ &= P\left((1-\epsilon)(N-1) < \frac{(N-1)\hat{\sigma}^2}{\sigma^2} < (1+\epsilon)(N-1)\right) = 1-\alpha \end{aligned} \tag{4.29}$$

Since $N > 50$, we can approximate the chi squared distribution by a Gaussian distribution, $\chi^2_{N-1} \sim \mathcal{N}(N-1, 2(N-1))$, and then we have:

$$P\left(-\frac{\epsilon(N-1)}{\sqrt{2(N-1)}} < z < \frac{\epsilon(N-1)}{\sqrt{2(N-1)}}\right) = 1-\alpha, \tag{4.30}$$

where z is a standard normally distributed random variable. From the previous expression we obtain the minimum required number of samples to obtain a desired precision in the estimated scaling parameter with a given confidence level.

$$N = \frac{2z^2_{1-\frac{\alpha}{2}}}{\epsilon^2} + 1. \tag{4.31}$$

Taking a sample of $N = 10000$ pixels we obtain a relative precision $\epsilon = 0.05$ for the scaling parameter with a confidence level greater than 99.9%.

4.4.3. Robust optimisation

We consider for comparison 3 different cost functions frequently used in the related literature: Huber [Huber, 1964a], Tukey biweight [Beaton and Tukey, 1974] and a Student's t-distribution-based estimator [Lange et al., 1989], which will be denoted as Student in advance. The constants for Huber and Tukey estimators are set up to 1.345 and 4.685 respectively, which yield an asymptotic relative efficiency (ARE) of 95% in case the error followed a Gaussian distribution. The number of degrees of freedom of the Student estimator is set up to $\nu = 5$ as in [Kerl et al., 2013b]. We have verified numerically that the ARE of the Student's estimator with this setup is near to 94% for a normally distributed error.

Given an M-estimator, once we have the initial residuals and the scaling parameters the computation of the weights for IRLS is straightforward. This is followed by the computation of $\boldsymbol{\xi}_B^{A(\gamma)}$ by solving the linear optimisation in (4.19).

4.4.4. Motion update

After each iteration the motion estimate between frames k and $k+1$ is updated by the current incremental estimate:

$${}_k\mathbf{T}^{k+1}_{(\gamma+1)} = \begin{pmatrix} \exp([\boldsymbol{\theta}_B^{A(\gamma)}]_\times) & \mathbf{r}_B^A \\ 0 & 1 \end{pmatrix}^{-1} {}_k\mathbf{T}^{k+1}_{(\gamma)}. \tag{4.32}$$

4.4.5. Keyframe selection by covisibility ratios

Taking a reference frame for estimation of the visual odometry requires a switching strategy. Very often the decision of switching the reference frame, or keyframe, is taken based on rotational and translational distance between frames. However we find that this might be a poor criteria since first, how the captured environment changes when the camera translates depends also on the depth of the elements in the scene, and secondly, camera motion does not necessarily result in a variation of the captured scene (consider for example the case of simultaneously translating the camera along its horizontal axis and rotating it around its vertical axis).

Recent works use instead a statistic criteria for reference frame switching. In [Meilland et al., 2011] authors use the MAD estimator of the standard deviation of the final pixel-wise

residuals between reference and current frame, and compare it to a reference value in order to make the decision. In [Kerl et al., 2013a] instead of the variance of the residuals, authors use the covariance matrix of the computed motion. They take as reference value the covariance of the motion estimate between the last inserted keyframe and its consecutive frame and take a keyframe switching decision based on the ratio between the entropy of the reference covariance and the current estimate's.

Using sparse features a frequently used criteria for keyframe insertion is the visibility of map features in current frame [Klein and Murray, 2007], [Huang et al., 2011]. This criteria has also been used for dense point clouds [Meilland and Comport, 2013a].

In this work we use a covisibility criteria described as follows. Given two frames A and B and the camera motion estimate between them $({}_A\mathbf{R}^B, \mathbf{r}_B^A) \in \mathbb{SE}(3)$, we transfer pixels from A to B using (4.20) and (4.21). Then if the following two conditions are met:

- $\mathbf{p}_B \in \Omega$
- $\left| \mathcal{W}_B(\mathbf{p}_B) - \frac{1}{\mathbf{e_z}^T \mathbf{X}_B} \right| < 3\sigma_w$

a pixel is tagged as visible. First condition rejects points out of the image domain, while the second condition rejects occluded pixels and where σ_w is computed using (4.27) after the last iteration of the visual odometry algorithm. After this test we can compute the visibility ratio:

$$vis\_ratio_{A\to B} = \frac{\#visible\_pixels_{A\to B}}{\#nohole\_pixels_{A\to B}}. \tag{4.33}$$

This procedure is repeated switching the role of A and B, and then we select the minimum visibility ratio. If this ratio is below a threshold, the reference frame is switched. Different thresholds will be tested in the experiments section.

4.4.6. Enhancement of the computational performance

Optimisation is performed in a coarse-to-fine scheme at 3 pyramid levels (160x120, 320x240 and 640x480). The naive approach offers the highest precision performing a fixed number of 10 iterations at each level. This results in cost per frame of about 50 ms, which is broke down into the costs of the different processes in Fig. 4.3. Alternatively, to improve the time performance we consider the following optimisations:

- Skip optimisation on the highest resolution level.
- In Fig. 4.3 it can be observed that one important fraction of the time is employed in estimating the scale parameters $\sigma_\mathcal{I}$ and $\sigma_\mathcal{W}$. This cost can be completely eliminated if we fix the scaling parameters for the optimisation. We propose taking $\sigma_\mathcal{I} = 5$, with $\mathcal{I}(\mathbf{p}) \in (0,\ 255)$ and $\sigma_\mathcal{W} = 0.0025\ \mathrm{m}^{-1}$. The choice for $\sigma_\mathcal{I}$ is justified by tests with static sequences while $\sigma_\mathcal{W}$ stems from the precision of the disparity measurements [Konolige and Mihelich, 2014], [Martinez and Stiefelhagen, 2013].
- Instead of warping at the highest resolution and then downsampling at each iteration, a coarser but more efficient alternative would be downsampling $\{\mathcal{I}_{k+1},\ \mathcal{W}_{k+1}\}$ before optimising and warping on the current pyramid level.

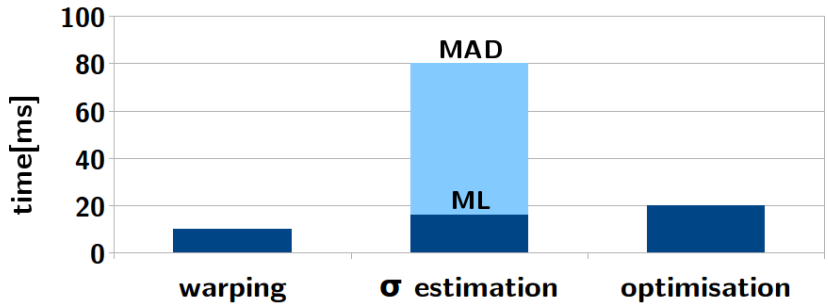


Figure 4.3: Costs of the processes involved in the computation of the RGB-D visual odometry.

4.5. Experiments

In the first experiments we use the RGB-D dataset from Technische Universität München (TUM) [Sturm et al., 2012] to evaluate the accuracy of our approach with different configurations, paying special attention to the comparison of performances using inverse depth or depth based geometric residuals. We also study with the same datasets, how both the accuracy and the computation speed are affected when applying the options for enhancement of computational performance from section 4.4.6.

Then after selecting the configuration which yields the better accuracy we compare our method, using different covisibility thresholds for keyframe switching, to other works in the literature. For this comparison we use not only the TUM datasets, but also the synthetic datasets presented in [Handa et al., 2014].

Finally we show a qualitative evaluation of our visual odometry method showing the 3D reconstructions obtained from some of the TUM datasets and also from our own sequences.

The experiments were performed on a desktop computer with Ubuntu 12.04 32-bits and equipped with an Intel Core i5-2500 CPU at 3.30 GHz, 8GB and a nVidia GeFroce 660GTX GPU with 2GB of memory. The implementation was done as an extension of the large scale KinectFusion large scale algorithm from the Point Cloud Library (PCL) [Rusu and Cousins, 2011], where the original ICP system for odometry estimation has been completely substituted by our method. The algorithms for dense volumetric mapping and volume shifting are kept unchanged. A fork of PCL including our modified version is available for download in `http://webdiis.unizar.es/~danielgg/code.html`.

4.5.1. Inverse depth vs depth residuals

We evaluated all the possible combinations of robust cost functions (Huber, Tukey and Student), geometric error parametrisation (depth, inverse depth) and residual scale estimators (MAD, ML). Table 4.1 shows the different values of the RMSE for the translational drift, measured in m/s, using different approaches. Best results are obtained with the Student estimator, with little difference between using MAD or ML estimators for the scaling parameters. Parametrisation of the geometric error with inverse depth yields an improvement over the depth parametrisation in all the datasets except in $fr2/desk$. It can be noted also that the Huber estimator offers in general the lowest accuracy and that the performance of the Tukey estimator is slightly worse if we use the ML estimator for σ. However its performance is comparable to Student's if the MAD estimator is used. For the rest of the experiments we use the configuration with Student robust cost function, inverse depth parameterisation for the geometric error and the ML estimator for the scale parameters.

Table 4.1: Translational drift relative root mean square error (RMSE) in meters per second using different methods for RGB-D visual odometry estimation

Estimator	Geom. error	σ	fr1/desk	fr1/desk2	fr1/room	fr2/desk
Student	depth	ML	0.0278	0.0425	0.0504	**0.0115**
Student	depth	MAD	0.0271	0.0439	0.0490	0.0121
Student	invDepth	ML	**0.0260**	**0.0387**	0.0491	0.0121
Student	invDepth	MAD	**0.0260**	0.0396	0.0485	0.0122
Tukey	depth	ML	0.0292	0.0808	0.0502	0.0143
Tukey	depth	MAD	0.0271	0.0527	0.0483	0.0142
Tukey	invDepth	ML	0.0381	0.0720	0.0485	0.0135
Tukey	invDepth	MAD	0.0287	0.0422	**0.0471**	0.0131
Huber	depth	ML	0.0322	0.0495	0.0681	0.0193
Huber	depth	MAD	0.0322	0.0496	0.0662	0.0233
Huber	invDepth	ML	0.0289	0.0453	0.0640	0.0209
Huber	invDepth	MAD	0.0280	0.0435	0.0606	0.0219

Table 4.2: Translational drift relative root mean square error (RMSE) in meters per second minimising different types and combinations of errors

Geom. error	Min. errors	fr1/desk	fr1/desk2	fr1/room	fr2/desk
none	phot	0.0312	0.0513	0.0503	0.0119
invDepth	phot+geom	0.0260	0.0387	0.0491	0.0121
depth	phot+geom	0.0278	0.0425	0.0504	0.0115
invDepth	geom	0.0332	0.0454	0.0495	0.0356
depth	geom	0.0377	0.0562	0.0555	0.0465

For a more detailed evaluation of how the combination of both geometric and photometric errors affect the accuracy of the estimate and better assess the gain of using inverse depth instead of depth for the geometric error, we have performed another set of tests with optimising on intensity error only, geometric only and the combination of both geometric and intensity error. For the cases where geometric error is used we have switched between inverse depth and depth based errors (Table 4.2). It can be observed that when considering the geometric error only the superiority of using inverse depth residuals is clearer.

4.5.2. Performance vs accuracy

We evaluated how the accuracy and computational performance are affected after applying the modifications proposed in Sec. 4.4.6 to decrease the computational cost of our approach. Following the order in which they are presented, we denote these as *lvl* followed by the level index at which optimisation is stopped, $sigmaFix$ and $pyrFirst$. Results of using one or a combination of these modifications are shown in Table 4.3. It can be observed that either stopping optimisation at level 1, *i.e.*, resolution of 320x240, or using constant scale we achieve a computation time in the limits of the camera frame rate of 30 Hz with little lost in accuracy, and even we can reach a reduction to 9ms applying all the proposed optimisations.

Table 4.3: Translational drift and average and maximum computation time per frame for different options to enhance the computational performance

Approach	RMSE[m/s]				time	
	fr1/desk	fr1/desk2	fr1/room	fr2/desk	mean[ms]	max[ms]
naïve	0.0260	0.0387	0.0491	0.0121	47	50
lvl1	0.0268	0.0407	0.0492	0.0120	26	28
pyrFirst	0.0270	0.0401	0.0491	0.0127	40	42
pyrFirst+lvl1	0.0282	0.0416	0.0498	0.0138	18	20
σFix	0.0260	0.0389	0.0498	0.0112	34	35
σFix+lvl1	0.0271	0.0399	0.0500	0.0121	17	17
σFix+pyrFirst	0.0272	0.0397	0.0498	0.0118	26	27
σFix+pyrFirst+lvl1	0.0287	0.0409	0.0500	0.0143	9	10

4.5.3. State-of-the-art comparative

In this section we first compare our method to state-of-the-art RGB-D visual odometry and SLAM approaches. For the comparison we use 2 publicly available benchmarking datasets: the TUM benchmarking dataset [Sturm et al., 2012], consisting of real image sequences, and a RGB-D benchmarking dataset generated from 2 synthetic scenes, one office and one living room. For our approach we have considered different thresholds for reference frame switching. The evaluation in the TUM datasets has been carried out in the two error metrics proposed by the authors of the TUM datasets: the Relative Pose Error (RPE) in meters per second and the Absolute Trajectory Error (ATE) in meters. For the synthetic datasets we only evaluated the ATE since RPE is not compared in the literature.

In the RPE evaluation (Table 4.4), we observe that there does not exist a clear difference in accuracy between taking consecutive frames or a reference frame for odometry estimation. With respect to the state-of-the-art, our frame-to-frame approach has the lowest error in the $fr1/desk2$ and $fr2/desk$ datasets. In the $fr1/desk$ dataset our approach is not the best but the results are close to [Meilland and Comport, 2013a] and RGB-D+KF+Opt [Kerl et al., 2013a], even ours not using reference frames. In $fr1/room$, the better accuracy of [Meilland and Comport, 2013a] is clear.

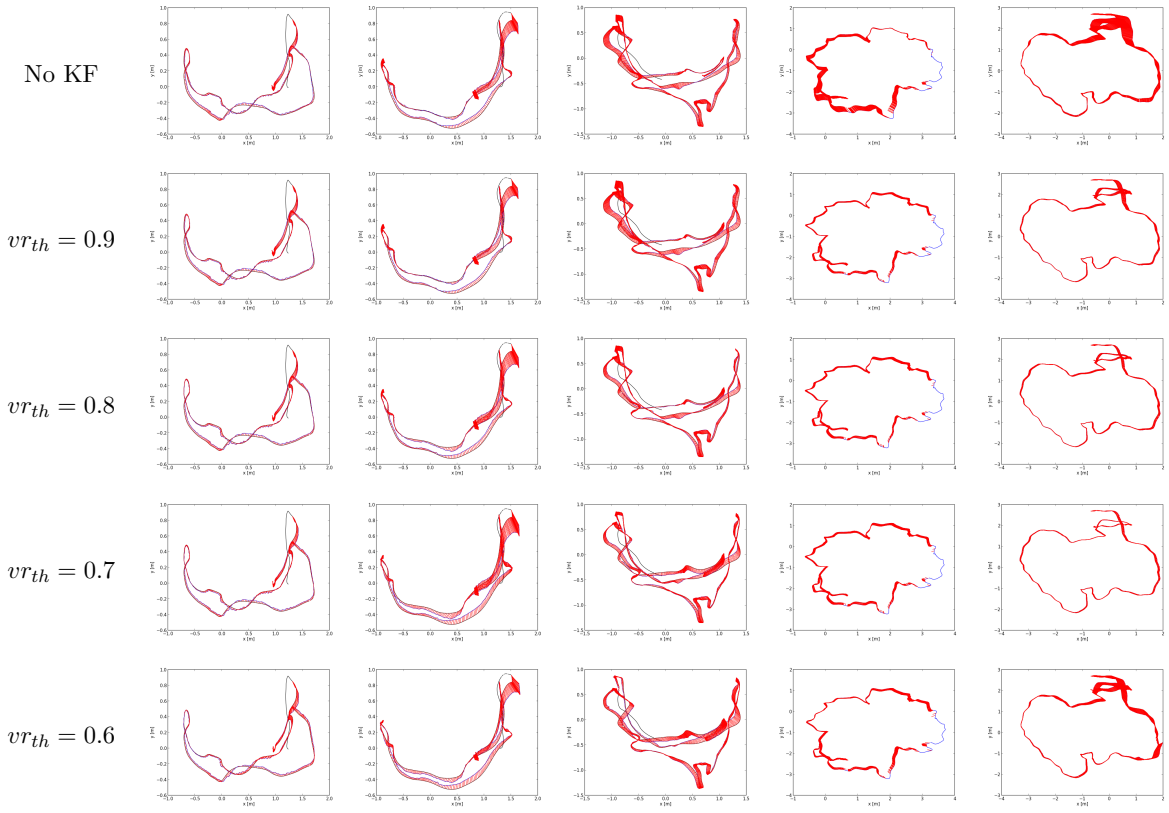
The ATE is qualitatively shown in Fig. 4.4 and quantitatively evaluated in Table 4.5. We found that though our approach is not the best in the tested real TUM datasets, it shows competitive results considering that many of the methods considered in the comparison include some kind of loop closure method, which significantly helps into reducing this error. As in the RPE comparative it can be observed that taking a reference frame has a moderate or almost unnoticeable effect on the accuracy except for the $fr2/desk$ and $fr3/office$ datasets, where the ATE metrics are twice larger when not taking reference frames. Given that these datasets were acquired with a slow moving camera, this greater error is likely to be caused by a motion between consecutive frames producing in some parts an optical flow beyond the limits of the pixel accuracy. Fixing the visibility ratio threshold to a high value (0.8 or 0.9) seems to be enough to prevent this problem from happening in sequences with slow camera motion and has no negative effects on sequences with faster motion.

In the synthetic datasets the comparison was performed against the RGB-D odometry methods originally evaluated by the authors of the benchmark. In Table 4.6 we show the ATE for different values of the visibility ratio threshold which have been compared against the lowest ATE of the approaches evaluated in [Handa et al., 2014], which generally corresponds to

Table 4.4: Translational drift relative root mean square error (RMSE) in meters per second using different visibility ratio thresholds for keyframe switching and comparison with state-of-the-art approaches

Visibility ratio	fr1/desk	fr1/desk2	fr1/room	fr2/desk	fr3/office
No KF	0.0260	0.0387	0.0491	0.0121	0.0168
$vr_{th} = 0.9$	0.0255	0.0384	0.0473	**0.0115**	**0.0118**
$vr_{th} = 0.8$	0.0253	**0.0382**	0.0472	0.0124	0.0120
$vr_{th} = 0.7$	**0.0246**	0.0385	0.0441	0.0131	0.0121
$vr_{th} = 0.6$	0.0264	0.0413	**0.0423**	0.0258	0.0812
FOVIS ([Huang et al., 2011])$*$	0.0604	-	0.0642	**0.0136**	-
ICP+RGB-D [Whelan et al., 2013a]	0.0393	-	0.0622	0.0208	-
VP [Meilland and Comport, 2013a]	0.0259	-	**0.0351**	0.0147	-
ESM + Tukey + Aff. Il. [Klose et al., 2013]	0.0302	0.0526	0.0397	0.0147	-
RGB+D [Kerl et al., 2013a]	0.036	**0.049**	0.058	-	-
RGB+D+KF [Kerl et al., 2013a]	0.030	0.055	0.048	-	-
RGB+D+KF+Opt [Kerl et al., 2013a]	**0.024**	0.050	0.043	-	-

$*$ comp. in [Whelan et al., 2013a]



(a) fr1/desk (b) fr1/desk2 (c) fr1/room (d) fr2/desk (e) fr3/office

Figure 4.4: Trajectories on real TUM datasets. Estimated trajectory is shown in blue, ground truth is in black. Error between visual estimate and ground truth is shown in red.

Table 4.5: Absolute trajectory error (RMSE, median and max) in meters using different visibility ratio thresholds for keyframe switching and comparison with state-of-the-art approaches

Visibility ratio	fr1/desk			fr1/desk2			fr1/room			fr2/desk			fr3/office		
	RMSE	median	max	RMSE	median	max	RMSE	median	max	RMSE	median	max	RMSE	median	max
No KF	**0.032**	0.027	**0.078**	0.070	0.047	0.189	**0.087**	**0.077**	0.218	0.170	0.144	0.299	0.186	0.135	0.515
$vr_{th} = 0.9$	0.033	**0.026**	0.086	**0.066**	**0.044**	**0.180**	0.097	0.086	0.195	**0.075**	0.077	**0.104**	0.082	0.036	0.143
$vr_{th} = 0.8$	0.033	0.027	0.087	0.081	0.052	0.230	0.088	0.085	**0.162**	0.077	0.076	0.111	**0.064**	**0.021**	**0.112**
$vr_{th} = 0.7$	0.033	0.028	0.083	0.092	0.069	0.257	0.096	0.077	0.195	0.078	**0.075**	0.109	0.091	0.040	0.173
$vr_{th} = 0.6$	0.043	0.038	0.097	0.073	0.053	0.199	0.158	0.123	0.305	0.102	0.098	0.232	0.279	0.193	0.950
VP [Meilland and Comport, 2013a]	-	**0.018**	**0.066**	-	-	-	-	0.144	0.339	-	**0.093**	**0.116**	-	-	-
ICP+RGB-D [Whelan et al., 2013a]	-	0.069	0.234	-	-	-	-	0.158	0.421	-	0.119	0.362	-	-	-
6D RGB-D odometry [Dong et al., 2014]	-	-	-	-	-	-	0.095	**0.067**	0.254	0.197	0.174	0.416	-	-	-
SDF tracking [Bylow et al., 2013]	0.035	-	-	0.062	-	-	0.078	-	-	-	-	-	0.040	-	-
RGB-D SLAM [Endres et al., 2014]∗	0.023	-	-	**0.043**	-	-	0.084	-	-	0.057	-	-	0.032	-	
MRSMap [Stuckler and Behnke, 2012]∗	0.043	-	-	0.049	-	-	0.069	-	-	0.052	-	-	0.042	-	
RGB+D+KF+Opt [Kerl et al., 2013a]∗	**0.021**	-	-	0.046	-	-	**0.053**	-	-	**0.017**	-	-	0.035	-	-
RGb-D SLAM Vol. Fusion [Whelan et al., 2014]∗	0.037	0.031	0.078	0.071	-	-	0.075	0.068	0.231	0.034	0.028	0.079	**0.030**	-	-

∗ with loop closure and pose-graph optimisation

Table 4.6: Absolute trajectory error (RMSE) in meters in the synthetic RGB-D dataset using different visibility ratio thresholds and comparison with state-of-the-art approaches

Dataset	Visibility ratio threshold					Best in
	No KF	0.9	0.8	0.7	0.6	[Handa et al., 2014]
office_tr0	0.1081	0.0921	0.0253	**0.0145**	0.0193	0.0216
office_tr1	0.0715	0.0384	0.0145	**0.0114**	0.0147	0.3778
office_tr2	0.0956	0.0327	0.0080	**0.0067**	0.0086	0.0109
office_tr3	0.0547	0.0550	0.0201	0.0108	**0.0084**	0.0838
livingRoom_lt0	0.0982	0.0365	0.0102	**0.0061**	0.4846	0.0724
livingRoom_lt1	0.0476	0.0229	0.0153	0.0162	0.0113	**0.0054**
livingRoom_lt2	0.1798	0.1683	0.0780	0.0109	**0.0108**	0.0154
livingRoom_lt3	0.1131	0.0950	0.0553	0.0365	**0.0355**	0.3554
office_tr0_noNoise	0.0040	0.0040	0.0040	0.0040	**0.0027**	0.0029
office_tr1_noNoise	0.0114	0.0114	0.0229	0.0096	**0.0032**	0.0385
office_tr2_noNoise	0.0293	0.0296	0.0295	0.0309	0.0085	**0.0016**
office_tr3_noNoise	0.0341	0.0341	0.0427	0.0222	0.0128	**0.0021**
livingRoom_lt0_noNoise	0.0431	0.0419	0.0403	**0.0099**	0.0183	0.1138
livingRoom_lt1_noNoise	0.0123	0.0123	0.0126	0.0071	**0.0023**	0**.0023**
livingRoom_lt2_noNoise	0.0058	0.0058	0.0058	0.0042	0.0028	**0.0015**
livingRoom_lt3_noNoise	0.0166	**0.0136**	0.0307	0.0319	0.0499	0.0200

the one resulting from using a model-to-frame ICP algorithm originally used in KinectFusion. Surprisingly, though our approach computes the odometry in a frame-to-frame fashion we obtain the best accuracy in most of the sequences with simulated noise and some of the noiseless ones. A qualitative evaluation of the trajectories for all the sequences using different visibility ratio thresholds is shown in Figs. 4.5 and 4.6. As occurs with the $fr2/desk$ dataset, quantitative and qualitative results show that using a keyframe switching strategy provide a better accuracy.

4.5.4. Odometry covariance and image filtering

In the proposed approach we initially avoided to apply any filtering to the intensity and inverse depth maps in order to conserve the raw sensor measurements and we dealt with the sensor noise by computing the scaling parameters of photometric and geometric residuals. As shown in the experiments on structurally and texturally rich scenes, the effect of not filtering the noise showed not to be harmful, obtaining accurate motion estimates.

However it must be noted that uncertainty in the odometry estimate is modeled by its covariance matrix which is computed as the inverse of the Hessian at the last iteration of the minimisation problem. The Hessian depends on the intensity and inverse depth gradients computed at the reference frame. This means that if the captured environment is poorly textured and highly planar, *e.g.*, a white wall, the Hessian must be nearly singular and thus the problem is ill conditioned.

We noted nevertheless that in some tests with planar and poorly textured scenes this hypothesis was not verified, due to the noise making the Hessian well conditioned without actually providing useful information. For this reason we noticed that, though the motion estimate would be wrong as well, applying a bilateral filter [Tomasi and Manduchi, 1998] only as a prior step to the computation of the gradients of the intensity and depth maps, would likely produce ill-posed Hessians when they are meant to arise, and thus help to detect poor odometry estimates.

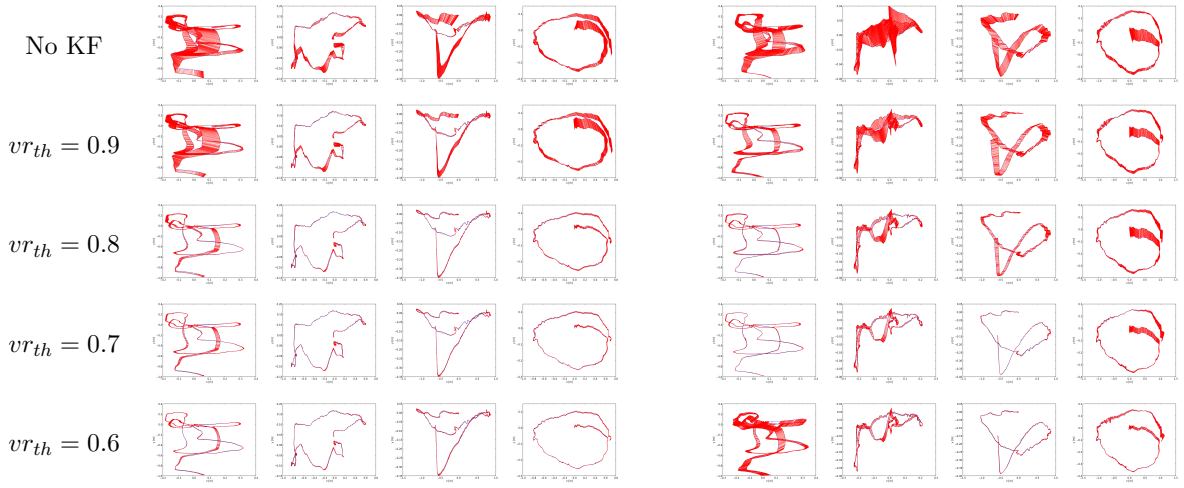


Figure 4.5: Trajectories on office (a)-(d) and living room (e)-(h) synthetic datasets with simulated noise. Estimated trajectory is shown in blue, ground truth is in black. Error between visual estimate and ground truth is shown in red.

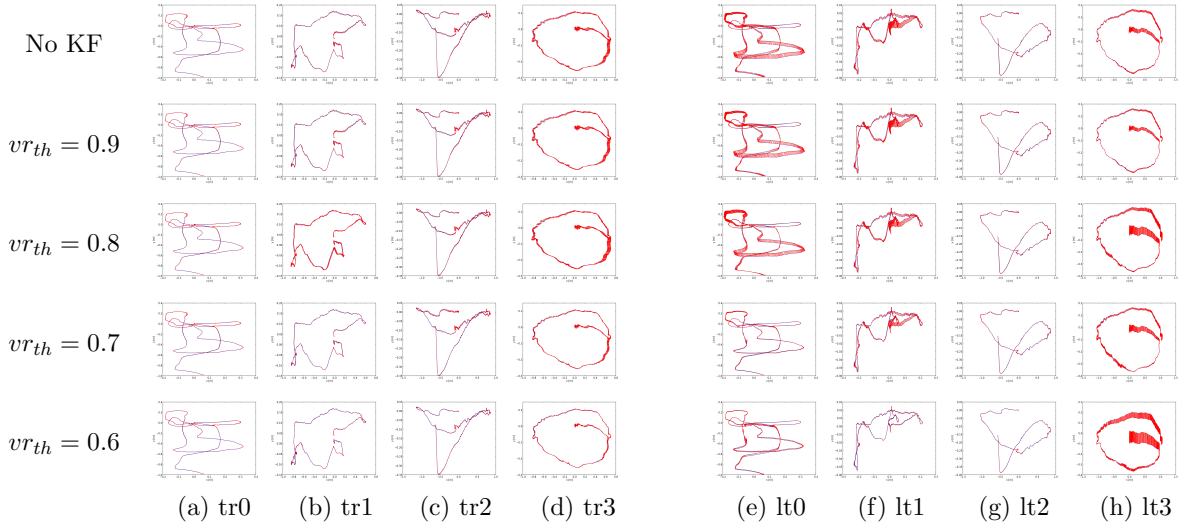
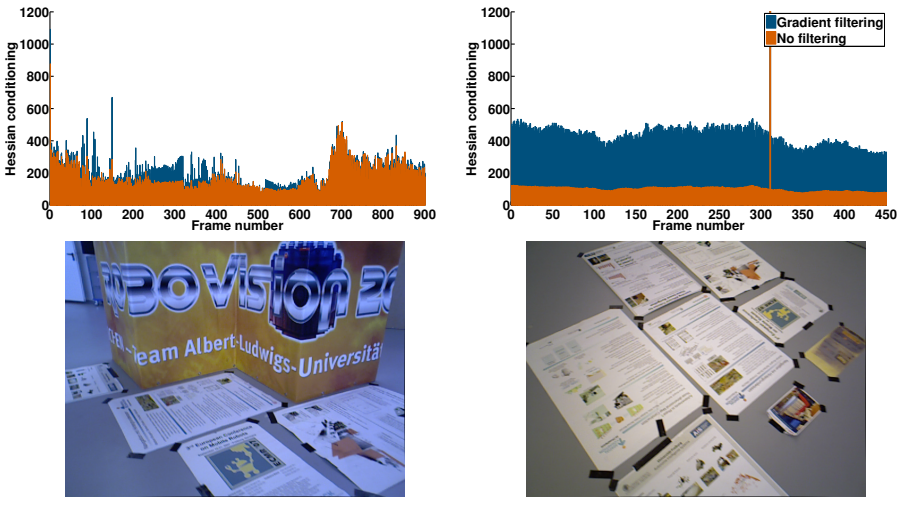


Figure 4.6: Trajectories on office (a)-(d) and living room (e)-(h) synthetic datasets without noise. Estimated trajectory is shown in blue, ground truth is in black. Error between visual estimate and ground truth is shown in red.



(a) (b)

Figure 4.7: Hessian conditioning with and without filtering of the inverse depth gradient map in visual odometry with geometric error minimisation only in (left) structurally rich sequence and (right) structurally poor sequence.

To evaluate how the noise filtering affects the estimate of the odometry covariance we propose to evaluate the condition number of its inverse, the Hessian, for the case of minimising only on the geometric error. Looking at (4.17) it can be verified that when viewing a plane perpendicular to the camera optical axis, $\nabla \mathcal{W}_A(\mathbf{p}) = (0,\ 0,\ 0)$ for every pixel and thus the Hessian of the resulting linear system when minimising the residual becomes singular. This occurs because the translation on the image axes and rotation around the optical axis are unobservable, while the rotation around camera axes and translation on the optical axis are still observable. A similar reasoning can be applied with respect to the movements on the axes parallel and perpendicular to a plane with arbitrary orientation viewed by the camera.

The condition number of a Hessian $\mathbf{H}$ can be defined as the quotient between its highest and lowest singular values:

$$\kappa(\mathbf{H}) = \frac{\sigma_{max}(\mathbf{H})}{\sigma_{min}(\mathbf{H})} \tag{4.34}$$

If $\mathbf{H}$ is singular then $\kappa(\mathbf{H}) \to \infty$. This means that a higher $\kappa(\mathbf{H})$ indicates a more ill-conditioned matrix and thus less accurate estimates in some of the motion degrees of freedom.

The evaluation of how bilateral filtering helps into a better estimation of the conditioning of the Hessian was performed on two sequences of the TUM RGB-D dataset: one showing a rich structure, composed by several planes and one showing a poor structure, with only one plane (Fig. 4.7). It can be observed that the effect on the Hessian conditioning of filtering the inverse depth map is almost unnoticeable for the scenario with rich structure, but in the case of a planar environment filtering makes more evident the ill-posedness of the Hessian, which otherwise would likely remain unnoticed.

4.5.5. 3D reconstruction

Though for volumetric 3D mapping we use the original functions in KinectFusion, a good quality of the reconstructed dense 3D volume depends critically on the drift introduced by the

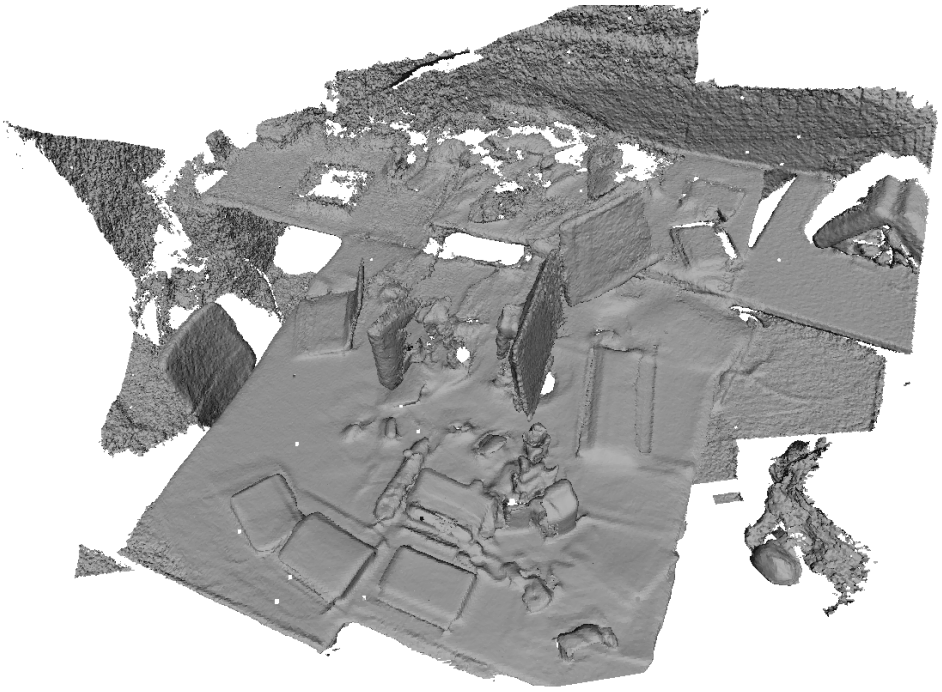


Figure 4.8: Dense 3D reconstruction of the $fr1/desk$ dataset.

visual odometry algorithm. Thus, we also present qualitative results of our approach showing the reconstruction of some of the tested RGB-D datasets from the TUM, and also two different datasets acquired by us in one laboratory of approximately 90 m^2 and the corridor of our department with a length of more than 80 m.

Our acquisitions were carried out with an Asus Xtion Pro RGB-D camera attached to a laptop by an arm-clamp system. The camera was only calibrated with a linear pinhole model without distortion parameters for the RGB sensor. The depth sensor is not calibrated, taking the depth values directly as provided by the sensor; and we use the hard-coded stereo pair calibration for depth and RGB registration. All the reconstructions are obtained just using our modified version of KinectFusion without performing loop closure.

Qualitative results for the datasets $fr1/desk$, $fr1/room$ and $fr2/desk$ are shown in Figs. 4.8, 4.9 and 4.10. It can be observed the great level of detail in $fr1/desk$, which indicates a low drift in the reconstruction. In the $fr1/room$ there are some zones, like the table at right, where the quality of the 3D reconstruction is poor. This occurs generally when mapping the same area under large camera motion and when revisiting a previously mapped place. In these cases new depth maps integrated into the mapped volume conflict with the stored map generating artefacts. However for zones which are swept during less time, as occurs in the rest of the sequence, the drift during mapping is low and the map reconstruction is more accurate. In the $fr2/desk$ the same zone is being constantly mapped moving slowly the camera in a loop around the desk. The final reconstruction shows a high precission without having applied loop closures nor any type of map correction.

Results for our laboratory and the corridor sequences are shown in the figure 4.11 and 4.12 respectively. The accuracy of the reconstruction can be assessed from the comparison to RGB images from similar points of view. Note also that given that our method is frame-to-frame, the drift both in the laboratory, reflected in the mismatch in the right wall, and in the corridor,

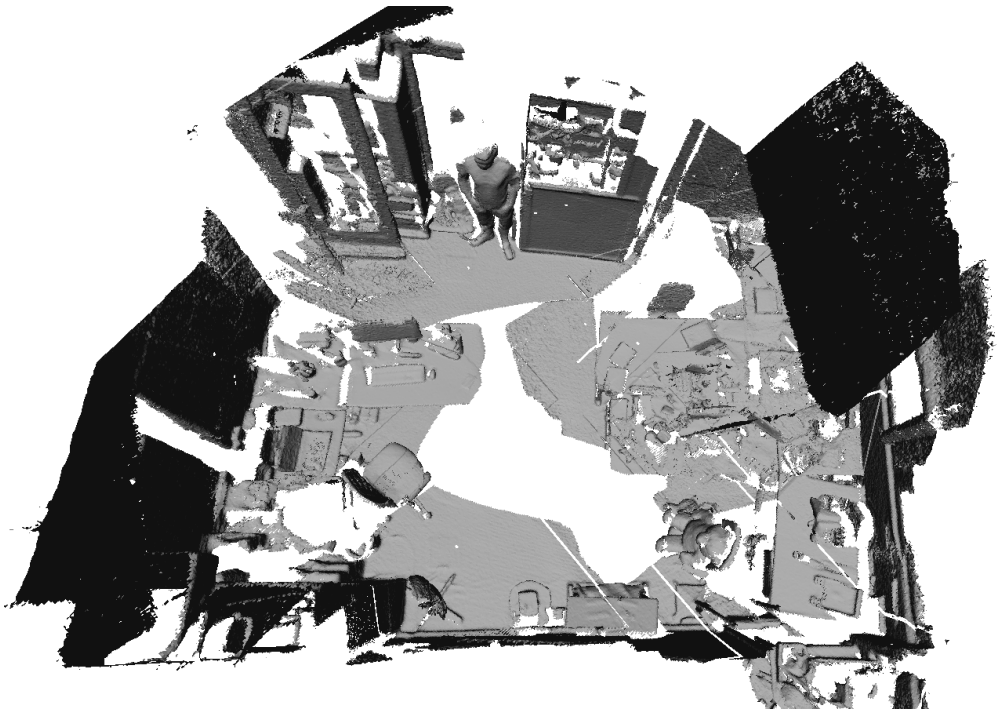


Figure 4.9: Dense 3D reconstruction of the $fr1/room$ dataset. Note how the shape of the room is accurately captured. Black part on the right top corner of the $fr1/room$ map corresponds to the ceiling reconstruction viewed from outside the volume.

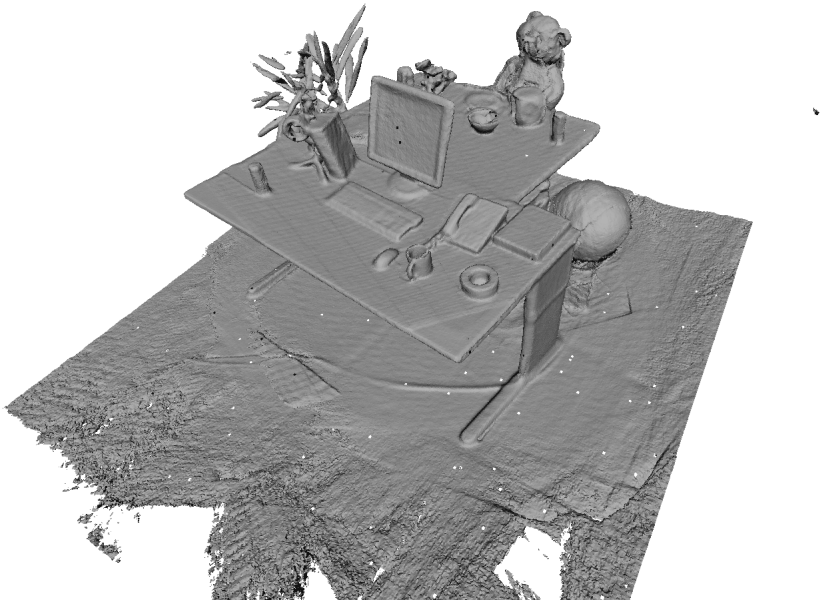


Figure 4.10: Dense 3D reconstruction of the $fr2/desk$ dataset. Note the high accuracy of the final reconstruction without having performed loop closure.

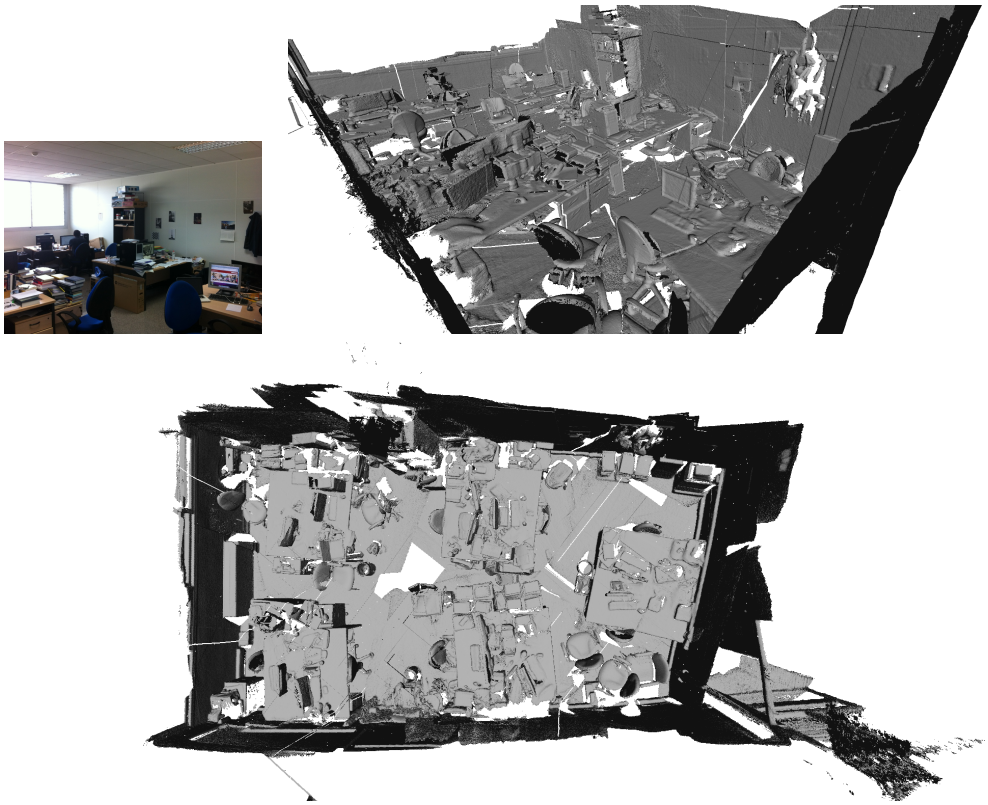


Figure 4.11: (Top-left) RGB image of the laboratory, (Top-right) KinectFusion 3D reconstruction using our method for visual odometry and (bottom) plant view of the complete 3D mesh.

reflected in the slight curvature of its side view, are relatively low.

4.6. Discussion

In this chapter we have presented a new visual odometry system on GPU based on the alignment between consecutive frames by minimisation both on the photometric and geometric error. Our system is implemented as an extension of the KinectFusion implementation Kinfu Large Scale in PCL, where the original ICP algorithm for frame alignment and visual odometry computation has been completely substituted by our method. The main contribution of our proposal is using the inverse depth instead of the depth to parametrise the geometric error, as well as allowing to switch between different robust estimators, residuals' scale estimators or geometric error parametrisation for comparative purposes. Our method shows its competitiveness with other state-of-the-art methods outperforming them in the majority of the tested datasets in terms of Relative Pose Error (RPE) and showing low Absolute Trajectory Error (ATE) in spite of not performing loop closure. With the introduction of some changes to increase the computational performance our system is able to reach a performance above 30 Hz with the GPU device nVidia GeForce 660GTX used in the experiments, without hindering the accuracy of the method. Also we show that extensions on the method such us taking reference frames for odometry estimation and performing bilateral filtering on the gradients of the images can first improve the accuracy when the camera moves slowly, and secondly allow for a better detection of bad odometry estimates.

Figure 4.12: (Top-left) RGB image of the corridor, (Top-right) KinectFusion 3D reconstruction with our visual odometry method, (middle) side view and (bottom) plant view of the complete 3D mesh. Note the challenging of the sequence due to the poor texture of the corridor and light reflexes.

Chapter 5

What Should I Landmark? Entropy of Normals in Depth Juts for Place Recognition in Changing Environments Using RGB-D Data

One open problem in the fields of place recognition and mapping is to be able to recognise a revisited place when its appearance and layout have changed between visits. In this chapter, we investigate this problem in the context of RGB-D mapping in indoor environments. We propose to segment the scene in juts (neighbourhood of 3D points with normals that stick out from the surroundings) and look at low-level features, like textureness or entropy of the normals. These could differentiate those zones of the scene that change or move along time from those that are likely to remain static. We also present a method which improves the matching between images of the same place taken at different times by pruning details basing on these features. We evaluate on a number of communal areas and also on some scenes captured 6 *months apart. Experiments with our approach, show an increase up to* 70% *in inlier matching ratio at the cost of pruning only less than* 20% *of correct matches, without the need of performing geometric verification.*

5.1. Introduction

In computer vision, the problem of place recognition consists in being able to tell if two given images correspond to the same scenario or not. Place recognition is the key element to perform topological mapping [Ulrich and Nourbakhsh, 2000], but also it is important in the context of geometric localisation and mapping to relocalise when the system is lost or to be able to close loops when revisiting previously mapped areas [Williams et al., 2011].

Robust place recognition is a fundamental step to perform life long mapping. In [Konolige and Bowman, 2009], Konolige and Bowman give a concise definition of what a lifelong map implies. A lifelong map system must carry out an incremental mapping, be able to operate in dynamic environments and to relocalise and close loops by recognising revisited places when given the chance. Focusing on the dynamic environment problem, two different issues are pointed: ephemeral objects, which move at the same time a zone is being mapped, and long-

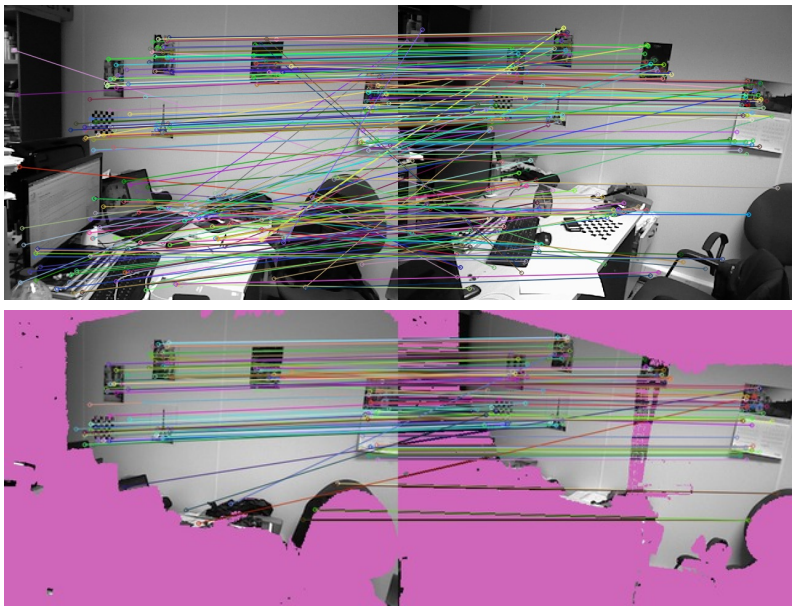


Figure 5.1: Depth-based detection of zones which moved between mappings. A simple masking based on the entropy of different zones of the scene improves the inliers ratio of matched visual features.

term changes, which involve changes in the scene which take place between different tracks of the scene. Ephemeral objects are usually ignored during mapping when using the traditional approaches such as visual odometry, and this makes such maps brittle beyond a few moments after being captured. In some cases, optimisation techniques based on RANSAC [Civera et al., 2010] or robust M-estimators [Kerl et al., 2013b] could be used to diminish the effect of ephemeral objects on the localisation and mapping, but this can only be considered an indirect approach to address the fundamental nature of scene change.

Naturally occurring long-term scene changes, such as objects on a common room or bedroom appearing, disappearing or changing places can affect loop closure detection and this thus must be dealt with at the place recognition stage. Standard approaches for place recognition though perhaps featuring some robustness to scene changes, do not handle these alterations actively nor identify where potential future changes may occur.

In this chapter we address the issue of place recognition under long-term changes. Concretely we focus on robust place recognition with RGB-D cameras in indoor scenes. From the point cloud and the RGB images we can compute low-level features of parts of the scene like textureness, planarity or entropy of normals. This information can be very valuable to both decide where not to place landmarks but also to identify moving scene objects.

To define and evaluate our proposed method we perform two types of experiments. The first experiment is performed with images taken at fixed poses of different real scenes over which we have no control of changes and over different days, and where Ground Truth of static and changing zones is available. This experiments aims to inform us on which features are better to discriminate static from changing scene parts.

In the second experiment, we compute 3D maps of some of the scenes on different days with KinectFusion method [Newcombe et al., 2011a], which is more akin to the case of a system of lifelong mapping for a personal or robotic device. Here we evaluate the matching between snapshots from reconstructions on different days by masking out zones below a given score,

based on the previously studied features. We show how this masking improves the inlier ratio (Fig. 5.1).

5.2. Related work

The work of Konolige and Bowman [Konolige and Bowman, 2009] is focused towards efficient storage, keeping a map whose size scales with the explored area instead of with time. Keyframes are clustered by appearance similarity, trying to keep the maximum number of clusters, while fixing the maximum number of total stored keyframes. However this work does not address the problem of maintaining matches over long term changes, but propose the idea of picking up stable features as a path for improvement of long-term life-long mapping.

In the context of place recognition with RGB-D cameras Gee and Mayol-Cuevas [Gee and Mayol-Cuevas, 2012] pointed out the robustness to map changes as one important property for place recognition. They propose a regression of small synthetic views of 80x60, or even 20x15 pixels, which offers a degree of tolerance to changes in order to recognise revisited places. Whelan et al. [Whelan et al., 2013b] focus on using optimisation to produce a consistent map deformation when closing the loop and address the problem of place recognition by using a visual based bag of words scheme with SURF descriptors.

Also, when considering RGB-D sensors, many authors have focused in discovering moving objects in a scene. This problem is related to lifelong mapping in the sense that moving objects are precisely what we want to discard when performing place recognition. Herbst et al. [Herbst et al., 2011] discover objects in the scene after aligning two 3D maps of the same scene with different objects. For the alignment they assume that the moving objects occupy a small fraction of the map. Finman et al. [Finman et al., 2013] detect objects from changes in maps and then train a segmentation method to segment the discovered objects in future runs.

Karpathy et al. [Karpathy et al., 2013] propose a method to discover objects, which in contrast does not rely in an object displacement to detect them. Instead they perform a non-semantic segmentation [Felzenszwalb and Huttenlocher, 2004] of the scene based on the map of normals, and then each segment is ranked with different objectness features. To avoid ambiguities, in this work we name the 3D segments as superjuts as the result of extending the concept of superpixel to 3D points with normals.

However, when dealing with lifelong mapping with RGB-D, though recognising specific objects can be useful, it would be desirable to discard anything which can move, no matter which object it is or if it has been seen or not before. To do this, we can take advantage of some results and proposals from works in object discovery, like for example a non-semantic superjut segmentation. But instead of detecting objects explicitly, we propose the use of statistics and entropy of low-level features to identify areas that are good and those that are likely to be unstable landmarks. This approach removes the elusive definition of object.

When looking for features which characterise movable objects, some ideas can also be drawn from supervised segmentation methods for RGB-D scenes [Cadena and Kosecka, 2013], [Ren et al., 2012], [Silberman et al., 2012], [Couprie et al., 2013]. These methods usually classify the scene into coarse classes (*e.g.*walls, floor, furniture, and props). The "props" class usually makes reference to objects which can be easily carried, which corresponds precisely to the kind of objects we want to remove from place recognition routines. Thus, features used in these methods as inputs for categorisation can provide some clues for the task of discarding movable objects.

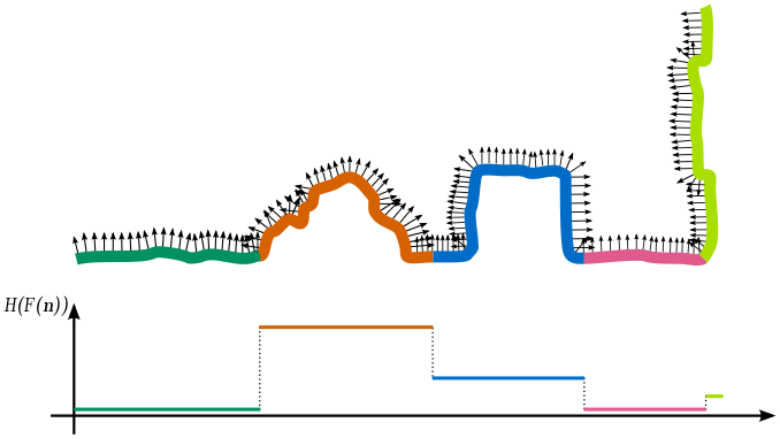


Figure 5.2: 2D simplified scheme of the segmentation in superjuts and the entropy of the normals $H(F(\mathbf{n}))$ of each superjut.

5.3. Proposed method

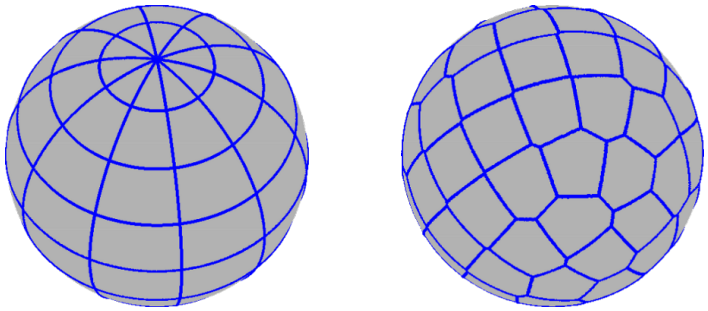
Our proposal for robust place recognition tries to detect zones which can move no matter which objects are in them, using the information obtained from both 3D point clouds and RGB images. Fig. 5.2 shows an intuition of the proposed method. With a point cloud segmentation algorithm based on the similarity of normals between adjacent points [Karpathy et al., 2013], a point cloud is segmented in superjuts, represented with different colours. Then at each superjut we extract low level features which allow to discriminate static from moving parts of the scene. Looking at everyday indoor environments one can probably note that zones of the environment which are likely to change often show a high degree of derangement, which yields a more cluttered map of normals and more textured areas in the image. In the figure for example, the entropy $H(F(\mathbf{n}))$ of the histogram of normals is represented, noting that for superjuts with irregular shape is greater than in superjuts composed by planar surfaces.

5.3.1. Point cloud segmentation

First we use the method proposed in [Triebel et al., 2010] and extended in [Karpathy et al., 2013] to segment a dense 3D point cloud in objects or small groups of objects. It is based on the segmentation algorithm by Felzenszwalb and Huttenlocher [Felzenszwalb and Huttenlocher, 2004], but applied on a map of normals instead of a RGB image. A graph $G = (V, E)$ is constructed, where each vertex in V represents a 3D point $\mathbf{X}_i$ and the edges E represent the neighbouring relations between points. Having the normals at every point, a weight w_{ij} for the edge joining vertices i and j is computed:

$$w_{ij} = \begin{cases} (1 - \mathbf{n}_i^T \mathbf{n}_j)^2 & \text{if } \mathbf{n}_j^T(\mathbf{X}_j - \mathbf{X}_i) > 0 \\ 1 - \mathbf{n}_i^T \mathbf{n}_j & \text{if } \mathbf{n}_j^T(\mathbf{X}_j - \mathbf{X}_i) \leq 0 \end{cases} \tag{5.1}$$

where the squared weight is applied to convex edges, reflecting the fact that convex regions usually contain points belonging to the same object and concave regions are likely to arise in frontiers between objects. After computing the weights, the segmentation algorithm is run and essentially groups points sharing edges with low weights in the same segment. A parameter k



(a) (b)

Figure 5.3: Two possible binnings of the 3D sphere to compute the histogram of normals: (a) by discretising azimuth and elevation angles and (b) by approximate uniform distribution of points in the sphere. The number of bins is set to $M = 80$ in both cases.

must be tuned such that the higher k, the larger segments will be obtained. In this work we always use $k = 0.6$ for all the experiments.

Depending on the characteristics of the point cloud we compute the normals and edges required for segmentation in two different ways.

Triangular mesh: The first option is to build a triangular mesh from a given point cloud. This is specially preferable if we have dense 3D map resulting from mapping as the camera moves, e.g. with KinectFusion [Newcombe et al., 2011a] approach. This way, an edge between two points $\mathbf{X}_i$ and $\mathbf{X}_j$ is created if they have any triangle, among the ones they belong to, in common. The normal for a point is computed as the average of the normals of the triangles it belongs to. Using this approach the segmentation algorithm is like the one proposed in [Karpathy et al., 2013], resulting in a set of superjuts.

Image domain: Alternatively the point cloud can be projected onto an image domain, for example if the point cloud has been cast from a single depth image and the 3D map might be so noisy to build a triangular mesh. In this case the segmentation would be done directly in image projected superjuts, with the edges being created between adjacent pixels horizontally, vertically and diagonally. The normal for a point a 3D point projected in pixel (k, l) is computed as:

$$\mathbf{n}_{k,l} = \frac{(\mathbf{X}_{k,l+1} - \mathbf{X}_{k,l}) \times (\mathbf{X}_{k+1,l} - \mathbf{X}_{k,l})}{||(\mathbf{X}_{k,l+1} - \mathbf{X}_{k,l}) \times (\mathbf{X}_{k+1,l} - \mathbf{X}_{k,l})||} \tag{5.2}$$

5.3.2. Computation of superjut low-level features

Once the segmentation is done, we extract the low-level features to discriminate static from moving parts of the scene. We take the hypothesis that parts which are likely to move can be discriminated by their 3D structure and their textureness.

Structure by histogram of normals

To gather information about the structure of each superjut we compute its histogram of normals $F(\mathbf{n})$. Since normals are 3-dimensional unit vectors, they can be represented as points over the surface of a unit sphere. Thus, a first attempt would be to compute a 2-dimensional histogram by binning the angles of azimuth ϕ and elevation θ as proposed in [Tang et al., 2012]. However we noted that this binning might not be adequate if we want to obtain a good distribution of the bins on the sphere. This is graphically shown in Fig. 5.3a, where it can

be noted that the area covered by each bin greatly changes with the elevation angle. Also it must be noted that points situated near the poles would be spread among all the confluent bins. These two facts can severally affect the quality of the histogram.

To overcome these problems we opt instead for the computation of a 1-D histogram with M bins. Each bin is a Voronoi cell corresponding to one point out of M points uniformly distributed over the sphere. Since there is no analytical solution to the problem of evenly distributing M points on the sphere for any M, we use a simple solution based on the golden section spiral [Boucher, 2006], which results in a discretisation with bins covering areas of similar size and avoids the confluence of more than 4 bins in a single corner (Fig. 5.3b).

Given the histogram of normals, we extract the following properties:

- Entropy of the histogram $ent(F) = \sum_{i=1}^{M} F(\mathbf{n}_i) \log(F(\mathbf{n}_i))$.
- Planarity, measured as the relative frequency of the dominant normal $plan(F) = \max F(\mathbf{n})$.
- Horizontality, given by $hor(F) = \mathbf{e}_\mathbf{y}^T \mathbf{n}_{mode}$, with $\mathbf{n}_{mode} = \arg\max_\mathbf{n} F(\mathbf{n})$, where $\mathbf{e}_\mathbf{y}^T = (0\ 1\ 0)$.

Textureness from the eigen-transform

To obtain texture information we apply the eigen Transform proposed by Targui et al. [Targhi et al., 2006] over the grey scale images. This approach produces a grey-scale map, where more textured areas of the image yield a higher response than less textured ones. Then for each superjut projected in a superpixel over the intensity image, we compute the mean eigentransform value over the region it covers.

5.4. Experiments

We have performed two experiments. In the first experiment we recorded different scenarios from fixed poses in order to capture the changes which took place during time and facilitate a Ground Truth separation of moved and static zones, on which discriminative power of structural and texture characteristics can be evaluated.

In the second experiment, we recorded sequences of 3 scenarios on different days with a moving camera and reconstructed a 3D volume to evaluate the use of structural information with more accurate depth maps in the context of place recognition. To demonstrate the performance even with very long term changes one set of sequences was acquired more than 6 months after the first one.

5.4.1. Evaluating features for detection of moved areas

RGB and depth images have been taken on 9 scenes at different locations: two laboratories at different universities and one bedroom (Fig. 5.4). For each scene, images have been acquired on different days to capture the changes in their layout. Each day, the camera is carefully positioned at each of the selected poses so that the same scene is captured as in the acquisitions of previous days. To diminish the effect of illumination noise in intensity images, and decrease the holes in the depth images, we captured frames during some seconds and computed the average of all. In the averaging of the depth image, hole pixels are zero-weighted.

The first step is building a Ground Truth to separate static and moved zones of the different scenes. To do so, first, images of to the same scene on different days are robustly aligned to

Figure 5.4: Images taken on 9 different scenes at different locations (2 labs and 1 beedroom).

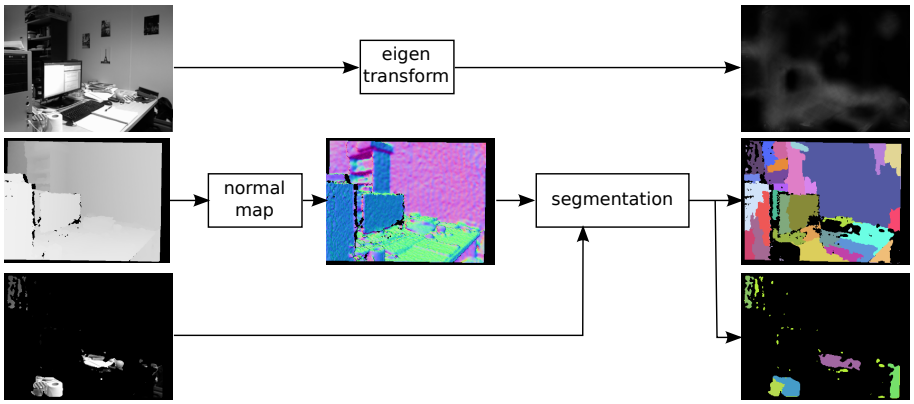


Figure 5.5: Starting with the aligned RGB and depth images and the masks for moved objects, we compute the eigen transform as well as a segmentation separated in static and moved superjuts. This data is used to compute superjut features for the first experiment.

reduce the disparity due to non exact camera positioning with respect to previous days. To do so we use a similar scheme as for computing the RGB-D dense visual odometry [Gee and Mayol-Cuevas, 2012].

Once we have the aligned images and depth maps, we have to extract the parts of the scenes which may correspond to moved objects. This is done by computing the disparity in the depth maps. Note, however, that the existence of a disparity does not imply a moved object since it might correspond to a static area in the background revealed by an object which was occluding it. For a depth map D_i of a scene M, the parts which have changed are those which show a negative disparity in any difference with every other of the N_M depth maps of that scene. Mathematically we can express this by:

$$maskMoved_i = \left(\sum_{j=1}^{N_M} \mathbf{1}_{x>th} \left(\text{medFilt}(D_j - D_i) \right) \right) > 0 \tag{5.3}$$

where $\mathbf{1}_{x>th}$ is an indicator function. To eliminate disparity caused by a non-perfect alignment between frames, all the disparity maps are median filtered with a window size of 10x10, and the threshold for the indicator function is set to $50mm$. After this step we have N_M masks per scene extracting the changing zones.

Once we have aligned RGB and depth images as well as computed the masks, we follow the pipeline shown in Fig. 5.5 to obtain the map of normals, the Eigen-transform and the segmentation, divided in static and moved superjuts. This data is used to compute both the structural and textureness features for the superjuts as described in section 5.3.2.

The results shown in Fig. 5.6 support the intuition of moving areas being more likely a higher entropy of the normals and lower planarity. It is shown also a correlation between entropy and planarity, which is not a surprise since the existence of a dominant bin in a histogram reduces its entropy. Texture information from the Eigen transform and horizontality seems not to be very useful to discriminate static from moved areas.

In some scenes however, distinction of moved from static areas based on the used features is less clear. It must be noted that in this experiment we lack from some accuracy due to the restriction of recording scenes from fixed poses. On the one hand the Ground Truth we take for moved and static elements is not perfect, due to residual misalignment of the images corresponding to the same poses and noisy depth measurements of far away areas of the scene. On the other hand, the segmentation on the image domain (Fig. 5.5) is not as good as computed on smooth 3D meshes as it is done in the next experiment (see $3th$ row of Fig. 5.7).

5.4.2. Improving place recognition on 3D meshes

For this experiment, we have selected 3 of the scenes captured in the previous experiment. These scenes have been selected due to their availability to record sequences on them. The idea is to prove that using information about 3D structure, based on the features proposed and analysed in previous experiment, to discard zones of the scene can benefit place recognition algorithms.

First, a point cloud and its corresponding triangular mesh are constructed for every scene for different days using the large scale KinectFusion implementation from the PCL library [Rusu and Cousins, 2011]. Then meshes are segmented using the algorithm of [Karpathy et al., 2013] and for every superjut, a histogram of normals is computed as explained in section 5.3. Since entropy was shown to be quite discriminative in previous experiment, we use it as the only feature to prune moved parts of the scene. Each segment is scored then by computing its normalised neg-entropy:

$$negent(F) = 1 - \frac{ent(F)}{ent_{max}(F)} = 1 - \frac{ent(F)}{\log(M)} \tag{5.4}$$

To evaluate our approach we match keypoints between snapshots selected from the image sequences of the same scene at different days, using SURF descriptors. During mapping, the poses of the camera are saved so that the 3D point cloud can be projected and then obtain the scores and the segmentation in the corresponding 2D image domain (Fig. 5.7). The matching pipeline was set to take keypoints with a Hessian score greater than 200 and discard matches with a ratio greater than 0.85 for the distance between the two best matching hypotheses.

For a given image pair, a Ground Truth set of inliers I_{GT} is obtained by computing a robust fundamental matrix by RANSAC to select the geometrically consistent matches. On the other hand, the set of successful matches S_{th} for a given threshold th for the normalised neg-entropy score are those whose keypoints do not belong to any superjut with a score below the threshold. Given these definitions we can compute the precision and the recall for a given threshold

$$precision(th) = \frac{\text{size}(I_{GT} \cap S_{th})}{\text{size}(S_{th})}, \tag{5.5}$$

$$recall(th) = \frac{\text{size}(I_{GT} \cap S_{th})}{\text{size}(I_{GT})} \tag{5.6}$$

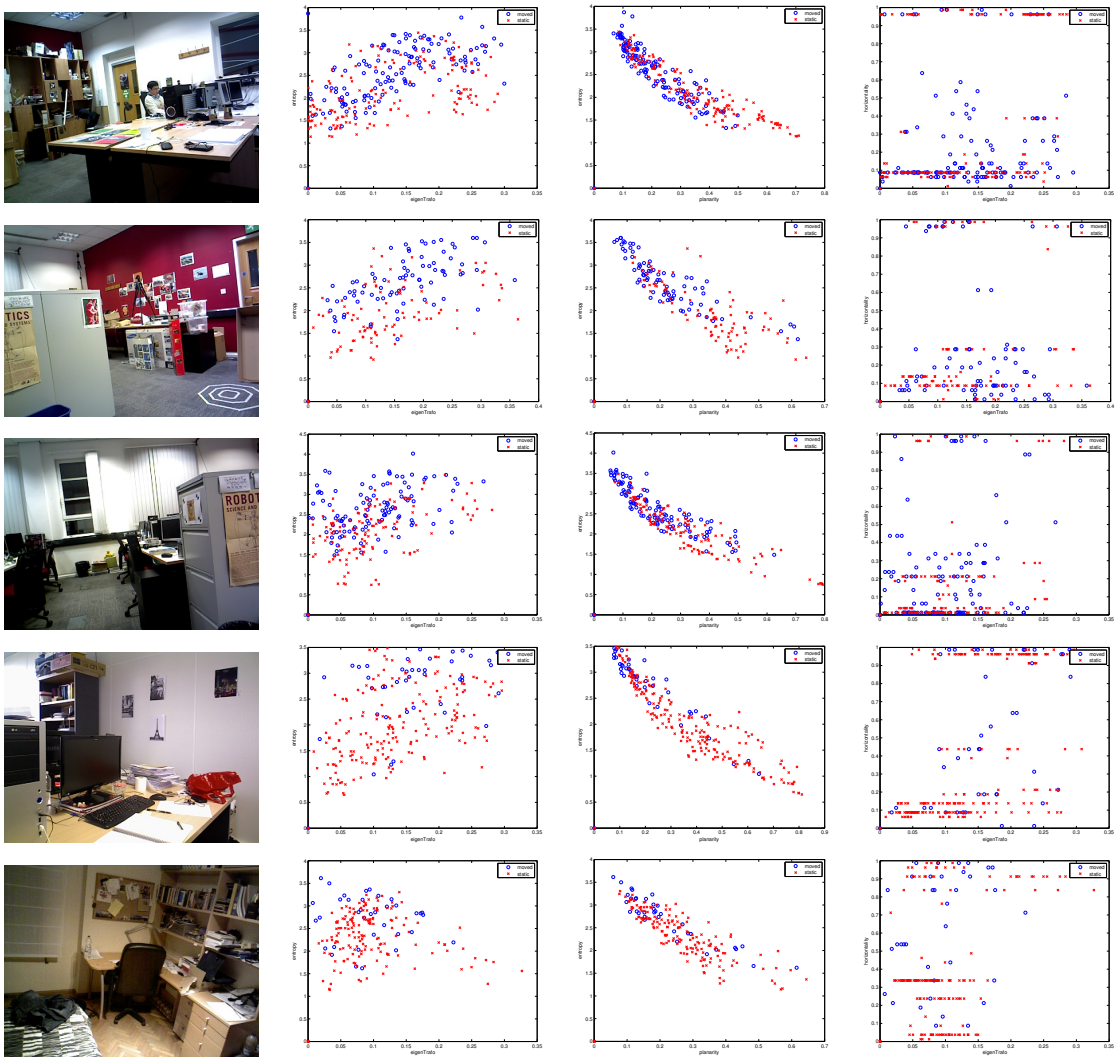
For every scene, we have computed the precision-recall curves on each of the possible combination of image pairs, by shifting the neg-entropy threshold th to prune likely-to-move areas. Fig. 5.8 show how the masking varies for different values of th. In Fig. 5.9 it is shown how the matching is affected only between image pairs corresponding to pairs of sequences acquired with 6 months of difference, for each of the 3 considered scenes. Note that for pairs of images where the zones of high entropy have been masked we obtain a higher ratio of inliers than matching the raw images. The neg-entropy threshold established for the masking is in each image pair, the one which produces the highest peak in the precision-recall curve. It can be observed that the zones that remain unmasked correspond in their majority to planar zones like walls, which contain very stable features. On the other hand masking can also eliminate zones which remain static between acquisitions, such as the selves in the *roomBed* scenario. Also planar objects which are likely to change their position, such as laptop screens can remain unmasked. However, in spite of discarding some stable zones and accepting unstable ones for matching, the precission and recall curves indicate that the effect of that masking is beneficial for some neg-entropy threshold.

In Fig. 5.10, we show the average precision and recall curves for each scene, including all the possible combinations of image pairs from all the sequences acquired for the 3 scenarios. Note that for the first scene (*labDesk*) we obtain an increase in precision of more than 75% with a decrease in recall of only 20% with respect not using any masking. This is caused by the presence of textured elements on walls which leads to a great number of matches in a low entropy zones. However, in the third scene (*roomBed*), the lack of highly textured low-entropy areas makes the beneficial effect less noticeable.

Note that though being evaluated in an image matching context, our approach computes the masks prior to the matching process. Considering this fact and also that a greater inliers rate between two matched images is caused by a greater appearance similarity between both images, our approach is likely to be applied within the framework of efficient image search algorithms [Cummins and Newman, 2008] increasing their performance. Though the computational cost of performing the mesh segmentation and superjut projection and scoring is relatively high, around the order of 5 seconds, it must be noted that every time a new keyframe is tested for place recognition the cost of our approach is constant and does not grow linearly with the number of frames as it surely would a computationally costly exhaustive matching and geometric verification scheme over all the keyframe search space.

5.5. Discussion

This work is concerned with the problem of long-term indoor mapping with RGB-D cameras. Concretely we focused on finding and evaluating properties of the elements of the scene which can allow to discard parts of the scene which easily change with time and could affect severally to place recognition algorithms for example when relocalising or detecting loop closures. In the experiments we have shown that parts of the scene which move along time tend to present a more chaotic structure, which is reflected in a high entropy of the histogram of normals. We



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

Figure 5.6: Distribution of static and moved superjuts in (a) some of the tested scenes, for (b) entropy-eigen transform, (c) entropy-planarity, (d) horizontality-eigen transform scores. Superjuts corresponding to moved/static areas are shown in blue/red.

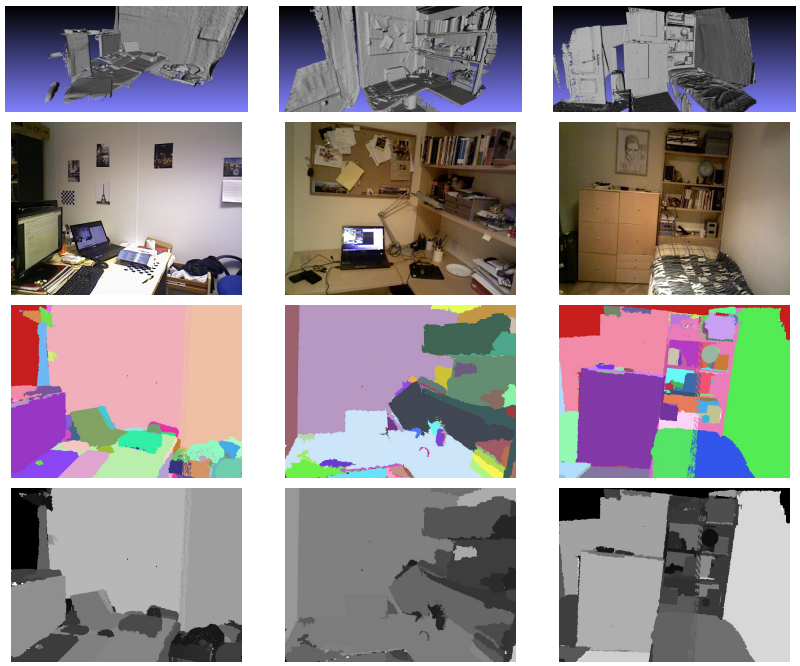


Figure 5.7: Samples of data used in the experiments with depth map. 1*st* row shows the obtained 3D maps, 2*nd* row represents snapshots selected for matching, 3rd row is the segmentation of the map of normals computed on the 3D mesh and 4th row shows the normalised neg-entropy for every superjut projected on the image.

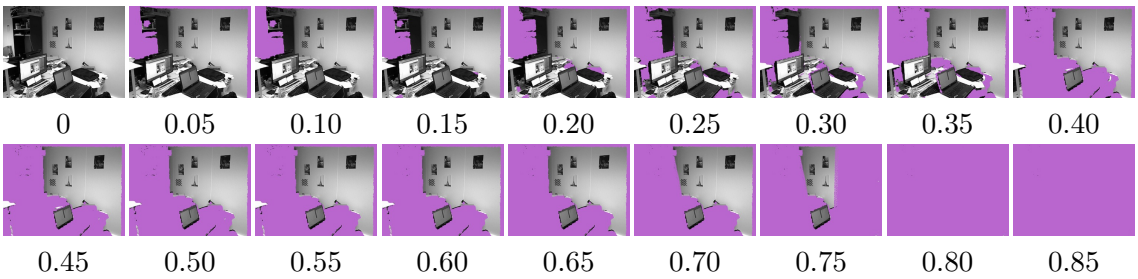


Figure 5.8: Masks obtained in one frame of the *labDesk* scene for different values of the score threshold *th* of the superjut's entropy of normals.

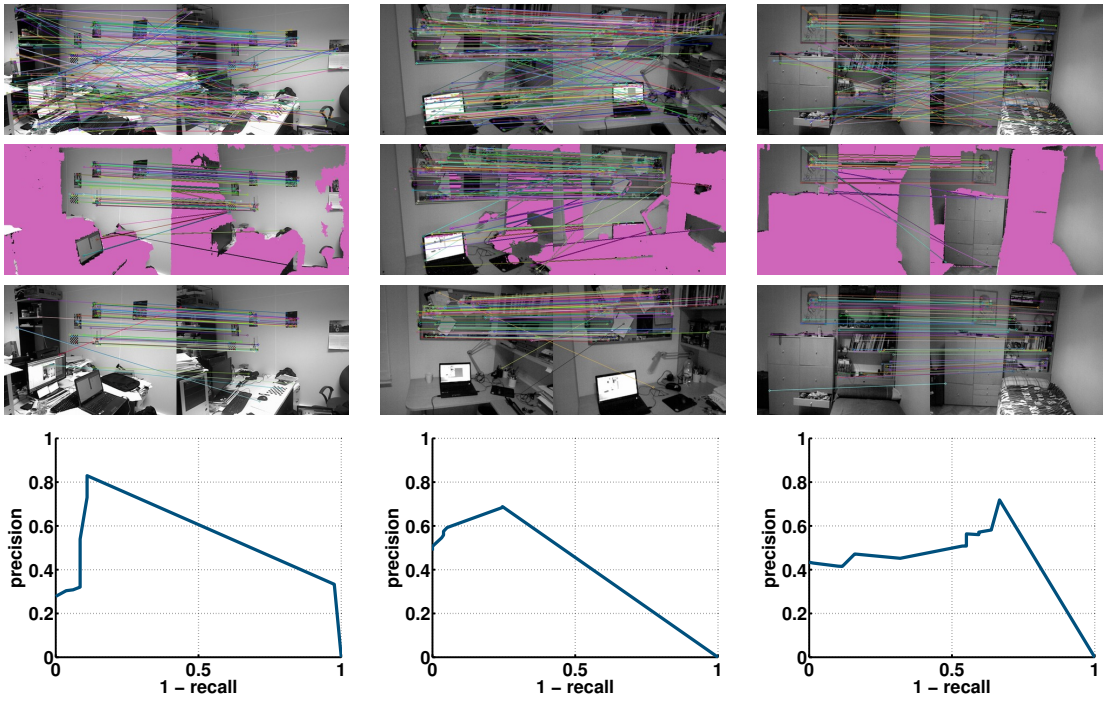


Figure 5.9: Evaluation of our method in 3 different scenes taken with 6 months of difference in an uncontrolled area (from left to right: *labDesk*, *roomDesk* and *roomBed*). (1st row) Unmasked raw matches, (2nd row) matches after masking for the score corresponding to the highest peak in the precision-recall curves in 4th row (masked areas are in magenta), (3rd row) Ground Truth matches by geometric consistency.

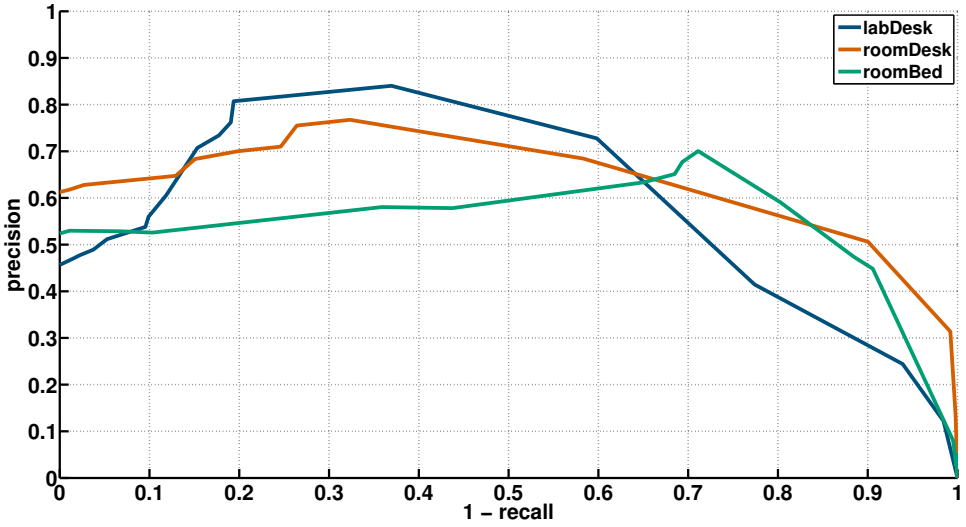


Figure 5.10: Average precision-recall curves for the 3 considered scenes

have validated this observation, by matching images of the 3D map built on different days, with a maximum difference between acquisitions of 6 months apart, for 3 selected scenes. Though the validation was performed in a standard matching scheme, the results are promising and can be extended to robustify appearance based place recognition approaches by allowing to prune words in high entropy areas. This is important considering that for large keyframe search databases appearance based methods are the alternative to the maybe more precise but also much more computationally expensive matching and geometric verification schemes.

Chapter 6

Robust RGB-ID SLAM in Changing Environments

In this Chapter we develop a complete RGB-D SLAM system, which builds upon our proposals presented in previous chapters. First, our accurate and efficient dense RGB-D odometry algorithm in GPU, and secondly our proposal based on point cloud segmentation for robust place recognition. Our system consists in 2 threads working in parallel. The first thread is a front-end operating at frame rate, which processes every incoming frame from the RGB-D sensor to compute the incremental odometry and integrate it in a keyframe which is changed periodically following a covisibility-based strategy. The second thread is a back-end which receives keyframes from the front-end. This thread is in charge of segmenting the keyframes based on their structure, describing them using Bags of Words, trying to find potential loop closures with previous keyframes, and in such case perform pose-graph optimisation for trajectory correction. The first experiments with our approach in the TUM RGB-D benchmark datasets show results superior in accuracy to the state-of-the-art in many of the sequences. These promising initial evaluation encourages us to further improve our method.

6.1. Introduction

In the last years visual SLAM has become one fertile research topic in the fields of computer vision and robotics. A complete visual SLAM system usually consists of the following modules:

- Camera tracking
- 3D reconstruction
- Place recognition and loop closure when a zone is revisited

To develop our complete SLAM system we build on our previous work presented in prior chapters. The camera tracking module builds upon our dense RGB-D odometry algorithm incorporating some minor improvements. The incorporation of a place recognition and loop closure module is new in this chapter and it is based in a Bag of Words scheme using ORB descriptors [Mur-Artal and Tardós, 2014]. Nevertheless we will use of the structure-based segmentation method discussed in previous chapter for pruning of potentially unstable parts of the scene.

Regarding the 3D reconstruction, though RGB-D sensors already provide a dense depth map of the scene, raw depth maps direct from the sensor show a large noise which grows quadratically with the distance of the observed point. For this reason, in order to obtain an accurate dense 3D reconstruction of the environment one needs to fuse multiple consecutive depth maps. In the literature we find that this fusion can be addressed by two different approaches: integration on a 3D volume of voxels [Newcombe et al., 2011a] or integration in keyframes wich are sampled from the sequence of frames [Meilland and Comport, 2013b].

Volumetric fusion can be viewed as the integration of every incoming frame within a set of concatenated 2D slices each slice corresponding to a depth value. This representation in many depth slices has the advantage of allowing for handling occlusions; however it demands large amounts of memory and also, in the process of quantisation and integration of depth measures in the volume, important information about the depth uncertainty is lost.

In keyframe fusion in turn, integration is performed in a single slice represented by the depth map of the frame at which the keyframe was initialised. The main benefits are two. First, a more lightweight and efficient memory representation and secondly, since integration is performed in the frame of the original image space without discretising the depth values, depth uncertainty can be rigorously handled. The main disadvantage of this representation is that, with a single slice, occlusions between elements in the scene cannot be handled. In our keyframe-based approach, since we generate new keyframes periodically this is not a problem. Also it is worth remarking that a keyframe representation comes on handy when performing applying state of the art appearance-based loop closure strategies in visual SLAM, since it provides a direct mapping of the 3D scene to the RGB image of the keyframe. Furthermore, in our keyframe integration scheme, as in our previous work on direct RGB-D odometry, we take advantage of the usage of inverse depth maps instead of depth maps, which allows us to use a constant uncertainty-based tolerance for the integration of a new inverse depth value.

6.2. Related Work

One of the earliest and most influential works on real time monocular SLAM was presented by Davison [Davison, 2003]. It is a visual feature-based SLAM system, where the camera motion and position of landmarks of the environment are estimated using an Extended Kalman Filter (EKF) from the image displacement of the image projections, or features, of such landmarks. Following the success of this work, several contributions introduced new improvements and functionalities like the inverse depth parametrisation of the landmarks [Civera et al., 2008], a relocalisation module [Williams et al., 2007], and efficient scheme for data association and outlier rejection [Civera et al., 2010] or a generalised projection model to support catadioptric sensors [Gutiérrez et al., 2011].

One weakness of this approach is that following a probabilistic filter paradigm, camera tracking and mapping are performed both at frame rate, which limits severally the number of landmarks which could be reconstructed. Klein and Murray [Klein and Murray, 2007] addressed this issue with their PTAM (parallel Tracking and Mapping), where camera tracking and mapping are separated into two different threads. The tracking thread operates at frame rate and is on charge of estimating the camera position given a fixed map; while the mapping thread operates at a lower rate, by building or updating a map by performing bundle adjustment between a set of keyframes selected from the total number of frames.

Basing on PTAM Mur-Artal et al. presented in [Mur-Artal et al., 2015] a new visual SLAM approach addressing some of the weaknesses of PTAM, like handling of landmarks occlusions or mapping scalability, as well as including a novel loop closure and relocalisation method based

on Bags of Binary Words using ORB descriptors.

In the last years visual SLAM has been has been very influenced by two differential events. First the development of a new generation of GPUs and CPUs, which allowed to design algorithms with high parallelisation; and secondly the advent of depth sensors, which are able to provide a dense depth image of the observed environment. The earlier prompted the development of real-time dense visual SLAM methods, while the later enabled SLAM, both feature-based and direct, using RGB and depth channels from a RGB-D camera.

The main challenge in feature-less or direct monocular SLAM methods is the lack of depth measurements from vision sensors. Maybe the earliest and most influencing work in real-time dense monocular method was DTAM (Dense Tracking and Mapping) presented by Newcombe et al. [Newcombe et al., 2011b]. The authors propose to reconstruct the depth of the scene in keyframes and at the same time estimate the pose of the camera. The problem of estimating a dense depth map from just RGB images is ill-posed and thus authors used an optimisation framework based on variational methods and implemented on GPU. In [Engel et al., 2013], Engel et al. propose a direct monocular SLAM method which obtains a semidense mapping based on variable baseline matching.

Maybe the best example of a feature-based RGB-D SLAM system is the approach proposed by Endres *et al.* [Endres et al., 2012]. To estimate the camera motion they first compute an initial seed by RANSAC-based 3D alignment of sparse features, followed by a refinement step where they only minimise the point cloud alignment error from the ICP algorithm. The method is improved in [Endres et al., 2014], including an Environment Measurement Model to prune wrong motion estimates which passed undetected in the RANSAC and ICP steps. The system is able to close loops taking a random sample from a set of keyframes and each frame-to-frame motion calculation between candidates for loop closure is parallelised, thus the computational cost is close to the frame rate of the camera, between 5 and 15 Hz.

KinectFusion by Newcombe *et al.* [Newcombe et al., 2011a] was a ground-breaking contribution in dense RGB-D SLAM. KinectFusion is composed by two different modules, one for camera tracking and one for dense volumetric mapping. For each new frame first the motion is estimated by frame-to-model ICP alignment of depth maps, *i.e.*, current depth map is aligned with the depth map raycasted from a voxelized 3D model. Then, current depth map is integrated in the 3D model using a truncated signed distance function. The main constraint of KinectFusion is its limitation to small workspaces, which was nevertheless solved in latter works by using a cyclical buffer to shift the volume as the camera explores the environment [Bondarev et al., 2013], [Whelan et al., 2013a]. However, artefacts are likely to appear in the reconstructed 3D model when an area is revisited. This is solved in recent works [Whelan et al., 2014, Whelan et al., 2015] where a loop closure back-end is introduced. This back-end is able to correct potential artefacts in the 3D volume by enforcing the loop constraints not only on the camera trajectory but also on the dense 3D map using a deformation graph.

Following the paradigm of working on 3D models [Newcombe et al., 2011a], Stuckler and Behnke propose in [Stuckler and Behnke, 2012] and [Stückler and Behnke, 2014] converting the RGB and depth images into multiresolution surfel maps by using a voxel octree representation. Each surfel maintains a shape-texture descriptor, which guide data association between surfels in different maps during camera pose estimation. To alleviate the odometry drift they register the current frame with respect to the latest keyframe. A new keyframe is inserted when camera motion w.r.t. last keyframe is large enough. They also propose a loop closure technique where loop closure candidates are randomly sampled from a probability density function which positively weights the selection of spatially closer keyframes.

Kerl *et al.* [Kerl et al., 2013a] propose an RGB-D SLAM system where camera motion is

estimated with respect to keyframes, which are switched following an entropy-based criteria. They include of a simple but effective loop closure method based on keyframes spatial proximity to further refine the final odometry estimation. The final 3D is obtained by simply joining all the raw depth maps from the keyframes.

Contrary to volumetric-based approaches like KinectFusion, Meilland and Comport [Meilland and Comport, 2013a], integrate the depth map of each received frame in a set of close keyframes. To track every new frame they use the integrated keyframes to synthesise a reference keyframe from which the new frame is tracked. Though not reporting the use of any loop closure technique their method shows a compelling accuracy in terms of Absolute Trajectory Error, keeping a low trajectory drift.

In this work we base our RGB-ID SLAM approach on a representation of the environment in keyframes rather than by 3D volumes. We find that a keyframe representation of a 3D model facilitates the process of loop closure by state-of-the-art methods. We use this keyframe representation to reliably perform a structure based segmentation on the image domain of smooth depth maps. This segmentation based purely on the structure allows then to rate elements of the scene by the entropy on their normals. This rating can be used later to generate different masks filtering potentially unstable elements on the scene to facilitate appearance based place recognition system.

However contrary to [Meilland and Comport, 2013a], though we use reference frames for camera tracking, given our successful previous results [Gutiérrez-Gómez et al., 2016], we keep the raw representations for the depth map instead of using integrated keyframes.

6.3. Models and functions

In our RGB-ID SLAM system we frequently use some models and functions which are summarised in this section.

6.3.1. Projection model

A world point $\mathbf{X}$ is projected in the image point $\mathbf{p}$ by:

$$\mathbf{p} = \pi\left(\mathbf{X}\right) = \mathbf{K}\frac{\mathbf{X}}{\mathbf{e}_\mathbf{z}^T\mathbf{X}} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \frac{\mathbf{X}}{\mathbf{e}_\mathbf{z}^T\mathbf{X}}\ , \tag{6.1}$$

where $\mathbf{K}$ is the conventional calibration matrix, including the camera intrinsic parameters.

Inverse depth measurements $\mathcal{W}(\mathbf{p}) = \frac{1}{\mathbf{e}_\mathbf{z}^T\mathbf{X}}$ allow to lift 2D points from the image to 3D coordinates by the inverse projection function:

$$\mathbf{X} = \pi^{-1}(\mathbf{p}, \mathcal{W}(\mathbf{p})) = \frac{1}{\mathcal{W}(\mathbf{p})}\mathbf{K}^{-1}\mathbf{p}. \tag{6.2}$$

6.3.2. Photometric and geometric constraints

Let us denote two camera frames as A and B, at instants t and $t + \Delta t$ respectively. Let us denote the 3D camera motion between A and B by the rotation and translation pair $\left({}_B\mathbf{R}^A, \mathbf{r}_B^A\right) \in \mathbb{SE}(3)$. Assuming that the scene is static, given the intensity images $\mathcal{I}_A$ and $\mathcal{I}_B$, and inverse depth maps $\mathcal{W}_A$ and $\mathcal{W}_B$ defined over the image domain $\Omega \subset \mathbb{P}^2$, for an image point $\mathbf{p} = (u\ v\ 1)^T \in \Omega$ in frame A, the following constraints hold:

$$\mathcal{I}_B\left(\pi\left({}_B\mathbf{R}^A\pi^{-1}(\mathbf{p},\mathcal{W}_A(\mathbf{p}))+\mathbf{r}_B^A\right)\right)=\mathcal{I}_A(\mathbf{p}) \tag{6.3}$$

$$\mathcal{W}_B\left(\pi\left({}_B\mathbf{R}^A\pi^{-1}(\mathbf{p},\mathcal{W}_A(\mathbf{p}))+\mathbf{r}_B^A\right)\right)=\frac{1}{\mathbf{e}_\mathbf{z}^T\left({}_B\mathbf{R}^A\pi^{-1}(\mathbf{p},\mathcal{W}_A(\mathbf{p}))+\mathbf{r}_B^A\right)}, \tag{6.4}$$

where $\mathbf{e}_\mathbf{z}^T = (0\ \ 0\ \ 1)$. The constraint in intensity assumes constant illumination of one scene point. The second constraint is the measurement model of the depth sensor at frame B.

6.3.3. Reverse warping

Given two frames A and B we want to warp the intensity and inverse depth maps of B towards A to generate a new warped frame $B*$. Warping is performed for every pixel in the new generated frame following these steps

Given a pixel $\mathbf{p}$ in the destination warped image, the corresponding pixel $\mathbf{p}_w$ in the source image is obtained as:

$$\mathbf{X}_w = {}_B\mathbf{R}^A\pi^{-1}(\mathbf{p},\mathcal{W}_A(\mathbf{p}))+\mathbf{r}_B^A, \tag{6.5}$$

$$\mathbf{p}_w = \pi\left(\mathbf{X}_w\right) \tag{6.6}$$

By using the intensity constraint (6.3) the generation of the new intensity map is straightforward:

$$\mathcal{I}_{B*}(\mathbf{p}) = \mathcal{I}_B(\mathbf{p}_w) \tag{6.7}$$

To obtain the new inverse depth maps we need to perform some algebraic manipulation. From the inverse depth constraint (6.4) we have:

$$\mathcal{W}_B\left(\mathbf{p}_w\right) = \frac{1}{\mathbf{e}_\mathbf{z}^T\left({}_B\mathbf{R}^A\frac{1}{\mathcal{W}_{B*}(\mathbf{p})}\mathbf{K}^{-1}\mathbf{p}+\mathbf{r}_B^A\right)} \tag{6.8}$$

and solving for $\mathcal{W}_{\mathbf{B}*}(\mathbf{p})$:

$$\mathcal{W}_{B*}\left(\mathbf{p}\right) = \frac{\mathbf{e}_\mathbf{z}^T\,{}_B\mathbf{R}^A\mathbf{K}^{-1}\mathbf{p}}{1-\mathcal{W}_B\left(\mathbf{p}_w\right)\mathbf{e}_\mathbf{z}^T\mathbf{r}_B^A}\mathcal{W}_B\left(\mathbf{p}_w\right) \tag{6.9}$$

During the warping process, the projections $\mathbf{p}_w$ on the source image are usually obtained on float locations. Since intensity and inverse depth maps are represented over a discrete pixel grid we need to interpolate to compute the new intensity and inverse depth values. In a CUDA capable NVIDIA GPU interpolation in a grid is quite efficiently performed using *texture memory*. For the intensity we use a bilinear interpolation with the 4 neighbouring pixels. For the inverse depth values, in turn, we use a more simple nearest neighbour interpolation. The reasons for this decision are two fold. First, a pixel falling on an edge might yield an unrealistic intermediate value if linearly interpolating with 4 distinct inverse depth values. Secondly, given that depth maps from depth sensor show some areas without depth values, the more pixels are used for interpolation, the more chances we have of picking a hole which spoils the inverse depth estimate for the warped frame.

6.3.4. Dense frame covisibility ratio

Computing the covisibility score between two given frames is a key element to prune redundant frames, selection of reference frames or keyframes, or computing visibility graphs. In this work, taking advantage of the availability of dense depth maps, we use a covisibility computed from all the pixels in the image. Given two frames A and B and the camera motion estimate between them $({}_B\mathbf{R}^A, \mathbf{r}_B^A) \in \mathbb{SE}(3)$, we transfer pixels from A to B using (6.5) and (6.6). Then if the following two conditions are met:

- $\mathbf{p}_w \in \Omega$
- $\left|\mathcal{W}_B(\mathbf{p}_w) - \frac{1}{\mathbf{e_z}^T \mathbf{X}_w}\right| < 3\sigma_{\mathcal{W}}$

a pixel is tagged as visible. First condition rejects points out of the image domain, while the second condition rejects pixels which became occluded, by checking that their inverse depth values are in the uncertainty range $\sigma_{\mathcal{W}}$ allowed by the sensor noise model. After this test we can compute the covisibility ratio:

$$vis\_ratio_{A\to B} = \frac{\#visible\_pixels_{A\to B}}{\#nohole\_pixels_{A\to B}}. \tag{6.10}$$

This procedure is repeated switching the role of A and B, and then we select the minimum ratio as the final covisibility ratio.

6.3.5. Dense frame alignment

Dense frame alignment or registration consists into estimating the camera motion ${}_A\hat{\mathbf{T}}^B = ({}_A\hat{\mathbf{R}}^B, \hat{\mathbf{r}}_A^B)$ between two frames A and B by pixel-wise minimisation of the photometric and inverse depth residuals $r_{\mathcal{I}}(\mathbf{p})$ and $r_{\mathcal{W}}(\mathbf{p})$. This motion is computed iteratively in a coarse-to-fine manner using image pyramids, performing a given number of iterations at each pyramid level. At the start and every time we step down in the image pyramid, we downsample $\{\mathcal{I}_A,\ \mathcal{W}_A\}$ and compute the gradients $\{\nabla\mathcal{I}_A,\ \nabla\mathcal{W}_A\}$ at current pyramid level.

The initial estimate of the rigid motion is set to the identity, ${}_A\hat{\mathbf{T}}^B_{(0)} = \mathbf{I}$, unless an initial guess is provided.

At each iteration γ, first intensity and depth maps in frame B, $\{\mathcal{I}_B,\ \mathcal{W}_B\}$, are warped towards frame A taking the up-to-date motion estimate ${}_A\hat{\mathbf{T}}^B_{(\gamma)}$. By using the reverse warping approach described in Section 6.3.3 we obtain the warped images $\{\mathcal{I}_{B^{(\gamma)}},\ \mathcal{W}_{B^{(\gamma)}}\}$. These images are downsampled to the pyramid level at which current iteration is taking place.

The next step is the update of the motion estimate between frames. This is done by minimising the following cost function:

$$\hat{\boldsymbol{\xi}}^A_{B^{(\gamma)}} = \underset{\boldsymbol{\xi}}{\operatorname{argmin}} \sum_{\mathbf{p}\in\Omega} \rho\left(\frac{r_{\mathcal{I}}(\mathbf{p},\boldsymbol{\xi})}{\sigma_{r_{\mathcal{I}}}}\right) + \rho\left(\frac{r_{\mathcal{W}}(\mathbf{p},\boldsymbol{\xi})}{\sigma_{r_{\mathcal{W}}}}\right), \tag{6.11}$$

with

$$r_{\mathcal{I}}(\mathbf{p},\boldsymbol{\xi}) = \mathcal{W}_A(\mathbf{p})\nabla\mathcal{I}_A(\mathbf{p})(\mathbf{K}-\mathbf{p}\mathbf{e}_\mathbf{z}^T)\begin{bmatrix}\mathbf{I} & -[\pi^{-1}(\mathbf{p})]_\times\end{bmatrix}\boldsymbol{\xi} + \mathcal{I}_{B^{(\gamma)}}(\mathbf{p}) - \mathcal{I}_A(\mathbf{p}) \tag{6.12}$$

$$r_{\mathcal{W}}(\mathbf{p},\boldsymbol{\xi}) = \mathcal{W}_A(\mathbf{p})\big(\nabla\mathcal{W}_A(\mathbf{p})(\mathbf{K}-\mathbf{p}\mathbf{e}_\mathbf{z}^T)+\mathcal{W}_A(\mathbf{p})\mathbf{e}_\mathbf{z}^T\big)\begin{bmatrix}\mathbf{I} & -[\pi^{-1}(\mathbf{p})]_\times\end{bmatrix}\boldsymbol{\xi} + \mathcal{W}_{B^{(\gamma)}}(\mathbf{p}) - \mathcal{W}_A(\mathbf{p}), \tag{6.13}$$

The expressions for the residuals are the result of the linearisation of the constraints in (6.3) and (6.4) assuming a small motion described by the vector $\hat{\boldsymbol{\xi}}^A_{B^{(\gamma)}} = \left(\hat{\mathbf{r}}^A_{B^{(\gamma)}}, \hat{\boldsymbol{\theta}}^A_{B^{(\gamma)}}\right)$, where $\hat{\boldsymbol{\theta}}^A_{B^{(\gamma)}}$ is the axis-angle representation of the rotation given by rotation matrix ${}_{B^{(\gamma)}}\hat{\mathbf{R}}^A$. $\rho(x)$ is a generic cost function which must be symmetric, definite positive and $\rho(0) = 0$. $\sigma_{r_\mathcal{I}}$ and $\sigma_{r_\mathcal{W}}$ are scaling parameters which capture the uncertainty in intensity and inverse depth residuals, and allow for normalisation of residuals in different magnitudes. The choice $\rho(x) = \frac{x^2}{2}$ results in standard least-squares linear optimisation. Nevertheless to gain robustness against outliers, *e.g.*, pixels belonging to non-static elements, robust M-estimators are usually employed. Optimisation with robust cost functions is addressed by the Iteratively Reweighted Least Squares algorithm (IRLS) [Holland and Welsch, 1977], which results in a linear least-squares problem to be solved at each iteration:

$$\hat{\boldsymbol{\xi}}^A_{B^{(\gamma)}} = \underset{\boldsymbol{\xi}}{\operatorname{argmin}} \sum_{\mathbf{p}\in\Omega} \omega\left(\frac{r_\mathcal{I}(\mathbf{p},\mathbf{0})}{\sigma_{r_\mathcal{I}}}\right)\frac{r^2_\mathcal{I}(\mathbf{p},\boldsymbol{\xi})}{\sigma^2_{r_\mathcal{I}}} + \omega\left(\frac{r_\mathcal{W}(\mathbf{p},\mathbf{0})}{\sigma_{r_\mathcal{W}}}\right)\frac{r^2_\mathcal{W}(\mathbf{p},\boldsymbol{\xi})}{\sigma^2_{r_\mathcal{W}}}, \tag{6.14}$$

where the weighting function $\omega(x)$ depends on the used M-estimator. In this case we use the Student M-estimator (see Appendix B) for details.

The scaling parameters can be either provided as constants or automatically obtained at each iteration by their Maximum Likelihood (ML) estimators $\sigma^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}$ given a choice of the robust cost function. To compute these estimators we have to solve the following non-linear optimisation problem:

$$\left[\mu^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}, \sigma^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}\right] = \underset{\mu,\ \sigma}{\operatorname{argmin}} \sum_{\mathbf{p}\in\Omega} \left(\log\sigma + \rho\left(\frac{r_{\{\mathcal{I},\mathcal{W}\}}(\mathbf{p},\mathbf{0}) - \mu}{\sigma}\right)\right), \tag{6.15}$$

Taking $\rho(x) = \frac{x^2}{2}$, the problem is linear. Thus, in the first iteration we compute the initial seed with this function and then switch to the Student loss function. Though not explicitly required, location parameters $\mu^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}$ are also calculated since the scaling parameters depend on their estimate.

When using the estimator based on the Student's t-distribution it is neccesary to adjust the degrees of freedom of the distribution. A typical choice is to set it to $\nu = 5$ which yields approximately an asymptotic relative efficiency (ARE) of 95% at the Gaussian distribution. However, using the t-distribution for robust minimisation gives us the option of estimating ν as we do with the location and scaling parameters. By maximising the log-likelihood of the Student distribution with respect to ν [Liu and Rubin, 1995] we obtain the following non-linear equation:

$$\sum_{\mathbf{p}\in\Omega}\left(-\phi\left(\frac{\nu}{2}\right) + \ln\left(\frac{\nu}{2}\right) + \phi\left(\frac{\nu+1}{2}\right) - \ln\left(\frac{\nu+1}{2}\right) + 1 + \ln(\omega(x(\mathbf{p}),\nu)) - \omega(x(\mathbf{p}),\nu)\right) = 0, \tag{6.16}$$

where $\phi(y)$ denotes the digamma function. This equation is solved by the bisection method, assuming $\nu \in [2, 10]$, both for the intensity and inverse depth residuals yielding $\nu_\mathcal{I}$ and $\nu_\mathcal{W}$, with a lower ν implying more robustness to outliers. Ideally robustness parameters should be estimated alternatively with location and scaling following a Expectation-Maximisation scheme. However to keep computational cost low we estimate first the final location and scaling values for intensity and depth residuals fixing $\nu_\mathcal{I} = \nu_\mathcal{W} = 5$, and then we solve (6.16) with the computed values.

In dense RGB-D tracking the main source of outliers are the high depth residuals caused by occlusions which are due to dynamic objects or areas which simply become revealed or hidden by the camera motion. Due to this fact in scenes where many occlusions occur may yield lower values of $\nu_{\mathcal{W}}$, while for scenes with no occlusions, *e.g.*, planar scenes, $\nu_{\mathcal{W}}$ will be typically higher. For this reason and because too much robustness can lead to a bad or slow convergence to the solution, we correct the robustness parameter of the intensity residuals by taking:

$$\nu_{\mathcal{I}} = \max(\nu_{\mathcal{I}}, \nu_{\mathcal{W}}). \tag{6.17}$$

Naive computation of the scaling and robustness parameters would use all the residuals. Given that every residual in intensity and inverse depth corresponds to a single pixel, the number of samples would be extremely large, more than 300000 samples at the highest resolution level, which would involve a high computational cost. To reduce this burden we compute the scaling parameters in a sample with a maximum size of $N = 19200$ points obtained by systematic selection. This sample size guarantees first an integer stride for pixel sampling at all the considered resolutions; and secondly, after leading an statistical analysis, a relative precision of 0.03 with a confidence level of 99.7%.

Finally, after computing the scaling and robustness parameters, minimisation of (6.14) leads us to solve the following linear system:

$$\mathbf{H}^{(\gamma)} \hat{\boldsymbol{\xi}}^{A}_{B^{(\gamma)}} = \mathbf{b}^{(\gamma)}, \tag{6.18}$$

and after computing $\boldsymbol{\xi}^{A}_{B^{(\gamma)}}$ we update the interframe motion estimate for the next iteration:

$${}_{A}\hat{\mathbf{T}}^{B}_{(\gamma+1)} = \begin{pmatrix} \exp([\hat{\boldsymbol{\theta}}^{A}_{B^{(\gamma)}}]_{\times}) & \mathbf{r}^{A}_{B^{(\gamma)}} \\ 0 & 1 \end{pmatrix}^{-1} {}_{A}\hat{\mathbf{T}}^{B}_{(\gamma)}. \tag{6.19}$$

The covariance of the motion estimate is computed as the inverse of the Hessian. The Hessian depends on the intensity and depth maps gradients computed in the frame A. Due to sensor noise it is possible that some pixels provide misleading information making the Hessian larger, which would yield a wrongly optimistic covariance matrix. In order to prevent this, we estimate the Hessian by performing an extra iteration where the motion estimate is not updated and where $\{\mathcal{I}_A,\ \mathcal{W}_A\}$ are applied a bilateral filter prior to the computation of their gradients.

$$\boldsymbol{\Sigma}^{B\to A}_{B} = \mathbf{H}^{-1}_{filt} \tag{6.20}$$

We want the covariance referenced in the frame A. So we apply a change of coordinates:

$$\boldsymbol{\Sigma}^{A\to B}_{A} = \begin{pmatrix} -{}_{A}\mathbf{R}^{B} & \mathbf{0} \\ \mathbf{0} & -{}_{A}\mathbf{R}^{B} \end{pmatrix} \boldsymbol{\Sigma}^{B\to A}_{B} \begin{pmatrix} -{}_{A}\mathbf{R}^{B} & \mathbf{0} \\ \mathbf{0} & -{}_{A}\mathbf{R}^{B} \end{pmatrix}^{T} \tag{6.21}$$

6.4. RGB-ID SLAM system

Our RGB-ID SLAM system is implemented in two CPU threads running concurrently. One thread executes a front-end for every incoming frame from the RGB-D sensor, performing camera tracking and fusion of inverse depth measurements in a single keyframe. For both of these tasks the CPU thread calls functions which execute in GPU where pixel-wise operations can be easily

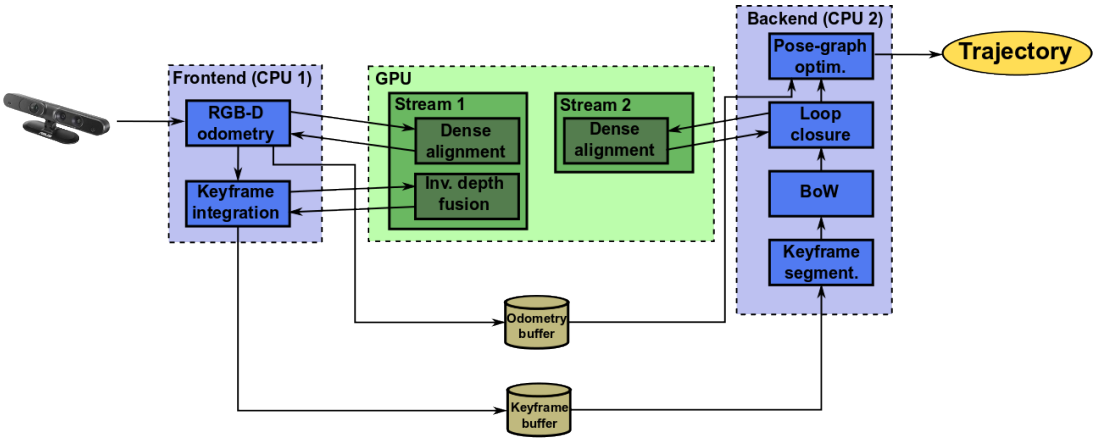


Figure 6.1: Scheme of our complete RGB-ID SLAM system

parallelised. The second thread executes a back-end in charge of managing the keyframes passed by the front-end through a buffer. First it performs a segmentation of the keyframe in juts using the map of normals. Secondly for every keyframe it obtains different BoW histograms and stores the keyframe in a database. Finally it attempts to close loops comparing the last processed keyframe against the keyframes in the database. All the tasks of the back-end thread are performed in CPU except for a dense alignment in GPU between keyframes when a successful loop is detected. In order to allow for simultaneous access to GPU by both threads without blocking, each thread calls its GPU functions on its own CUDA stream which is executed asynchronously.

6.4.1. Camera tracking

For every upcoming frame $\{\mathcal{I}_k,\ \mathcal{W}_k\}$ we have to compute the rigid body transformation ${}_{rf}\hat{\mathbf{T}}^k$ which best aligns it with a given reference frame $\{\mathcal{I}_{rf},\ \mathcal{W}_{rf}\}$. This motion is computed by the dense RGB-D alignment method described in Section 6.3.5. We provide an initial guess of the motion estimate using a constant velocity model, *i.e.*, ${}_{rf}\hat{\mathbf{T}}^k_{(0)} = {}_{rf}\hat{\mathbf{T}}^{k-1}\,{}_{k-2}\hat{\mathbf{T}}^{k-1}$.

Once we compute the motion estimate between the reference and current frame ${}_{rf}\hat{\mathbf{T}}^k$ and its covariance $\mathbf{\Sigma}^{rf\to k}_{rf}$, we compute the sequential odometry constraint as it will be required in the latter pose-graph optimisation step.

$$ {}_{k-1}\mathbf{T}^k = \left({}_{ref}\mathbf{T}^{k-1}\right)^{-1}\,{}_{ref}\mathbf{T}^k \tag{6.22} $$

$$ \mathbf{\Sigma}^{k-1\to k}_{k-1} = \begin{pmatrix} -{}_{rf}\mathbf{R}^{k-1} & \mathbf{0} \\ \mathbf{0} & -{}_{rf}\mathbf{R}^{k-1} \end{pmatrix}^T \left(\mathbf{\Sigma}^{rf\to k}_{rf} + \mathbf{\Sigma}^{rf\to k-1}_{rf}\right) \begin{pmatrix} -{}_{rf}\mathbf{R}^{k-1} & \mathbf{0} \\ \mathbf{0} & -{}_{rf}\mathbf{R}^{k-1} \end{pmatrix} \tag{6.23} $$

The naive approach to set the reference frame would be just taking the frame of the previous time step $k-1$. However the odometry estimate would be likely to show a high drift specially in sequences where the camera moves so slowly that interframe motion is below pixel accuracy.

Instead, at each time step we check if a new reference frame has to be taken by computing the covisibility ratio between the reference frame and current frame (see Section 6.3.4). If this ratio is below a threshold we reset the reference frame to current frame. We found that a threshold of 0.9 provides a good compromise between the requirements of taking spatially close frames to

perform a dense alignment, and at the same time showing a motion large enough to be perceived in the pixel-wise photometric and geometric residuals.

6.4.2. Keyframe fusion

After the visual odometry, the inverse depth map of current frame is fused in the last selected keyframe. The criteria for keyframe selection is the same covisibility criteria as used for the reference frames taken for odometry. We must note that generally keyframes do not have to coincide with the odometry reference frames, since we allow for using a different covisibility threshold. Normally it will be set to a lower value for keyframe switching, *i.e.*, we will initialise new keyframes at a lower rate than reference frames, using a visibility threshold of 0.7. Once a keyframe is initialised, the inverse depth maps of the incoming frames are fused with the keyframe's. We not only fuse the incoming frames but also the frames recorded prior to the keyframe selection in order to ensure that parts of the image no longer observed after keyframe initialisation. To do so, frames are stored temporally in a buffer and erased once integrated in the later keyframe. Every time we integrate an incoming frame, we integrate also the frame closest in time of the ones remaining in the buffer.

Fusion in keyframes is performed in GPU using the reverse warping approach detailed in section 6.3.3, where the frame to be integrated is warped towards the keyframe. Note that since we are using a reverse warping scheme parts of the scene with missing depth measurements in the frame from which the keyframe is initialised cannot be reconstructed from subsequent frames. We find this a minor disadvantage given first, that reverse warping in CUDA is computationally cheap and easy to implement and secondly, that the generation of many keyframes generally guarantees that zones missing in one keyframe can be observed in another one.

Once the frame has been warped we have to integrate every new inverse depth value for every pixel in the keyframe. To do this we first verify that the pixels on both the keyframe and the warped frame correspond to the same scene point by performing the same covisibility check as in Section 6.3.4. If a pixel passes this test we update its inverse depth value in the keyframe as follows:

$$\mathcal{W}_{kf}(\mathbf{p}) \leftarrow \frac{\mathcal{W}_{kf}(\mathbf{p})\mathcal{C}_{kf}(\mathbf{p}) + \mathcal{W}_{k}(\mathbf{p}_w)}{\mathcal{C}_{kf}(\mathbf{p}) + 1} \tag{6.24}$$

$$\mathcal{C}_{kf}(\mathbf{p}) \leftarrow \mathcal{C}_{kf}(\mathbf{p}) + 1 \tag{6.25}$$

where is a weight map $\mathcal{C}_{kf}$ computed pixel-wise and set to 1 upon a new keyframe initialisation. When a new keyframe is initialised, normals are computed in the last keyframe, which is then stored in a buffer, awaiting to be processed by the back-end in the second CPU thread.

6.4.3. Superjuts and entropy of normals

For every keyframe which is passed to the back-end we generate a normal map from its inverse depth map. Then we perform an object-like segmentation from the point cloud generated by a keyframe on the map of normals as in the methods proposed by [Triebel et al., 2010] and [Karpathy et al., 2013], which are based in the segmentation algorithm for RGB images developed by Felzenszwalb and Huttenlocher [Felzenszwalb and Huttenlocher, 2004].

A graph $G = (V, E)$ is constructed, where each vertex in V represents a 3D point $\mathbf{X}_i$ lifted from a pixel location and the edges E represent the neighbouring relations between points. Edges

are inserted between points with adjacent pixel locations on the image. Having the normals at every point, a weight w_{ij} for the edge joining vertices i and j is computed:

$$w_{ij} = \begin{cases} (1 - \mathbf{n}_i^T \mathbf{n}_j)^2 & \text{if } \mathbf{n}_j^T(\mathbf{X}_j - \mathbf{X}_i) > 0 \\ 1 - \mathbf{n}_i^T \mathbf{n}_j & \text{if } \mathbf{n}_j^T(\mathbf{X}_j - \mathbf{X}_i) \leq 0 \end{cases} \tag{6.26}$$

where the squared weight is applied to convex edges, reflecting the fact that convex regions usually contain points belonging to the same object and concave regions are likely to arise in frontiers between objects. After computing the weights, the segmentation algorithm is run and essentially groups points sharing edges with low weights in the same superjut. A parameter k must be tuned such that the higher k, the larger segments will be obtained. We set this value always to 0.6.

After the segmentation we compute a histogram of normals $F(\mathbf{n})$. Each of the M bins in the histogram corresponds to a Voronoi cell associated to one point out of M points uniformly distributed over the sphere. Since there is no analytical solution to the problem of evenly distributing M points on the sphere for any M, we use a simple approximate solution based on the golden section spiral [Boucher, 2006], which results in a discretisation with bins covering areas of similar size. Finally given the histogram of normals of a jut, we can compute its entropy:

$$ent(F) = \sum_{i=1}^{M} F(\mathbf{n}_i) \log(F(\mathbf{n}_i)). \tag{6.27}$$

6.4.4. Keypoint extraction and BoW histograms

Loop detection primarily relies on the DBoW2 place recognition system based in Bags of Binary Words proposed by Gálvez-López and Tardós [Gálvez-López and Tardos, 2012]. Mur-Artal et al. [Mur-Artal and Tardós, 2014] used this approach with ORB descriptors obtaining a quite reliable and fast loop closing system. As the authors suggest, we compute a pyramid of 8 levels and a relative scale of 1.2 between levels from the intensity image, and then we extract FAST keypoints at each pyramid level. In order to distribute the keypoints uniformly over the image we divide the image in cells and set the firing threshold for the FAST extractor adaptively to extract the desired number of points at each cell. Once the keypoints are extracted, they are lifted to 3D points using the inverse depth map of the keyframe and ORB descriptors are computed in the intensity image.

For the classification of the ORB descriptors in a BoW we use the same dictionary as in [Mur-Artal and Tardós, 2014]. For each keyframe a BoW histogram is computed with the frequency of each word in the given image. In order to gain robustness against long-term changes in the scene taking place more probably on zones of high entropy, different BoW histograms can be generated for the same keyframe by masking the keyframe for different entropy thresholds. Finally the processed keyframe is stored in a keyframe database from which keyframes are queried in the loop closure proccess.

6.4.5. Loop closure

After extracting keypoints and computing the BoW histogram the keyframe database is searched for potential keyframe candidates for loop closure. Given two keyframes, DBoW2 returns a similarity score based on the distance between their BoW histograms, which we

normalise with the score between the currently querying keyframe and the previous keyframe. If this score is above a threshold for any of the histogram comparisons a potential loop candidate is stored. In order to ensure that potential loop closures are obtained between temporally distant keyframes we establish a minimum keyframe separation.

For every possible keyframe candidate we match its 3D points with the 3D points of current keyframe using their ORB descriptors. Then we perform a geometric validation of the detected loop by computing a rigid motion estimate applying the method described in [Horn et al., 1988] for alignment of 3D point clouds in a 3-point RANSAC scheme. If more than 10 points agree with the computed motion between keyframes, and the convex hull spanned by these points takes up more than 5% of the image, the loop is tagged as correct. In order to compute the loop constraint, containing both a motion estimate and its covariance, we align both keyframes using the approach described Sec. 6.3.5 where we pass the RANSAC rigid motion estimate as initial guess.

Pose-graph optimisation

When a new loop is detected it is added to a cluster containing loop constraints which are temporally close, allowing a maximum separation of 10 keyframes between any two of the keyframes in the cluster. Pose-graph optimisation is applied when one of the clusters is no longer accepting new loop constraints, updating the graph with the new constraints in the cluster, as well as the poses and odometry constraints computed by the front-end, since the last graph update. Then, given a graph with a set of odometry constraints, $\mathcal{S} = \{(1,2), ..., (N-1, N)\}$, and a set of loop constraints, $\mathcal{R} = \{(i_1, j_1), ..., (i_L, j_L)\}$, we want to find the trajectory $\mathbf{x} = \{{}_W\mathbf{T}^1,\ {}_W\mathbf{T}^2,\ ...,\ {}_W\mathbf{T}^N\}$ which minimises the following cost function:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{(i,j)\in\mathcal{S}} \mathbf{e}_{ij}^T(\mathbf{x})\boldsymbol{\Sigma}_i^{i\to j}\mathbf{e}_{ij}(\mathbf{x}) + \sum_{(i,j)\in\mathcal{R}} \mathbf{e}_{ij}^T(\mathbf{x})\boldsymbol{\Sigma}_i^{i\to j}\mathbf{e}_{ij}(\mathbf{x}), \tag{6.28}$$

where the error term $\mathbf{e}_{ij}(\mathbf{x})$ represents the residual between a constraint ${}_i\hat{\mathbf{T}}^j$ and the relative motion between the trajectory poses ${}_W\mathbf{T}^i$ and ${}_W\mathbf{T}^j$ it is connecting, expressed with a minimal parametrisation in the tangent space of the manifold of rigid body motions $\mathbb{SE}(3)$:

$$\mathbf{e}_{ij}(\mathbf{x}) = \log_{\mathbb{SE}(3)}\left({}_i\hat{\mathbf{T}}^j({}_W\mathbf{T}^j)^{-1}\,{}_W\mathbf{T}^i\right) \tag{6.29}$$

In order to speed-up the process we perform the optimisation in a multi-layered scheme. In the first layer, odometry constraints between consecutive frames, are substituted by constraints between consecutive keyframes, which are also computed by the tracking front-end, and thus we only optimise the poses of the keyframes. In the second level the keyframe poses are fixed, and the poses of the rest of the frames are optimised by enforcing the odometry constraints.

6.5. Experiments

We first compare our method quantitatively with the state-of-the-art in terms of trajectory estimation. Then we perform a qualitative evaluation of the 3D reconstructions both in the Freiburg and our own RGB-D sequences, and finally we measure the computational performance, both of the front-end, in charge of camera tracking, and the back-end, performing keyframe processing and loop closing.

Table 6.1: Absolute trajectory error (RMSE, median and max) in meters of our method with and without loop closure and comparison with state-of-the-art approaches. For a given error measure in a dataset, we show in bold the best approach in the state of the art and ours if it is the absolute best.

Approach	fr1/desk			fr1/desk2			fr1/room			fr2/desk			fr3/office			fr3/nst		
	RMSE	median	max	RMSE	median	max	RMSE	median	max	RMSE	median	max	RMSE	median	max	RMSE	median	max
Ours (w/o loop closure)	0.034	0.030	0.087	0.054	0.037	0.137	0.087	0.072	0.203	0.037	0.026	0.084	0.057	0.042	0.118	0.041	0.030	0.105
Ours (w/ loop closure)∗	**0.017**	**0.014**	**0.056**	**0.035**	**0.026**	**0.103**	**0.038**	**0.032**	**0.132**	0.019	**0.017**	**0.043**	0.027	**0.026**	**0.048**	0.018	0.016	0.062
Unified VP [Meilland and Comport, 2013b]	-	**0.018**	**0.066**	-	-	-	-	0.144	0.339	-	0.093	0.116	-	-	-	-	-	-
ICP+RGB-D [Whelan et al., 2013a]	-	0.069	0.234	-	-	-	-	0.158	0.421	-	0.119	0.362	-	-	-	-	-	-
6D RGB-D odometry [Dong et al., 2014]	-	-	-	-	-	-	0.095	**0.067**	0.254	0.197	0.174	0.416	-	-	-	-	-	-
SDF tracking [Bylow et al., 2013]	0.035	-	-	0.062	-	-	0.078	-	-	-	-	-	0.040	-	-	-	-	-
DIFODO [Jaimez and González-Jiménez, 2015]	0.047	-	-	0.094	-	-	0.109	-	-	0.342	-	-	-	-	-	-	-	-
ElasticFusion (w/o loop closure) [Whelan et al., 2015]	0.022	-	-	-	-	-	-	-	-	-	-	-	0.025	-	-	0.027	-	-
RGB-D SLAM [Endres et al., 2014]∗	0.023	-	-	**0.043**	-	-	0.084	-	-	0.057	-	-	0.032	-	-	0.017	-	-
MRSMap [Stuckler and Behnke, 2012]∗	0.043	-	-	0.049	-	-	0.069	-	-	0.052	-	-	0.042	-	-	2.018	-	-
DVO-SLAM [Kerl et al., 2013a]∗	0.021	-	-	0.046	-	-	**0.053**	-	-	**0.017**	-	-	0.035	-	-	0.018	-	-
Kintinious [Whelan et al., 2014]∗	0.037	0.031	0.078	0.071	-	-	0.075	0.068	**0.231**	0.034	**0.028**	**0.079**	0.030	-	-	0.031	-	-
ElasticFusion (w/ loop closure) [Whelan et al., 2015] ∗	**0.020**	-	-	-	-	-	-	-	-	-	-	-	**0.017**	-	-	**0.016**	-	-

∗ with loop closure and pose-graph optimisation (deformation graph for ElasticFusion)

6.5.1. Trajectory Estimation

We first compare our method to state-of-the-art visual odometry and SLAM approaches with RGB-D systems. For the comparison we use the TUM benchmarking dataset [Sturm et al., 2012]. The evaluation has been carried out taking the error metric of the Absolute Trajectory Error (ATE) in meters (see Table 4.5). Since we did some new improvements and modifications on the code, the performance of our approach without loop closure detection nor pose-graph optimisation slightly differs from the performance of the RGB-D odometry method presented in Chapter 4. However it is worth noticing that there exists a large improvement in the $fr2/desk$ dataset. Apparently the reason is a correction we applied to the depth maps of the Freiburg 2 sequences by a constant scaling factor. This correction was in principle reported as done by the authors of the dataset. However it seems from the observation made by Mur-Artal et al. in [Mur-Artal and Tardós, 2014] of a scale bias in the RGB-D SLAM system of Endres et al. [Endres et al., 2014], that it has not been applied in the Freiburg 2 datasets available for download.

The table shows that our method outperforms most of the odometry and SLAM methods in the literature and it is close to the ElasticFusion of Whelan et al. [Whelan et al., 2015], which shows the best performance in the state-of-the-art in the datasets $fr3/office$ and $fr3/nst$. Among the methods performing only odometry, ElasticFusion in tracking-only mode outperforms them and even some of the methods performing loop closure. However in other datasets our method shows overall the best performance both in tracking-only mode and with loop closures. It is worth noticing that challenging datasets like $fr1/desk2$ and $fr1/room$, showing a fast camera motion, our method provides a great precision.

6.5.2. 3D reconstruction

In this section we perform a qualitative evaluation of the 3D models reconstructed of our SLAM system. To build the 3D model of the scene we concatenate the point clouds of different keyframes in real-time. To avoid unnecessary redundant points we only concatenate partial point clouds which correspond to novel parts of the scene between consecutive keyframes. Redundant points due too loop closure are eliminated at the end of the execution of our method by performing a voxel grid filtering with a voxel size of 1 cm. In Fig. 6.2 we show the 3D models of the evaluated TUM datasets, showing a great detail of the reconstructions.

6.5.3. Computational performance

The computational performance of our system has been evaluated in a laptop PC with an Intel Core i7-4710HQ CPU at 2.50GHz, 16GB of RAM and a nVidia GeForce GTX 850M GPU with 4GB of memory. As shown in Table 6.2, the execution time of the front-end varies depending on whether the system operates in tracking only-mode, *i.e.*, with the back-end disabled or with its complete functionalities. This is due to the division of the GPU resources upon successful loop closure between the visual odometry and integration modules from the front-end, and the keyframe alignment module between loop keyframes from the back-end.

In Fig. 6.3 we show graphically the computational cost of the back-end pipeline, which has been dissected in the different processes involved. As we can observe there is a nearly constant cost for keyframe segmentation and descriptor extraction, while the cost for loop closure detection and pose-graph optimisation shows a great variability. The reason is that most of the cost of loop closure detection is governed by the step of geometric verification and loop constraint computation, which is run only on a reduced list of loop candidates delivered by

fr1/desk

fr1/desk2

fr1/room

fr2/desk

fr3/office

fr3/nst

Figure 6.2: Final 3D models from the TUM datasets on which our method has been evaluated. Despite the discontinuities in color, we can appreciate a good level of detail in the final reconstruction.

Table 6.2: Computational cost of the front-end pipeline in milliseconds

	fr1/desk	fr1/desk2	fr1/room	fr2/desk	fr3/office	fr3/nst
w/o loop closure	39	39	42	38	38	36
w/ loop closure	46	55	76	49	53	41

Table 6.3: Mean computational cost of the processes involved in the back-end pipeline in milliseconds. To achieve online performance the mean total cost should be lower than the mean keyframe rate.

	fr1/desk	fr1/desk2	fr1/room	fr2/desk	fr3/office	fr3/nst
Segmentation	147	156	152	151	174	203
ORB desc. + BoW hist.	18	19	18	17	18	19
Loop detection	54	61	52	46	33	36
Pose-graph optim.	1	1	1	1	1	1
Total keyframe proc.	243	260	246	237	250	286
Mean keyframe rate	320	222	246	1017	773	1735

the efficient appearance-based BoW algorithm for loop detection. Red dashed line represents the time it takes the front-end to deliver a new keyframe to the back-end, assuming that it is able to process al the frames in the sequence at the frame rate of 30 Hz. We can observe that in datasets where the camera is moved slowly in a loop around the same scene, the back-end processing time is widely below the average keyframe rate. However in datasets like $fr1/desk2$ or $fr1/room$, where the camera moves quickly and/or the area being mapped changes a lot, the back-end struggles to operate at keyframe rate. These observations are embodied numerically in Table 6.3.

6.6. Discussion

In this Chapter we have presented an initial version of a complete RGB-ID SLAM system which combines an accurate dense and direct RGB-D odometry system and state-of-the-art loop closing procedure to achieve a great accuracy in the estimation of the trajectory. The algorithm also performs the integration of the tracked frames into keyframes with smooth inverse depth maps. The fusion of depth measurements in keyframes allows first to achieve a greater accuracy in the computation of relative camera transformation at loop closure, and secondly to perform an online segmentation of the keyframe based solely on the structure of the captured scene. The concatenation of the novel parts between consecutive keyframes allow to obtain precise 3D models of the environment. Future work to improve this system include exploring the possibilities of taking advantage of the segmentation method, for example, to perform robust loop closure and relocalisation under long-term changing environments or investigating more sophisticated ways of efficiently generating lightweight 3D models from the keyframes in real-time.

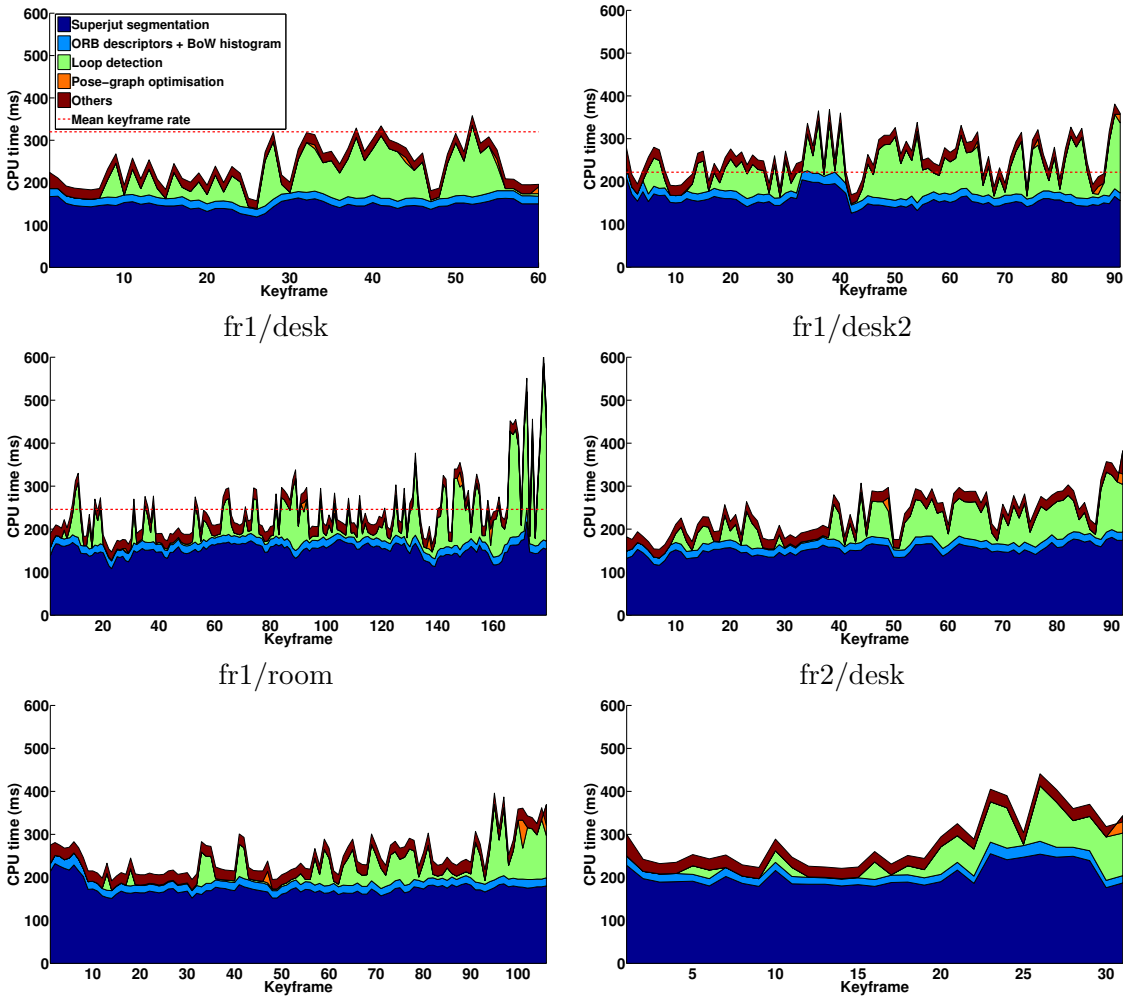


Figure 6.3: Computational cost of the keyframe processing in the back-end divided by processes (area graph), and average acquisition time of a keyframe (dashed red). In the datasets where the average acquisition time is not shown, it means that it is out of the plot bounds (above 600 ms). This usually occurs in datasets when the camera executes a loop around some scene, which normally involves that keyframes are switched with low frequency.

Chapter 7

Curve-graph odometry: Orientation-free error parameterisations for loop closure problems

During the incremental odometry estimation from mobile sensors, the accumulation of estimation error produces a drift in the trajectory, as it was shown to occur in previous chapters with visual and RGB-D SLAM. This drift becomes observable when returning to previously visited areas, where it is possible to correct it by applying loop closing techniques. Ultimately a loop closing process leads to an optimisation problem where new constraints between poses obtained from loop detection are applied to the initial incremental estimate of the trajectory. Typically this optimisation is jointly applied on the position and orientation of each pose of the robot using the state-of-the-art pose graph optimisation scheme on the manifold of the rigid body motions. In this chapter we propose to address the loop closure problem using only the positions and thus removing the orientations from the optimisation vector. The novelty in our approach is that, instead of treating trajectory as a set of poses, we look at it as a curve in its pure mathematical meaning. We define an observation function which computes the estimate of one constraint in a local reference frame using only the robot positions. Our proposed method is compared against state-of-the-art pose graph optimisation algorithms in 2 *and* 3 *dimensions. The main benefits of eliminating orientations are twofold. First, the objective function in the optimization does not mix translation and rotation terms, which may have different scales. Secondly, computational performance can be improved due to the reduction in the state dimension of the nodes of the graph.*

7.1. Introduction

The probabilistic nature of Simultaneous Localisation and Mapping (SLAM) techniques, and the incremental estimation, lead to an unavoidable error build-up. The accumulated error gives raise to a drift in the trajectory, which becomes evident when the sensor platform revisits a previous location. It is expressly in these situations when the so called loop closing techniques can be applied to correct the drift.

The loop closure process can be divided into three steps: loop detection, computation of the loop closing constraint and trajectory correction with the new constraints. The detection and the constraint computation techniques are sensor dependent and are managed by the front-end.

The last step, trajectory correction, is the one which this work is mainly focused on. Traditionally, the new loop constraints are enforced by defining a non-linear least squares optimisation problem where the final trajectory is the one which minimises the combined cost of violating the initial odometry constraints from the SLAM estimate and the new loop closure constraints.

Following the state-of-the-art pose graph formulation [Kümmerle et al., 2011], the loop closure optimisation problem is presented as a graph of nodes where each node represents one pose and the arcs represent the odometry and loop closure constraints. An odometric constraint encapsulates the incremental motion estimate between two consecutive poses $i-1$ and i in a reference frame attached to $i-1$, in such a way that an arbitrary spatial transformation applied to both poses should not modify the cost of violating this constraint. This applies similarly to a loop closure constraint between two poses, say, i and j, on the relative motion of pose j with respect to a reference frame attached to i or vice versa.

In a pose graph formulation, each pose includes a translation and a rotation and is represented as an element of the special Euclidean group, a manifold which describes the rigid body kinematics in 2 ($\mathbb{SE}(2)$) or 3 ($\mathbb{SE}(3)$) dimensions. In the context of optimisation, variables lying on manifolds different from the usual Euclidean space $\mathbb{R}^n$ are prone to violate the manifold constraints if no special care is taken. One typical approximation, thus subject to inaccuracies, is to impose these constraints after optimisation. However, for greater correctness and accuracy, smarter solutions impose the manifold topology directly during optimisation [Absil et al., 2007].

Another problematic specific to Euclidean Groups and pose-graph optimisation in SLAM is the inability to define an unambiguous metric [Park, 1995], which arises from the different magnitudes in which rotation and translation are measured. Since error functions in pose-graph SLAM combine rotation and translation, the set up of the scaling between rotation and translation can have a strong influence in the final result of the optimisation. The information matrix for a given constraint solves this ambiguity by normalising both magnitudes. However it is not rare the case where the exact information matrix for some constraints of the graph is not available or easy to compute.

We propose to reformulate the loop closure optimisation problem using a state representation which removes the orientation of the poses. The novelty in our approach is that, instead of treating trajectory as a set of poses, we look at it as a curve in its pure mathematical meaning. A curve is a mathematical entity defined in an Euclidean Space $\mathbb{R}^n$ and has a set of local properties like speed, curvature and torsion, which can be defined point wise and are invariant to arbitrary rigid transformations. This leads to the idea that proper odometry and loop closure constraints can be computed using only the positions and that these constraints are related with the local properties of a discrete curve.

Resulting from our proposal three main advantages arise:

- We avoid mixing translation and rotation magnitudes in the same optimised vector, avoiding heuristic scalings in the norm of the residuals when the information matrix is not available.

- The number of degrees of freedom per pose is reduced from 6 to 3 in the 3D case and from 3 to 2 in the 2D case. This leads to a dimensionality reduction of the optimisation problem which can involve a potential increase in computational performance.

- The optimisation could be performed directly in a Euclidean space, with no need for defining an error function and special operators for non-Euclidean spaces.

One issue of our this new representation of a pose-graph problem are the potential singularities when poses are redundant or are aligned in a straight. To tackle this, we first perform a graph reduction to prune redundant and aligned poses. The evaluation of our method is performed in a extensive dataset containing trajectories in 2 and 3 dimensions.

7.2. Related Work

Several related works address efficiency and convergence issues of the optimisation algorithms for pose graphs. However the discussion on different representations of the nodes of the graph is less prevalent and, to the best of our knowledge, all of them assume that the optimisation must be performed both in the orientation and the position of the poses.

Concerning the optimisation algorithms the main objective is to make it robust to local minima and lowering the computational needs. Standard approaches to solve non-linear least squares problems are based on the Gauss-Newton method, which consists in iteratively linearising the energy function around the current solution and solving a linear system until convergence. However this involves a large computational cost and, unless a good initial estimate is provided, it is likely to stuck on a local minima. Approaches based on this method like iSAM [Kaess et al., 2008] and g^{2}o [Kümmerle et al., 2011] tend to exploit the structure of the graph to reduce the computational cost.

Another family of approaches introduce a relaxation of the problem, i.e., at each iteration they compute an update of only a subset of the nodes. Although these updates are approximate, the robustness to local minima sticking is generally increased. In this sense Duckett et al. proposed the use of Gauss-Seidel relaxation [Duckett et al., 2002] and based on this, Frese et al. [Frese et al., 2005] introduced a multilevel relaxation (MLR). Olson et al. [Olson et al., 2006] propose a relaxation based approach using a Stochastic Gradient Descent algorithm (SGD) and an incremental pose parametrisation. Their results showed a dramatic reduction in computational cost compared with other existing approaches at that time. In [Grisetti et al., 2009], Grisetti et al. extend Olson's method including a novel tree parametrisation for the poses and extending to 3D poses. Grimes et al. [Grimes et al., 2010] propose to apply a stochastic relaxation while solving the linearised system around current estimate. In [Peasley and Birchfield, 2014], Peasley and Birchfield proposed first performing a SGD with relative pose parametrisation to get a quick initial solution close to the minimum and the switching to the Gauss-Seidel method to reach the minimum.

Martinez et al. [Martínez et al., 2010] and Carlone et al. [Carlone et al., 2011] proposed a linear approximation to compute a first suboptimal solution in 2D pose graphs without requiring an initial seed for the state of the nodes. This suboptimal solution could then be refined by non-linear optimisation approaches. Also for 2D graphs, Carlone and Censi [Carlone and Censi, 2014] proposed recently a method for orientation estimation which is more robust to local minima than state-of-the-art approaches. In [Dubbelman et al., 2010], Dubbelman et al. propose an efficient method which obtains a closed form solution for loop closure in trajectories where there is a single loop constraint. This work was further extended [Dubbelman and Browning, 2013] enabling it to close multiple loops on the same trajectory.

Recently Anderson et al. [Anderson et al., 2014] proposed a continuous-time SLAM approach, which addresses the pose-graph optimisation by parametrising the motion of the platform by continuous wavelet functions based on B-splines instead of conventional parametrisation with

discrete poses. This is beneficial when using high-rate sensors, multiple unsynchronized sensors, or scanning sensors, such as lidar and rolling-shutter cameras, during motion.

Concerning different descriptions of the states of the nodes, the proposal by Strasdat et al. [Strasdat et al., 2010] is specially convenient for pose graph optimisation from loop closure of visual odometry estimates obtained by some monocular front-end. By describing the poses and the constraints between them by similarity transforms instead of by rigid body motions, it allows scale drift correction in the optimisation back-end.

Another important issue in the scope of pose-graph optimisation which have been addressed in the literature is the robustness to false positives during the loop detection step. In this sense authors of [Kümmerle et al., 2011] propose several robust cost functions in their implementation to decrease the effect of outliers. More sophisticated methods for robust pose-graph optimisation include the introduction of switchable constraints [Sünderhauf and Protzel, 2012] which act as weights for the loop closure constraints, the RRR algorithm [Latif et al., 2013] based on consistency checks of topologically similar loop constraints, or the Max-Mixture Model [Olson and Agarwal, 2013] which accounts for the possibility of false loop closures using a high variance Gaussian distribution.

7.3. Standard Pose Graph Optimisation

Let $\mathbf{x} = (x_1 \cdots x_N)$ be the optimisation state vector which contains the configuration x_i of each node in the graph. Let $\hat{\mathbf{z}} = (\hat{z}_1 \cdots \hat{z}_M)$ be the vector containing all the constraints between the nodes in the graph. These constraints can not only represent edges joining pairs of nodes but also, more generally, cliques relating an undetermined number of nodes. Let also $f_k(\mathbf{x})$ be an observation function which computes the estimate of the constraint $\hat{z}_k$ given the current state $\mathbf{x}$ of the graph.

7.3.1. Node Parametrisation and Constraints in $\mathbb{SE}(n)$

When defining a pose-graph optimisation problem in the context of SLAM, nodes in the graph represent poses. One pose consists of a location in the space and an orientation and both can be jointly described in the manifold of rigid body motions in 2D ($\mathbb{SE}(2)$) or 3D ($\mathbb{SE}(3)$) by using transformation matrices. Then the configuration of a node x_i in the graph is parametrised as $x_i = {}_W\mathbf{T}^i$, i.e., the spatial transformation from the world frame W to the frame associated to pose i.

The observation function for a constraint between two poses is defined as:

$$f_{ij}(\mathbf{x}) \doteq f_k(\mathbf{x}) = \left({}_W\mathbf{T}^i\right)^{-1} {}_W\mathbf{T}^j. \tag{7.1}$$

The odometry constraints for the optimisation problem are computed by applying the observation function to the initial state $\hat{\mathbf{x}} = \{{}_W\hat{\mathbf{T}}^1,\ {}_W\hat{\mathbf{T}}^2,\ ...,\ {}_W\hat{\mathbf{T}}^N\}$, while the loop closure constraints are provided by a module in charge of both loop detection and computation of transforms $\boldsymbol{T_{LC}} = \{{}_{i_1}\hat{\mathbf{T}}^{j_1}_{LC},\ ...,\ {}_{i_L}\hat{\mathbf{T}}^{j_L}_{LC}\}$. That is:

$$\hat{z}_k = {}_i\hat{\mathbf{T}}^j = \begin{cases} \left({}_W\hat{\mathbf{T}}^i\right)^{-1} {}_W\hat{\mathbf{T}}^j & \text{if} \quad (i,j) \in \mathcal{S} \\ {}_i\hat{\mathbf{T}}^j_{LC} & \text{if} \quad (i,j) \in \mathcal{R} \end{cases}, \tag{7.2}$$

where $\mathcal{S} = \{(1,2), ..., (N-1, N)\}$ and $\mathcal{R} = \{(i_1, j_1), ..., (i_L, j_L)\}$ are respectively the sets of pose pairs sharing odometry and loop closure constraints.

7.3.2. Generalised Optimisation on Manifolds

The unambiguous description of elements lying on a manifold usually require more parameters than dimensions has the manifold. Such is the case of elements in $\mathbb{SE}(n)$, usually described by transformation matrices. For each additional parameter a constraint related with the manifold topology is established. To comply with these constraints during iterative optimisation, updates of the estimation must be computed in the tangent space of the manifold using a minimal parametrisation. For this purpose the operators $\boxminus$ and $\boxplus$ are used. Roughly speaking, the operator $\boxminus$ computes the difference between two transformations with a minimal parametrisation, while the operator $\boxplus$ applies a minimally parametrised perturbation to a rigid transformation (for a more detailed and rigorous explanation we refer the reader to [Hertzberg et al., 2013]).

Then, the error e_k resulting from violating the graph constraint between poses i and j is:

$$e_k(\mathbf{x}) = f_k(\mathbf{x}) \boxminus \hat{z}_k. \tag{7.3}$$

The uncertainty for a constraint $\hat{z}_k$ is represented by the information matrix Ω_k . Assuming that all the constraints are independent, the cost function for pose graph optimisation is:

$$\begin{aligned} \mathbf{x}^* &= \arg\min_{\mathbf{x}} \sum_{(i,j)\in\mathcal{S}} e_k^{\mathsf{T}}(\mathbf{x})\Omega_k e_k(\mathbf{x}) + \sum_{(i,j)\in\mathcal{R}} e_k^{\mathsf{T}}(\mathbf{x})\Omega_k e_k(\mathbf{x}) \\ &= \arg\min_{\mathbf{x}} \mathbf{e}^{\mathsf{T}}(\mathbf{x})\boldsymbol{\Omega}\mathbf{e}(\mathbf{x}), \end{aligned} \tag{7.4}$$

where $\mathbf{e} = (e_1 \cdots e_M)$ and $\boldsymbol{\Omega} = \operatorname{diag}(\Omega_1, ..., \Omega_M)$ and $M = (N-1) + L$ is the total number of constraints.

Given the initial guess $\breve{\mathbf{x}} = \hat{\mathbf{x}}$, (7.4) can be solved iteratively until convergence by computing a first order Taylor expansion of the error at each iteration :

$$\mathbf{e}(\breve{\mathbf{x}} \boxplus \boldsymbol{\delta}) = \mathbf{e}(\breve{\mathbf{x}}) + \left.\frac{\partial\left(\mathbf{e}\left(\breve{\mathbf{x}} \boxplus \boldsymbol{\delta}\right)\right)}{\partial \boldsymbol{\delta}}\right|_{\boldsymbol{\delta}=\mathbf{0}} \boldsymbol{\delta} = \breve{\mathbf{e}} + \mathbf{J}\boldsymbol{\delta}, \tag{7.5}$$

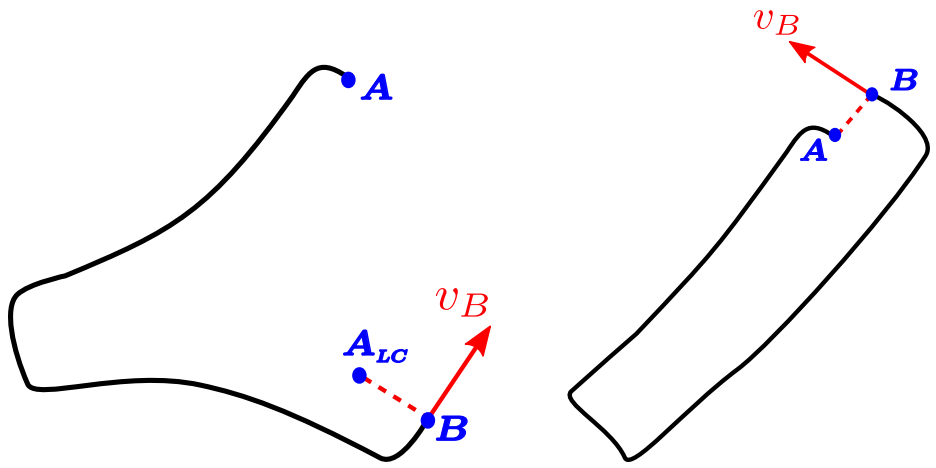
where $\boldsymbol{\delta} = (\delta_1 \cdots \delta_N)$ and abusing from notation $\breve{\mathbf{x}} \boxplus \boldsymbol{\delta} = \{\breve{x}_1 \boxplus \delta_1, ... , \breve{x}_N \boxplus \delta_N\}$. The Jacobian can be computed analytically (Chap. 2 in [Strasdat, 2012]).

Then, the resulting linear optimisation problem is solved to obtain the state incremental update $\boldsymbol{\delta}$ which is used to compute the state for next iteration:

$$\breve{\mathbf{x}} \leftarrow \breve{\mathbf{x}} \boxplus \boldsymbol{\delta}. \tag{7.6}$$

7.4. Our Approach: Graph Optimisation on $\mathbb{R}^n$

Instead of jointly estimating the position and orientation of the poses by carrying on an optimisation in the manifold of rigid body motions, we propose imposing the loop closure constraints by taking only the position part of the poses. The underlying idea behind this proposal is that a trajectory can be considered a discrete curve in the Euclidean space where new loop constraints between some points are imposed modifying as less as possible the local properties of the curve, which are encoded in the odometric constraints. This is intuitively shown in the example of Fig. 7.1. Given a trajectory where no information about the orientation is



(a) (b)

Figure 7.1: (a) Curve with an open loop where the point **A** should be at the same position as $\mathbf{A_{LC}}$ and keep a relative orientation w.r.t. the vector tangent to the trajectory at **B**. (b) Intuition of how this loop should be closed.

shown, one can perceive however how the trajectory has to be bended so that the point **A** is at the relative position w.r.t. **B** given by the loop closure constraint (dashed line) and the vector $\mathbf{v}_B$ tangent to the trajectory at **B**.

Moreover, the removal of the orientations out of the optimised variables seems reasonable, since the position and orientation of a body which freely moves in the space do not have to be coupled generally. In this sense, the inclusion of the orientation in the optimisation may respond primarily to the need of expressing the odometry and loop constraints in a local reference frame, such that that the error function is invariant to rigid body motions applied to the whole trajectory.

Thus the main challenge is to define appropriate constraints given only the set of positions composing the trajectory. These constraints must keep the error invariant to an arbitrary rigid motion applied to the curve.

7.4.1. Curves in 2D and 3D

Let us first introduce some notions about curves in 2 and 3 dimensions [Pressley, 2001]. These notions though not applied in the implementation of our approach help to provide a mathematical insight of the implications of our approach, showing in fact that our proposed optimisation problem is equivalent to bending a curve so that it passes through new points, while trying to keep its original shape.

Generally, a curve $\mathbf{r}$ can be defined as a mapping of a scalar t in a given interval $I = [a, b]$ onto the euclidean space $\mathbb{R}^n$, i.e., $\mathbf{r} : I \rightarrow \mathbb{R}^n$.

A n-dimensional curve is characterised by n properties defined locally at every point of the curve. For a 2D curve these properties are the metric derivative or speed $||\mathbf{r}'||$, and the curvature $\kappa(t)$ which is defined as:

$$\kappa(t) = \frac{\mathbf{r}'^{\mathsf{T}}(t)\mathbb{J}\mathbf{r}''(t)}{||\mathbf{r}'(t)||^3}, \tag{7.7}$$

with

$$\mathbb{J} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}. \tag{7.8}$$

For the 3D case we have to consider a new property of the curve: the torsion $\tau(t)$. The torsion of a curve is given by the variation of its osculating plane which can be defined as the plane which locally contains the curve in the vicinity of one of its points. Note that planar curves have no torsion since they are contained in the same plane at every point.

Then, for a 3D curve the curvature and torsion are defined by:

$$\kappa(t) = \frac{||\mathbf{r}'(t) \times \mathbf{r}''(t)||}{||\mathbf{r}'(t)||^3}, \tag{7.9}$$

$$\tau(t) = \frac{\mathbf{r}'''^{\mathsf{T}}(t)(\mathbf{r}'(t) \times \mathbf{r}''(t))}{||\mathbf{r}'(t) \times \mathbf{r}''(t)||^2}. \tag{7.10}$$

Note that these properties are invariant to rigid transformations applied to the curve both in the 2D and 3D cases.

7.4.2. Node Parametrisation and Constraints

Our approach parametrises each node in the graph as $x_i = \mathbf{r}^i_W = {}_W\mathbf{T}^i\,(1{:}3, 4)$. Analogously to Sec. 7.3 we define an observation function for the constraints:

$$f_{ij}(\mathbf{x}) \doteq f_k(\mathbf{x}) = f(\mathcal{F}_{\mathbb{R}^n}(\mathbf{r}^i_W), \mathbf{r}^j_W), \tag{7.11}$$

where $\mathcal{F}_{\mathbb{R}^n}(\mathbf{r}^i_W)$ is a function which extracts from the graph, the minimum number of poses backwards from $\mathbf{r}^i_W$ to define a reference frame for a given number of spatial dimensions n. Details on how this reference frame is defined for different dimensions will be provided in next sections.

To apply our method we need the relative position measurements $\boldsymbol{\Delta}\hat{\mathbf{r}}^{ij}_i$ between all the pairs of frames (i, j) sharing one constraint. This is done in two steps. The first step consists in computing the absolute pose measurement of frame j as the result of concatenating the constraint between i and j with the absolute pose of frame i, and then taking the translation part of the pose. In the case of the odometric constraint this step is straightforward, since absolute positions correspond to the initial state of the graph $\hat{\mathbf{x}} = \{\hat{\mathbf{r}}^1_W, ..., \hat{\mathbf{r}}^N_W\}$ directly obtained from the odometry front-end. In the case of a loop closure constraint, the associated absolute pose ${}_W\hat{\mathbf{T}}^{j_l}_{LC}$ for a loop constraint between frames i_l and j_l is computed as:

$${}_W\hat{\mathbf{T}}^{j_l}_{LC} = {}_W\hat{\mathbf{T}}^{i_l}\,{}_{i_l}\hat{\mathbf{T}}^{j_l}_{LC} \tag{7.12}$$

where transformation ${}_W\hat{\mathbf{T}}^{i_l}$ is taken from the absolute poses given by odometry front-end, and ${}_{i_l}\hat{\mathbf{T}}^{j_l}_{LC}$ is computed by the loop detection front-end. The corresponding absolute position is extracted from the resulting transform, as $\hat{\mathbf{r}}^{j_l}_{W,LC} = {}_W\hat{\mathbf{T}}^{j_l}_{LC}\,(1{:}3, 4)$.

Note that this step is the only one where we use the rotation part of the poses and that it is agnostic of whether the platform is oriented in the direction of movement or not. This means that our approach does not need to assume that the orientation of the platform is aligned with the platform speed vector.

The second step is the computation of the constraints for our optimisation problem applying the observation function to the measures of the absolute positions we have computed in the first step:

$$\hat{z}_k = \boldsymbol{\Delta}\hat{\mathbf{r}}_i^{ij} = \begin{cases} f\left(\mathcal{F}_{\mathbb{R}^n}(\hat{\mathbf{r}}_W^i), \hat{\mathbf{r}}_W^j\right) & \text{if} \quad (i,j) \in \mathcal{S} \\ \\ f\left(\mathcal{F}_{\mathbb{R}^n}(\hat{\mathbf{r}}_W^i), \hat{\mathbf{r}}_{W,LC}^j\right) & \text{if} \quad (i,j) \in \mathcal{R} \end{cases} . \tag{7.13}$$

Then the optimisation is performed analogously to Sec. 7.3. Since we are optimising in $\mathbb{R}^n$, $\boxplus$ and $\boxminus$ operators are the conventional operators for addition and subtraction.

Since the number of properties of the curve is not the same in 2 and 3 dimensions, the definition of function $f(\cdot)$ and the structure of the optimisation problem slightly changes from one case to another. For sake of clearness we treat the two cases separately starting with the easiest 2D case and then stepping up to the 3D case. For each case we proceed as follows: first we compute an observation function $f(\cdot)$ and then show that it is related to the properties of the curve. In next sections we abuse of notation and merge the definition of both points in the curve, and positions, i.e., $\mathbf{r}_i \doteq \mathbf{r}_W^i$.

7.4.3. Loop closure in $\mathbb{R}^2$

In the 2D case we need two positions to define a reference frame, so we take $\mathcal{F}_{\mathbb{R}^2}(\mathbf{r}_W^i) = \{\mathbf{r}_W^{i-1},\ \mathbf{r}_W^i\}$ and define:

$$\Delta\mathbf{r}_0 = \mathbf{r}_W^i - \mathbf{r}_W^{i-1}, \tag{7.14}$$

$$\Delta\mathbf{r}_1 = \mathbf{r}_W^j - \mathbf{r}_W^i. \tag{7.15}$$

The observation function $f\left(\mathbf{r}_W^{i-1}, \mathbf{r}_W^i, \mathbf{r}_W^j\right)$ establishes ternary constraints and is computed as:

$$f\left(\mathbf{r}_W^{i-1}, \mathbf{r}_W^i, \mathbf{r}_W^j\right) = \begin{pmatrix} \mathbf{e}_\mathbf{x}^T \\ \mathbf{e}_\mathbf{y}^T \end{pmatrix} \Delta\mathbf{r}_1 = \frac{1}{||\Delta\mathbf{r}_0||} \begin{pmatrix} {\Delta\mathbf{r}_0}^\intercal \\ {\Delta\mathbf{r}_0}^\intercal \mathbb{J} \end{pmatrix} \Delta\mathbf{r}_1, \tag{7.16}$$

and expresses the odometry vector $\Delta\mathbf{r}_1$ in a reference frame whose unit vector $\mathbf{e}_\mathbf{x}$ is aligned with $\Delta\mathbf{r}_0$. Note that an arbitrary rotation applied both to $\Delta\mathbf{r}_0$and $\Delta\mathbf{r}_1$does not change the returned value of this function.

Now let us see how the observation function $f\left(\mathbf{r}_W^{i-1}, \mathbf{r}_W^i, \mathbf{r}_W^j\right)$, relates with the local properties of the curve in the case of odometric constraints. In our particular problem, the curve is discretised in a set of points, being the scalar which parametrises the curve the i^{th} position index. So we need to apply finite differences to compute the first and second order derivatives:

$$\mathbf{r}'(i) = \mathbf{r}_i - \mathbf{r}_{i-1} = \Delta\mathbf{r}_0, \tag{7.17}$$

$$\mathbf{r}''(i) = \mathbf{r}_j - 2\mathbf{r}_i + \mathbf{r}_{i-1} = \Delta\mathbf{r}_1 - \Delta\mathbf{r}_0. \tag{7.18}$$

Note that for the first order derivative we have taken the backwards difference convention. We hold this convention through the rest of the chapter for $(2n+1)^{th}$ order derivatives.

Applying the definitions of the derivative we can compute the curvature κ_i at a curve point $\mathbf{r}_i$:

$$\kappa_i = \frac{\Delta\mathbf{r}_0{}^{\mathsf{T}}\mathbb{J}\Delta\mathbf{r}_1}{||\Delta\mathbf{r}_0||^3}, \tag{7.19}$$

and putting it into (7.16) we get:

$$f\left(\mathbf{r}_W^{i-1}, \mathbf{r}_W^i, \mathbf{r}_W^j\right) = \begin{pmatrix} ||\Delta\mathbf{r}_1|| \sqrt{1 - \frac{||\Delta\mathbf{r}_0||^4}{||\Delta\mathbf{r}_1||^2}\kappa_i^2} \\ ||\Delta\mathbf{r}_0||^2 \kappa_i^2 \end{pmatrix}, \tag{7.20}$$

verifying that the odometry constraint encapsulates a preservation of the local properties of the curve.

Note that when treating the odometry as a curve, distances between adjacent points must not be zero. Otherwise it is not possible to compute the division by $||\Delta\mathbf{r}_0||$ and the optimisation is likely to fail. Special care must be taken by eliminating redundant points from the initial odometry estimate. It must be remarked that the two poses which a loop is closed at, do not have to obey this restriction since they are never used to compute $\Delta\mathbf{r}_0$. Intuitively speaking, a curve must be always "*in movement*" but it can intersect itself.

7.4.4. Loop closure in $\mathbb{R}^3$

While in a plane we only need one vector to define the reference frame, in the space we require a plane spanned by two vectors to define it unambiguously. Since we need an additional point to compute the second vector, we take $\mathcal{F}_{\mathbb{R}^3}(\mathbf{r}_W^i) = \{\mathbf{r}_W^{i-2},\ \mathbf{r}_W^{i-1},\ \mathbf{r}_W^i\}$ and define:

$$\Delta\mathbf{r}_{-\mathbf{1}} = \mathbf{r}_W^{i-1} - \mathbf{r}_W^{i-2}. \tag{7.21}$$

Thus, the observation function $f\left(\mathbf{r}_W^{i-2}, \mathbf{r}_W^{i-1}, \mathbf{r}_W^i, \mathbf{r}_W^j\right)$ establishes quaternary constraints between positions. To define the observation function we proceed analogously to the 2D case and build a local coordinate frame such that $\mathbf{e_x}$ is aligned with $\Delta\mathbf{r_0}$ and $\mathbf{e_z}$ is the normal of the plane defined by $\Delta\mathbf{r_0}$ and $\Delta\mathbf{r}_{-\mathbf{1}}$ (see Fig.7.2). Notating $[\Delta\mathbf{r_0}]_\times$ as the antisymmetric matrix formed with vector $\Delta\mathbf{r}_0$, the following observation function yields:

$$f\left(\mathbf{r}_W^i, \mathbf{r}_W^{i-1}, \mathbf{r}_W^{i-2}, \mathbf{r}_W^{i-3}\right) = \begin{pmatrix} \mathbf{e_x}^T \\ \mathbf{e_y}^T \\ \mathbf{e_z}^T \end{pmatrix} \Delta\mathbf{r_1} = \begin{pmatrix} \frac{\Delta\mathbf{r_0}^{\mathsf{T}}}{||\Delta\mathbf{r_0}||} \\ -\frac{\Delta\mathbf{r}_{-\mathbf{1}}^{\mathsf{T}}[\Delta\mathbf{r_0}]_\times^2}{\left|\left|[\Delta\mathbf{r_0}]_\times^2 \Delta\mathbf{r}_{-\mathbf{1}}\right|\right|} \\ -\frac{\Delta\mathbf{r}_{-\mathbf{1}}^{\mathsf{T}}[\Delta\mathbf{r_0}]_\times}{||[\Delta\mathbf{r_0}]_\times \Delta\mathbf{r}_{-\mathbf{1}}||} \end{pmatrix} \Delta\mathbf{r_1}. \tag{7.22}$$

However this observation function involves a division by zero when $\Delta\mathbf{r}_{-\mathbf{1}}$ and $\Delta\mathbf{r_0}$ are aligned. Unlike the case where $||\Delta\mathbf{r}_0|| = 0$, this situation is likely to arise in a curve (concretely in straight parts) and and it stems from the fact that it is impossible to define a plane and consequently a reference frame from 3 aligned points. In order to avoid this risky situation the 3 graph must

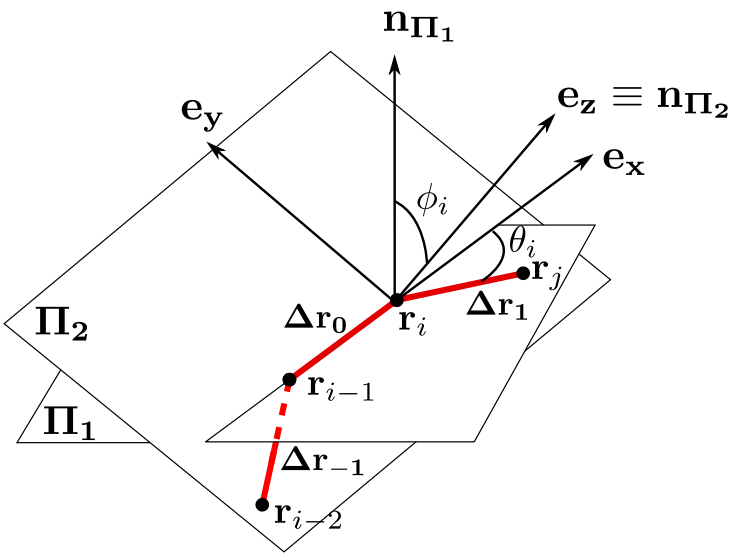


Figure 7.2: Detail of a discrete 3D curve and the variation of its osculating plane

be preprocessed by joining the odometry segments whose relative angle is near zero. As in the 2-dimensional case note, that an arbitrary rotation applied to $\Delta\mathbf{r}_{-1}$, $\Delta\mathbf{r}_0$and $\Delta\mathbf{r}_1$does not change the returned value of the observation function.

As in the 2-dimensional case now we show that in the case of the odometric constraints the crafted observation function encapsulate restrictions on local properties of the curve (length, curvature κ and torsion τ).

Let us first discretise the third derivative,

$$\mathbf{r}'''(i) = \Delta\mathbf{r_{-1}} + \Delta\mathbf{r_1} - 2\Delta\mathbf{r_0}, \tag{7.23}$$

and take (7.17) and (7.18) for the first and second order derivatives to obtain the discretised expressions for curvature and torsion:

$$\kappa_i = \frac{||\Delta\mathbf{r_0} \times \Delta\mathbf{r_1}||}{||\Delta\mathbf{r_0}||^3}, \tag{7.24}$$

$$\tau_i = \frac{\Delta\mathbf{r}_{-\mathbf{1}}^{\mathsf{T}}(\Delta\mathbf{r_0} \times \Delta\mathbf{r_1})}{||\Delta\mathbf{r_0} \times \Delta\mathbf{r_1}||^2}. \tag{7.25}$$

Recalling (7.24) and (7.25) and applying the definitions of the cross and dot product and some trigonometric properties we get

$$f\left(\mathbf{r}_W^{i-2}, \mathbf{r}_W^{i-1}, \mathbf{r}_W^i, \mathbf{r}_W^j\right) = \begin{pmatrix} ||\Delta\mathbf{r_1}|| \sqrt{1 - \kappa_i^2 \frac{||\Delta\mathbf{r_0}||^4}{||\Delta\mathbf{r_1}||^2}} \\ ||\Delta\mathbf{r_0}||^2 \kappa_i \sqrt{1 - \tau_i \frac{\kappa_i^2 ||\Delta\mathbf{r_0}||^8}{\kappa_{i-1}^2 ||\Delta\mathbf{r_{-1}}||^6}} \\ \tau_i \kappa_i^2 \frac{||\Delta\mathbf{r_0}||^5}{||\Delta\mathbf{r_{-1}}||} \end{pmatrix}, \tag{7.26}$$

proving that, as occurs in the 2D case, the crafted odometry function for the 3D case encapsulates a preservation of the local properties of the curve.

7.4.5. Graph Reduction

In [Latif and Neira, 2013] it was shown that graph sparsification by joining poses in segments can greatly reduce the computational cost of a pose-graph optimisation problem. In our approach graph reduction is used to avoid singularities in the observation functions. In the 2D case this is produced by redundant poses in the graph, while in the 3D case singularities are produced, in addition, when poses are aligned, i.e., they form a segment. To avoid this, graphs in both 2D and 3D must be potentially reduced such that all distances between consecutive poses are greater than a threshold th_d, and in addition, for the 3D cases a further reduction must be performed such that the angles between consecutive odometric segments are greater than a threshold th_θ.

7.4.6. Projection of pose-graph from 3D onto 2D space

In some cases, as occurs for example with a visual system on a platform which moves on a planar surface, the odometry and the initial pose-graph are estimated in 3 dimensions though the motion of the platform occurs mostly on a plane. In pose graph optimisation, 2D problems are far more simple than 3D ones, and even some approaches are only able to work in 2-dimensional graphs. For this reason, if it is allowed by the nature of the motion, it can be convenient to convert a 3D graph into a simplified 2D version. In our approach the gain of doing this is two fold. First, we not only reduce the dimension of the problem, but also the number of nodes implied in the constraints. Second, it eliminates the need of dealing with the singularities in 3D graphs which arise from the alignment of points in the trajectory. To generate a 2D graph from a 3D one, we propose the procedure described in Algorithm 2. This essentially consists in aligning the $z-axis$ of the reference frame attached to each pose with the direction of plane which fits better the point cloud formed by all the trajectory points, and then project the poses and the loop constraints onto the new $2D$ given by the computed plane.

Algorithm 2 Projection of 3D pose-graph onto 2D space

Require: $\hat{\mathbf{x}} = \{{}_W\hat{\mathbf{T}}^1,\ {}_W\hat{\mathbf{T}}^2,\ ...,\ {}_W\hat{\mathbf{T}}^N\}$ and $\boldsymbol{T_{LC}} = \{{}_{i_1}\hat{\mathbf{T}}^{j_1}_{LC},\ ...,\ {}_{i_L}\hat{\mathbf{T}}^{j_L}_{LC}\}$ in $\mathbb{SE}(3)$
Ensure: $\hat{\mathbf{x}}_p = \{{}_W\hat{\mathbf{T}}^1_p,\ {}_W\hat{\mathbf{T}}^2_p,\ ...,\ {}_W\hat{\mathbf{T}}^N_p\}$ and $\boldsymbol{T_{LC},p} = \{{}_{i_1}\hat{\mathbf{T}}^{j_1}_{LC,p},\ ...,\ {}_{i_L}\hat{\mathbf{T}}^{j_L}_{LC,p}\}$ in $\mathbb{SE}(2)$
  $\boldsymbol{\Delta}\mathbf{r}^i_W := \mathbf{r}^i_W - \text{mean}(\{\hat{\mathbf{r}}^1_W, ..., \hat{\mathbf{r}}^N_W\})$
  $[\mathbf{U},\ \mathbf{S},\ \mathbf{V}] := \text{svd}\left(\sum_i \boldsymbol{\Delta}\mathbf{r}^i_W \left(\boldsymbol{\Delta}\mathbf{r}^i_W\right)^T\right)$; $\mathbf{v}_{\lambda_{min}} = \mathbf{V}(:,3)$;
  $\mathbf{R}_{pre} := \mathbf{R} \in \mathbb{SO}(3) : \{\mathbf{v}_{pre} = \mathbf{R}\mathbf{v}_{\lambda_{min}}, v_{pre,z} = ||\mathbf{v}_{\lambda_{min}}||_\infty\}$
  $\mathbf{R}_{al} := \mathbf{R} \in \mathbb{SO}(3) : \mathbf{e_z} = \mathbf{R}\mathbf{v}_{pre} := \exp\left(\frac{\text{asin}||\mathbf{v}_{pre}\times\mathbf{e_z}||}{||\mathbf{v}_{pre}\times\mathbf{e_z}||}\mathbf{v}_{pre}\times\mathbf{e_z}\right)$
  for every $\mathbf{T}_{3D} \in \{\hat{\mathbf{x}},\ \boldsymbol{T_{LC}}\}$ **do**
    if $\mathbf{T}_{3D} \in \boldsymbol{T_{LC}}$ **then**
      $\mathbf{T}_{3D} \leftarrow {}_W\hat{\mathbf{T}}^{i_l}\mathbf{T}_{3D} \quad l = loop\_idx$
    end if
    $\mathbf{T}_{3D,al} = \begin{pmatrix}\mathbf{R}_{al} & \mathbf{0}\\ \mathbf{0}^T & 1\end{pmatrix}\begin{pmatrix}\mathbf{R}_{pre} & \mathbf{0}\\ \mathbf{0}^T & 1\end{pmatrix}\mathbf{T}_{3D}\begin{pmatrix}\mathbf{R}^T_{pre} & \mathbf{0}\\ \mathbf{0}^T & 1\end{pmatrix}\begin{pmatrix}\mathbf{R}^T_{al} & \mathbf{0}\\ \mathbf{0}^T & 1\end{pmatrix}$;
    $\theta = \text{atan2}\left(\mathbf{R}_{3D,al}(2,1),\ \mathbf{R}_{3D,al}(1,1)\right)$;
    $\mathbf{T}_{2D} = \begin{pmatrix}\cos\theta & -\sin\theta & r_{3Dal,x}\\ \sin\theta & \cos\theta & r_{3Dal,y}\\ 0 & 0 & 1\end{pmatrix}$;
    if $\mathbf{T}_{3D} \in \hat{\mathbf{x}}$ **then**
      $\hat{\mathbf{x}}_p \leftarrow \{\hat{\mathbf{x}}_p,\ \mathbf{T}_{2D}\}$
    else
      $\boldsymbol{T_{LC},p} \leftarrow \{\boldsymbol{T_{LC},p},\ \left({}_W\hat{\mathbf{T}}^{i_l}_p\right)^{-1}\mathbf{T}_{2D}\}$
    end if
  end for

Table 7.1: Convergence speed comparison of different optimisation approaches in 2D datasets (time in seconds)

Dataset	LM $\mathbb{R}^2$	LM $\mathbb{SE}(2)$	Linear 2D
Manhattan 3500	0.0708 (4 iters)	0.0845 (5 iters)	0.0611 (2 iters)
Manhattan 3500a	0.5144 (20 iters)	0.253 (11 iters)	0.056 (3 iters)
Manhattan 3500b	1.689 (75 iters)	0.6078 (20 iters)	0.049 (3 iters)
Manhattan 3500c	3.1317 (100 iters)	0.566 (30 iters)	0.3656 (30 iters)
Manhattan 10000	1.8314 (10 iters)	1.5707 (10 iters)	0.3648 (2 iters)
City 10000	4.9098 (24 iters)	1.7951 (10 iters)	0.3543 (4 iters)

Table 7.2: Convergence speed comparison of different optimisation approaches in 2D datasets (time in seconds)

Dataset	LM $\mathbb{R}^2$	LM $\mathbb{R}^2$ + pseudoHuber	LM $\mathbb{SE}(2)$	Linear 2D
MIT K.C.	0.6364 (282 iters)	0.1158 (50 iters)	0.0453 (21 iters)	0.00675 (3 iters)
Intel	1.1041 (300 iters)	0.6357 (200 iters)	1.455 (363 iters)	0.01099 (3 iters)

7.5. Experiments

In this section we provide experimental validation of our approach using publicly available data-sets and a comparison with state-of-the art pose graph optimisation methods. The implementation and comparison of our method has been performed within the g^{2}o framework [Kümmerle et al., 2011]. The experiments were performed in an Intel Core i5-2500 at 3.30 GHz.

For the 2D case we have compared three different approaches: our optimisation on $\mathbb{R}^2$ with CSparse solver and the Levenberg-Marquardt algorithm, optimisation on $\mathbb{SE}(2)$ with the CSparse solver and the Levenberg-Marquardt algorithm and a g^{2}o implementation of the linear approximation method for 2D pose graphs of Carlone et al. [Carlone et al., 2011]. The CSparse solver consists of an efficient implementation of a sparse Cholesky factorisation algorithm to solve linear systems and was selected among other available solvers in g^{2}o due to its accuracy and low computation times in the tested data-sets.

The experiments on synthetic 2D datasets (Fig. 7.3) show that in 2D our approach is comparable in accuracy to the other methods. In terms of convergence speed, as shown in Table 7.1, optimisation in $\mathbb{SE}(2)$ and our method (optimisation in $\mathbb{R}^2$) yield a similar performance. However, the linear approximation method outperforms both of them since it only takes few iterations to converge in most of the considered cases, but it is restricted to 2D graphs. Also, we can observe that for increasing levels of noise in the Manhattan3500 our proposal converges to qualitative better solutions than optimising on $\mathbb{SE}(2)$.

In real datasets (Fig. 7.4 and Table 7.2) our method with standard least squares suffers from low accuracy and convergence speed. However, we have noted surprisingly that, though the input graphs do not contain outliers in any of the considered datasets, using a pseudo-Huber robust cost function available in g^{2}o significantly improves both the accuracy and the convergence speed when using our parametrisation. This beneficial effect has been also noted when optimising the MIT Killian Court dataset in $\mathbb{SE}(2)$, which when using a pseudo Huber cost function converges to the same solution as the linear 2D method instead of getting stuck in a local minima.

The 3D case has been tested with two synthetic datasets, sphere2500 and torus10000, from [Kaess et al., 2008] and one real dataset, parking-garage, from [Grisetti et al., 2009].

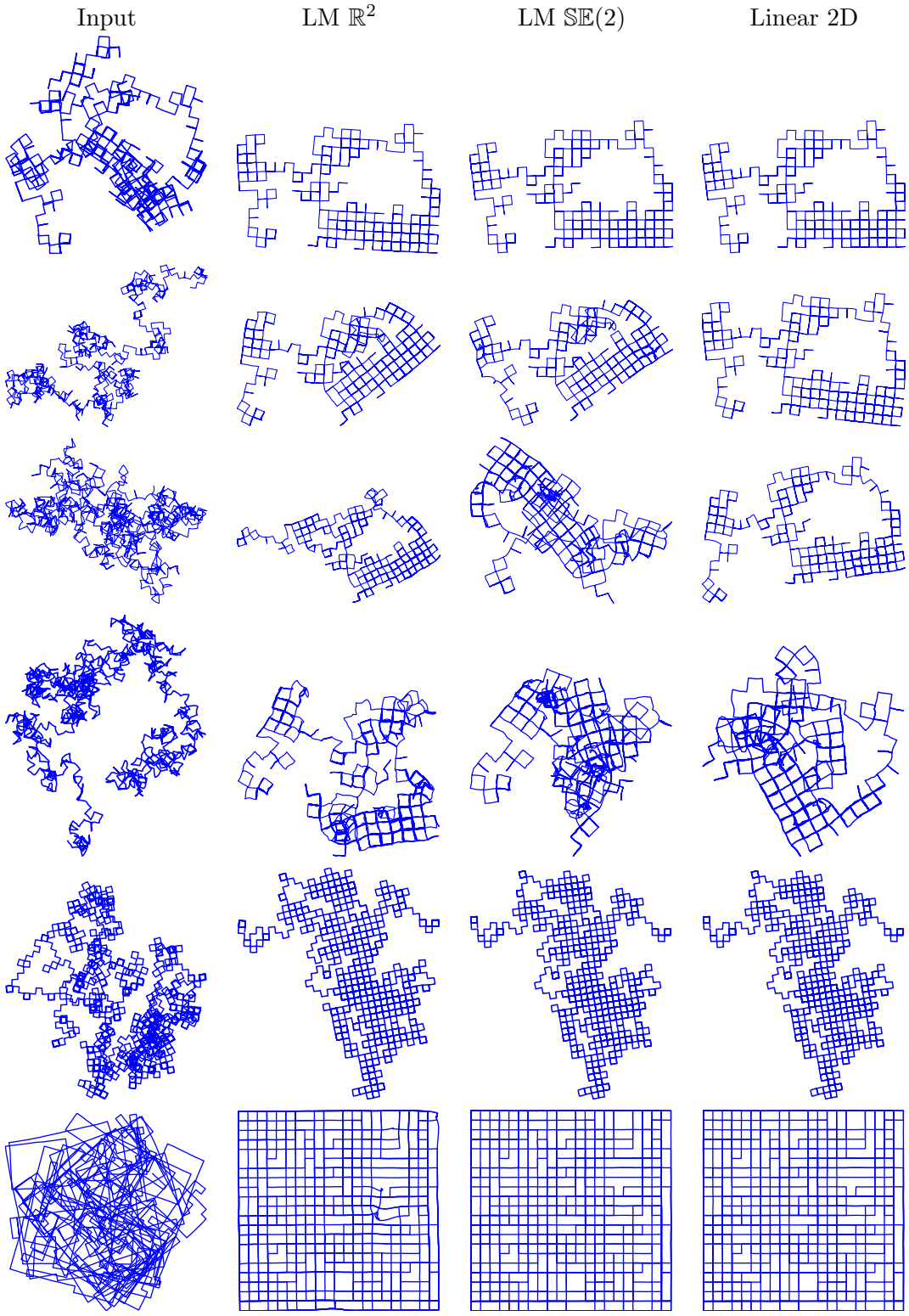


Figure 7.3: Comparison of different pose-graph optimisation methods on 2D synthetic datasets (from top to down) 4 versions of Manhattan3500 [Olson et al., 2006] dataset with increasing levels of noise [Carlone and Censi, 2014], Manhattan10000 [Grisetti et al., 2009] and city10000 [Kaess et al., 2008].

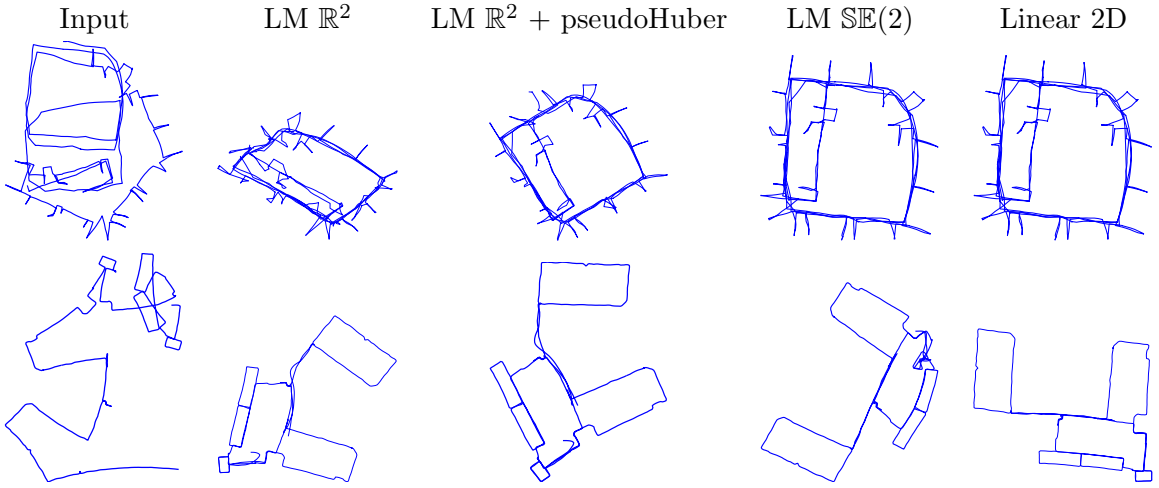


Figure 7.4: Comparison of different pose-graph optimisation methods on real 2D datasets (from top to down) Intel Research Lab. and MIT Killian Court, both from [Kümmerle et al., 2009]

Table 7.3: Convergence speed comparison of different optimisation approaches in the 3D datasets

Dataset	CSparse LM $\mathbb{R}^3$	CSparse LM $\mathbb{SE}(3)$
sphere2500	3.992 s (0.16 s/iter)	3.80634 s (0.30 s/iter)
torus10000	105.916 s (2.83 s/iter)	17.7078s (1.36 s/iter)
parking-garage	0.2652 s (0.039 s/iter)	0.4625 s (0.088 s/iter)

We have used the CSparse solver in the two compared approaches: optimisation in $\mathbb{R}^3$ with Levenberg-Marquardt (ours) and optimisation on $\mathbb{SE}(3)$ with Levenberg-Marquardt. In the tested datasets (Fig. 7.5) we observe that both methods yield similar accuracy. Concerning the convergence speed, the difference of performance between both methods greatly varies between datasets (Table 7.3). In the torus10000 dataset $\mathbb{SE}(3)$ shows a better performance requiring less time per iteration, while in the sphere2500 and the parking-garage datasets our method requires less time per iteration.

Due to the elimination of the orientation from the nodes parametrisation, our approach reduces the dimensions of the optimisation problem by a factor of 2/3 for 2D graphs, and by 1/2 for 3D graphs. Since the cost of the Cholesky factorisation required to solve the linear system at each iteration has order $\mathcal{O}(n^3)$ in dense problems the computational cost per iteration is around 3.5 and 8 times lower for 2D and 3D respectively. However, in the case of sparse problems, which can be efficiently optimised with appropriate solvers, the performance is not only affected by dimension of the problem, but also by the sparsity and also the distribution of the non-zero elements in the Hessian. In this sense the substitution in our approach of binary constraints by ternary or quaternary constraints affect also the computational performance.

In Fig. 7.6 we show the sparsity patterns both of the Jacobians and the Hessians for the cases of the torus10000 graph and a sphere graph with an identical number of poses and constraints. In torus10000 it can be observed that loop constraints in the lower part of the jacobian show some randomness in the distribution of the constraints, while in the case of the sphere graph the constraints show a better arrangement. Note that this last configuration is more realistic since in real world datasets it is a consequence of the spatio temporal consistency, i.e., if there exists

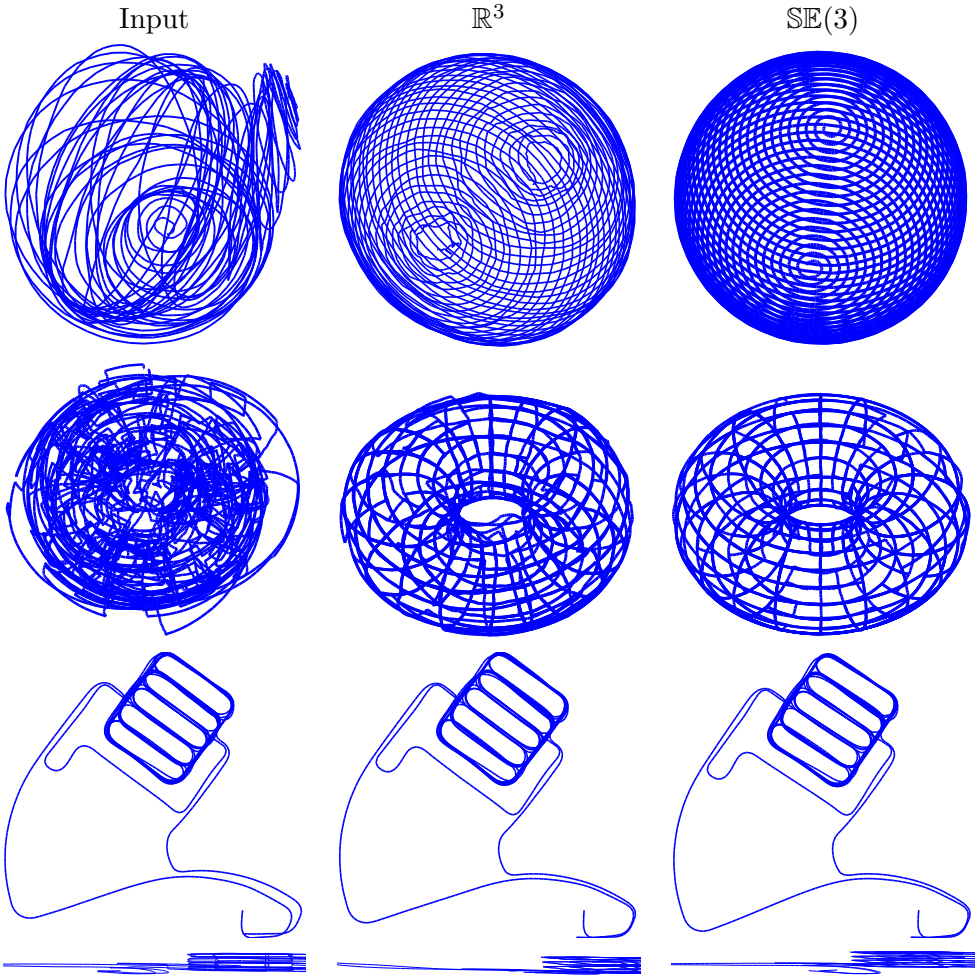
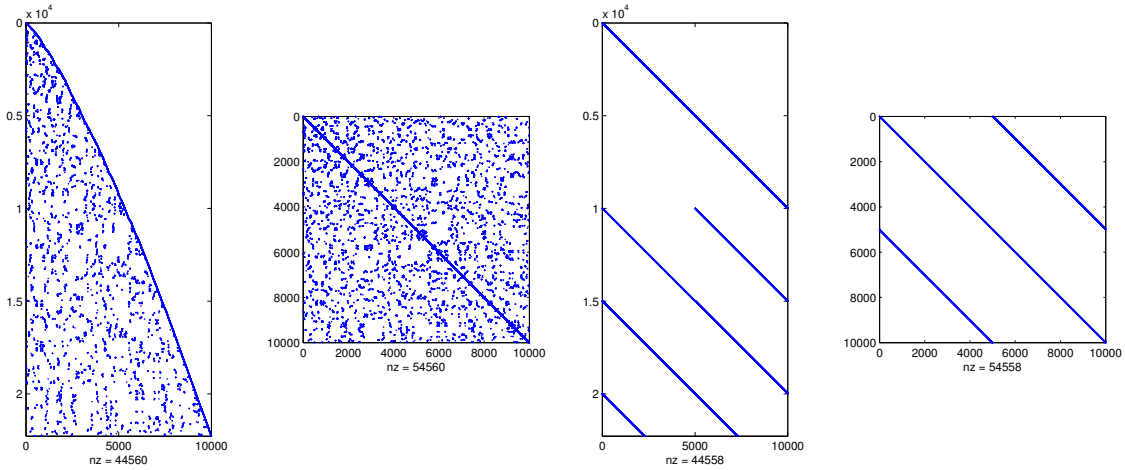


Figure 7.5: Comparison of different optimisation state vector parametrisation 3D datasets. From top to down, sphere2500, torus10000, parking-garage (plant view), parking-garage (side view).

a loop closure between poses i and j, it is quite likely that it holds also for poses $i+1$ and $j+1$. Being the Hessian of both graphs equally sparse, it has been verified that the graph of the sphere yielded a time per iteration one order of magnitude lower than for the torus10000 dataset, being the gain in speed of our parametrisation with respect to $\mathbb{SE}(3)$ similar to the obtained in the rest of the tested 3D datasets. This lead us to conclude that the negative effect on computational performance caused by constraints implying several nodes tends to be aggravated if the Hessian is poorly structured.

Since the dimension of the optimisation problems and the used cost functions are different, the quantitative comparison of χ^2 scores across different methods is not possible. Quantitative comparison is performed then only on those synthetic datasets whose Ground Truth is available (Table 7.4). The column for the $\mathbb{SE}(n)$ in the 2D datasets encapsulates both $LM + \mathbb{SE}(2)$ and $linear2d$ results since they produced the same final graph in the tabulated datasets. We have measured the translational $RMSE$ by taking the error in distance between the position of the corresponding nodes of the ground truth and the evaluated graphs. It can be noted that in the 2D case the accuracy of our proposal is similar and competitive with respect to the state of the art, while in the sphere2500 dataset the loss in accuracy of ours, though producing acceptable



(a) (b)

Figure 7.6: Jacobian and Hessian for (a) the torus10000 graph and for (b) a sphere graph with identical number of nodes and constraints. Note that the sparsity of both Hessians is approximately the same, but the more aleatory distribution of the non-zero elements over the Hessian matrix for the torus10000 datasets yields a higher computational cost of the optimisation.

Table 7.4: RMSE in translational units for synthetic datasets with available ground truth

Dataset	LM $\mathbb{R}^n$	LM $\mathbb{SE}(n)$
Manhattan 3500	0.4892	0.7982
City 10000	0.2875689	0.047672
sphere2500	6.7286	0.2568

visual results Fig. 7.5, is numerically noticeable.

In the last experiment we have evaluated the effect of projecting a nearly planar 3D graph on a plane as described in Sec. 7.4.6. We used one of the datasets where we evaluated of scaled monocular SLAM system from Chapter 3. The dataset was acquired with a wearable omnidirectional camera camera attached to the head over a total distance of 886 m. We apply our proposed curve-graph optimisation to the initial odometry estimate obtained by the SLAM system. Fig. 7.7 shows both the result after optimising the trajectory using our method both in its original 3D space and also in its projection over a 2D plane as explained in Sec. 7.4.6, taking advantage of the fact that the trajectory is quasi planar. It can be observed that the accuracy is improved by projecting the graph and optimising on $\mathbb{R}^2$. Also the computational cost is noticeably reduced from 1.292 s (10 iterations) in $\mathbb{R}^3$ to 0.2073 s (5 iterations) in $\mathbb{R}^2$.

7.6. Discussion

In this work we have presented an alternative formulation for the nodes and constraints in a graph for solving loop closure optimisation problems. Instead of building a pose graph in $\mathbb{SE}(n)$ including both translation and rotation we build a graph where nodes consists of points over the curve which represents the trajectory of the platform in the Euclidean space $\mathbb{R}^n$.

Our method has been evaluated in several public datasets yielding successful results, though

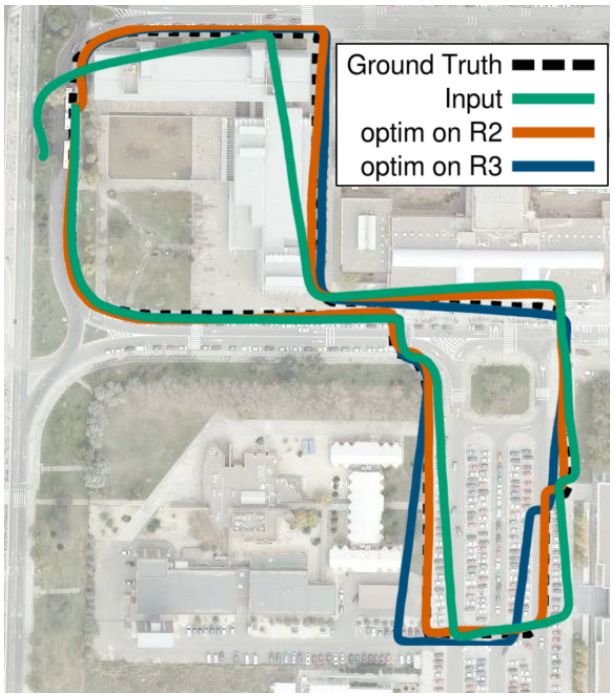


Figure 7.7: Evaluation of our approach in a data-set acquired with an omnidirectional camera, optimising on $\mathbb{R}^2$ after projecting the initial 3D graph on the dominant plane, and optimising on $\mathbb{R}^3$

the observation function used in our approach would seem initially prone to singularities. Compared to state of the art methods for pose graph optimisation yields similar accuracy and computational performance in the 2D datasets. In the 3D case however, our method suffers from a noticeable lack in accuracy, while the computational performance is generally improved but it seems to degrade severally in graphs with a large amount of constraints per nodes.

Although the experiments do not show a clear superiority of our graph parameterisation, we think that results are promising and it could benefit from an improvement in performance with an smarter choice or design of the cost function or the underlying optimisation algorithm. Also, it can find its utility in potential cases where the rotation of the platform is not known or considered or when the information matrix is not available for normalising the translation and rotation residuals.

Chapter 8

Conclusions and Future Work

In this thesis we delved into different problems within the field of visual localisation and mapping, with a special interest in developing solutions and algorithms which can be deployed in conventional or unconventional cameras which can be easily worn or handled by a human.

In the case of Monocular SLAM we have presented a real-time visual SLAM system using monocular omnidirectional vision. Our approach builds on a very successful Visual SLAM system based on the Extended Kalman Filter which initially worked with conventional cameras which can be easily modeled by the typical pin-hole camera model with distortion parameters. We substituted the pin-hole projection model by a generalised model which scopes not only conventional cameras but also any central projection system like the catadioptric omnidirectional cameras. Also to correct the deformation of the image induced by catadioptric cameras and improve the tracking of visual features we introduced an affine transformation to warp the descriptor patch which aims to reach rotation and scale invariance for catadioptric omnidirectional cameras. During experimental evaluation we confirmed that our proposed warping scheme improves the matching rate of image features under different kinds of camera motion. In the context of SLAM this translates into a larger lifespan of the tracked landmarks and larger map sizes.

Our next step in monocular SLAM was to address the scale unobservability problem inherent to monocular systems after noticing that in a wearable vision system body oscillations of amplitudes of the order of centimetres or millimetres caused by walking are accurately estimated when performing SLAM. We developed a method in which, after computing the step frequency of these body oscillations and basing on previous biomedical studies analysing human gait, we obtain a true scaled estimate of the walking speed. Estimating this walking speed in trajectory sections corresponding to windows of a few seconds, we are able to obtain dynamically a true scaled estimate of the map and the visual odometry, avoiding scale drift on long term trajectories. Although the algorithm requires the person to be walking in order to estimate the scale, the experiments, carried out in outdoor and indoor environments and with different types of cameras, showed that our method is reliable and robust to challenging situations like stops, changes in pace or stairs, and provides a significant improvement with respect to the initial unscaled estimate. In the comparison with state-of-the-art for scale drift correction we showed a better or similar performance, and best performance is achieved when both are combined. In this sense we find that as our method extracts the scale from a different source than other approaches it is not exclusive, but can also be combined with other present and future methods for scale computation, in order to get more robust information about the real camera trajectory and 3D observed scene.

In the case of RGB-D vision we have developed a new direct visual odometry method performing dense geometric and photometric error minimisation between frames in GPU.

Compared to the most successful state-of-the-art RGB-D odometry systems based on dense image alignment, our method parametrises the geometric error with the inverse depth instead of the standard depth. This has the advantage of fitting better with the noise model of depth cameras than the standard depth. The experiments leaded both in real and synthetic datasets showed that our method is competitive with the state-of-the-art in RGB-D visual odometry. We outperform it in the majority of the datasets in terms of Relative Pose Error (RPE) and showing low Absolute Trajectory Error (ATE) in spite of not performing loop closure. An implementation of this method has been made public within the scope of a fork from the PCL library.

Also, in parallel we have studied the problem of place recognition with RGB-D sensors when revisiting an environment whose appearance and layout has changed in the meantime. We proposed a series of features computed on superjuts which are segmented over the image based purely on the 3D structure of the scene; and analysed how these features correlated with juts which remained static or moved between images of the same scene captured at different days with a time which elapsed from a few days to months. In our analysis we found that superjuts of the scene which remained static tended to show a lower entropy in their normals, and showed that an image masking basing on this entropy provided a better performance in an standard frame-to-frame matching in the tested datasets.

Following our works with RGB-D systems we started to develop a real-time RGB-D SLAM system with an appearance based place recognition and loop closure system which is robust to changing environment. Our system works in two parallel threads. One thread consists of a front-end which is in charge of processing each incoming frame from the RGB-D camera and estimating the odometry as well as selecting keyframes and performing the integration of frames in keyframes. A second thread consists of a back-end which requests keyframes to the front-end when available and processes these keyframes to obtain a segmentation in superjuts and applying a robust appearance based loop closure with a posterior pose-graph optimisation for correction of the camera trajectory and the 3D map. In our first experiments with our RGB-ID SLAM system we have shown that our approach outperforms most of the state-of-the-art systems for RGB-D SLAM.

Concerning the loop closure problem, we have proposed a new alternative for pose-graph optimisation, which can impose loop closure constraints without the need of including the orientation in the error parametrisation. Our approach, named curve-graph odometry treats a trajectory just as a set of points in the Euclidean Space and it reparametrises the observation function in order to compute the estimate of one constraint in a local reference frame using only the positions. We evaluated this approach in several public datasets yielding successful results, though the observation function used in our approach would seem initially prone to singularities. Compared to state of the art methods for pose graph optimisation, ours yields similar accuracy and computational performance in the 2D datasets. In the 3D case however, our method suffers from a noticeable lack in accuracy, while the computational performance is generally improved but it seems to degrade severally in graphs with a large amount of constraints per nodes. Although the experiments do not show a clear superiority of our graph parameterisation, we think that results are promising and it could benefit from an improvement in performance with an smarter choice or design of the cost function or the underlying optimisation algorithm. Also, it can find its utility in potential cases where the rotation of the platform is not known or considered or when the information matrix is not available for normalising the translation and rotation residuals. Even more generally out of the context of computer vision or robotics, this new approach can be a useful solution to the problem of bending a curve to force it to pass through some control points.

In this work we have proposed different algorithms and methods for metric localisation and mapping with wearable or portable cameras. However the application of the developed algorithms in more concrete practical applications still remains as a work to carry out in the future. As a first step we have recently found that the method presented in this thesis for RGB-D visual odometry produces an increase in performance in an application used for stairs detection and modelling. Beyond this application, the developed methods in this work could be of great utility in a simulator of prosthetics for blind people to provide more accurate models of the environment; and maybe in long term be part of a navigation assistant.

Another path to explore in the future is the fusion with information of other type of sensors. Specially I find of great interest the combination of IMU and vision systems which has started to be explored in the last years in the fields both of robotics and wearable computers. Currently, on SLAM or visual odometry systems which combine IMU and cameras the trend is to introduce information from accelerometers or gyroscopes by means of a filtering paradigm where the linear and angular velocities of the platform is tracked together with the position and orientation. However, from my point of view, it could be possible to introduce information from an IMU without need of tracking velocities, just by introducing constraints between two or maybe more consecutive poses.

Appendix A

Equations Related to Visual EKF-SLAM

In this Appendix we detail the proccess to obtain discrete linear model for error propagation in a constant velocity motion model with white noise in both angular and linear accelerations. This Appendix is divided in two sections. In the first section we detail the obtention of a recursive formula for the computation of the $k-th$ order time integral for a rotation matrix when the angular velocity is constant. This formula will have its utility en the second section where we detail the obtention of the linear model for the error propagation from the continuous differential equations of the motion model.

A.1. Closed form solution for $k-th$ order time integral of rotation matrices

Let the rotation matrix $\mathbf{R}(\boldsymbol{\omega}t)$ describe the rotation undergone under constant velocity $\boldsymbol{\omega}$ during a temporal window t. Let us also denote the $k-th$ order integral of this matrix as:

$$\boldsymbol{\mathcal{IN}}^{(k)}_{\mathbf{R}(\boldsymbol{\omega}t)} \doteq \int_0^t \int_0^{t_1} \cdots \int_0^{t_{k-1}} \mathbf{R}(\boldsymbol{\omega}t_k) dt_k dt_{k-1} dt_1 \tag{A.1}$$

The rotation matrix can be decomposed by its Taylor expansion series:

$$\begin{aligned} \mathbf{R}(\boldsymbol{\omega}t) &= \exp([\boldsymbol{\omega}]_\times t) \\ &= \sum_{n=0}^{\infty} \frac{1}{n!} [\boldsymbol{\omega}]_\times^n t^n \end{aligned} \tag{A.2}$$

With the Taylor decomposition, assuming that $\boldsymbol{\omega}$ is constant along time and using some manipulation and cross product identities,

$$\begin{aligned}\mathcal{IN}^{(k)}_{\mathbf{R}(\boldsymbol{\omega}t)} &= \sum_{n=0}^{\infty} \frac{1}{(n+k)!}[\boldsymbol{\omega}]^n_\times t^{n+k} \\ &= \frac{1}{k!}\mathbf{I}t^k + \frac{1}{(k+1)!}[\boldsymbol{\omega}]_\times t^{k+1} + \sum_{n=2}^{\infty} \frac{1}{(n+k)!}[\boldsymbol{\omega}]^n_\times t^{n+k}\end{aligned} \tag{A.3}$$

$$\frac{1}{k!}\mathbf{I} - \frac{1}{t^k}\mathcal{IN}^{(k)}_{\mathbf{R}(\boldsymbol{\omega}t)} = -\frac{1}{(k+1)!}[\boldsymbol{\omega}]_\times t - \sum_{n=2}^{\infty} \frac{1}{(n+k)!}[\boldsymbol{\omega}]^n_\times t^n \tag{A.4}$$

$$\left(\frac{1}{k!}\mathbf{I} - \frac{1}{t^k}\mathcal{IN}^{(k)}_{\mathbf{R}(\boldsymbol{\omega}t)}\right)\frac{[\boldsymbol{\omega}]_\times}{||\boldsymbol{\omega}||^2 t} = -\frac{1}{(k+1)!}\frac{[\boldsymbol{\omega}]^2_\times}{||\boldsymbol{\omega}||^2} + \sum_{n=2}^{\infty} \frac{1}{(n+k)!}[\boldsymbol{\omega}]^{n-1}_\times t^{n-1} \tag{A.5}$$

$$\begin{aligned}\left(\frac{1}{k!}\mathbf{I} - \frac{1}{t^k}\mathcal{IN}^{(k)}_{\mathbf{R}(\boldsymbol{\omega}t)}\right)\frac{[\boldsymbol{\omega}]_\times}{||\boldsymbol{\omega}||^2 t} + \frac{1}{(k+1)!}\frac{\boldsymbol{\omega}\boldsymbol{\omega}^T}{||\boldsymbol{\omega}||^2} &= \frac{1}{(k+1)!}\mathbf{I} + \sum_{n=2}^{\infty} \frac{1}{(n+k)!}[\boldsymbol{\omega}]^{n-1}_\times t^{n-1} \\ &= \sum_{m=0}^{\infty} \frac{1}{(m+k+1)!}[\boldsymbol{\omega}]^m_\times t^m \\ &= \frac{1}{t^{k+1}}\mathcal{IN}^{(k+1)}_{\mathbf{R}(\boldsymbol{\omega}t)},\end{aligned} \tag{A.6}$$

we finally obtain a recursive formula for the $k-th$ order time integral of the rotation matrix $\mathbf{R}(\boldsymbol{\omega}t)$:

$$\frac{1}{t^k}\mathcal{IN}^{(k)}_{\mathbf{R}(\boldsymbol{\omega}t)} = \left(\frac{1}{(k-1)!}\mathbf{I} - \frac{1}{t^{k-1}}\mathcal{IN}^{(k-1)}_{\mathbf{R}(\boldsymbol{\omega}t)}\right)\frac{[\boldsymbol{\omega}]_\times}{||\boldsymbol{\omega}||^2 t} + \frac{1}{k!}\frac{\boldsymbol{\omega}\boldsymbol{\omega}^T}{||\boldsymbol{\omega}||^2} \qquad k = 1, 2, ... \tag{A.7}$$

with $\mathcal{IN}^{(0)}_{\mathbf{R}(\boldsymbol{\omega}t)} = \mathbf{R}(\boldsymbol{\omega}t)$.

Note that previous equation is indetermined for the limit $t \to 0$. For such cases we have to take the first order approximmation from (A.3):

$$\frac{1}{t^k}\mathcal{IN}^{(k)}_{\mathbf{R}(\boldsymbol{\omega}t)} = \frac{1}{k!}\mathbf{I} + \frac{1}{(k+1)!}[\boldsymbol{\omega}]_\times t \tag{A.8}$$

Of particular interest for us are the first and second order integrals of the rotation matrix:

$$\mathbf{Q}(\boldsymbol{\omega}\Delta t) \doteq \frac{1}{\Delta t}\mathcal{IN}^{(1)}_{\mathbf{R}(\boldsymbol{\omega}\Delta t)} \tag{A.9}$$

$$= \frac{1}{||\boldsymbol{\omega}||\Delta t}\left(\mathbf{I} - \mathbf{R}(\boldsymbol{\omega}\Delta t)\right)\frac{[\boldsymbol{\omega}]_\times}{||\boldsymbol{\omega}||} + \frac{\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}}{||\boldsymbol{\omega}||^2} \tag{A.10}$$

$$\mathbf{S}(\boldsymbol{\omega}\Delta t) \doteq \frac{1}{\Delta t^2}\mathcal{IN}^{(2)}_{\mathbf{R}(\boldsymbol{\omega}\Delta t)} \tag{A.11}$$

$$= \frac{1}{||\boldsymbol{\omega}||\Delta t}\left(\mathbf{I} - \mathbf{Q}(\boldsymbol{\omega}\Delta t)\right)\frac{[\boldsymbol{\omega}]_\times}{||\boldsymbol{\omega}||} + \frac{\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}}{2\,||\boldsymbol{\omega}||^2} \tag{A.12}$$

A.2. Discrete error model equations

To obtain the error model differential equations, first we apply a perturbation to each of the variables in the continuous time motion model in (2.21)-(2.24)

$$\dot{\mathbf{r}}_W^C + \boldsymbol{\delta}\dot{\mathbf{r}}_W^C = \mathbf{v}_W^C + \boldsymbol{\delta}\mathbf{v}_W^C \tag{A.13}$$

$$\exp\left([\delta\boldsymbol{\theta}_W^C]_\times\right)[\delta\dot{\boldsymbol{\theta}}_W^C]_\times {}_W\mathbf{R}^C + \exp\left([\delta\boldsymbol{\theta}_W^C]_\times\right){}_W\dot{\mathbf{R}}^C = \exp\left([\delta\boldsymbol{\theta}_W^C]_\times\right){}_W\mathbf{R}^C[\boldsymbol{\omega}_C^C + \boldsymbol{\delta\omega}_C^C]_\times \tag{A.14}$$

$$\dot{\mathbf{v}}_W^C + \boldsymbol{\delta}\dot{\mathbf{v}}_W^C = \left(\mathbf{I} + [\boldsymbol{\delta\theta}_W^C]_\times\right){}_W\mathbf{R}^C\mathbf{n}_{a_C} \tag{A.15}$$

$$\dot{\boldsymbol{\omega}}_C^C + \boldsymbol{\delta}\dot{\boldsymbol{\omega}}_C^C = \mathbf{n}_{\alpha_C} \tag{A.16}$$

And taking the first order error terms, we have the continuous error model equations:

$$\boldsymbol{\delta}\dot{\mathbf{r}}_W^C = \boldsymbol{\delta}\mathbf{v}_W^C \tag{A.17}$$

$$\boldsymbol{\delta}\dot{\boldsymbol{\theta}}_W^C = {}_W\mathbf{R}^C\boldsymbol{\delta\omega}_C^C \tag{A.18}$$

$$\boldsymbol{\delta}\dot{\mathbf{v}}_W^C = {}_W\mathbf{R}^C\mathbf{n}_{a_C} \tag{A.19}$$

$$\boldsymbol{\delta}\dot{\boldsymbol{\omega}}_C^C = \mathbf{n}_{\alpha_C} \tag{A.20}$$

We start by solving (A.20)

$$\boldsymbol{\delta\omega}_C^C(t) = \boldsymbol{\delta\omega}_{C,i}^C + \mathbf{n}_{\alpha_C}(t - t_i) \tag{A.21}$$

We use the solution to solve (A.18):

$$\begin{aligned}\boldsymbol{\delta\theta}_W^C(t) &= \boldsymbol{\delta\theta}_{W,i}^C + {}_W\mathbf{R}_i^C\left(\int_{t_i}^{t}\mathbf{R}(\boldsymbol{\omega}_{C,i}^C(\tau - t_i))d\tau\right)\boldsymbol{\delta\omega}_{C,i}^C + {}_W\mathbf{R}_i^C\left(\int_{t_i}^{t}(\tau - t_i)\mathbf{R}(\boldsymbol{\omega}_{C,i}^C(\tau - t_i))d\tau\right)\mathbf{n}_{\alpha_C} \\ &= \boldsymbol{\delta\theta}_{W,i}^C + {}_W\mathbf{R}_i^C\mathbf{Q}(\boldsymbol{\omega}_{C,i}^C(t - t_i))(t - t_i)\boldsymbol{\delta\omega}_{C,i}^C \\ &\quad + {}_W\mathbf{R}_i^C\left(\mathbf{Q}(\boldsymbol{\omega}_{C,i}^C(t - t_i)) - \mathbf{S}(\boldsymbol{\omega}_{C,i}^C(t - t_i))\right)(t - t_i)^2\mathbf{n}_{\alpha_C}\end{aligned} \tag{A.22}$$

Eq. (A.19) is solved as:

$$\begin{aligned}\boldsymbol{\delta}\mathbf{v}_W^C(t) &= \boldsymbol{\delta}\mathbf{v}_{W,i}^C + {}_W\mathbf{R}_i^C\left(\int_{t_i}^{t}\mathbf{R}(\boldsymbol{\omega}_{C,i}^C(\tau - t_i))d\tau\right)\mathbf{n}_{a_C} \\ &= \boldsymbol{\delta}\mathbf{v}_{W,i}^C + {}_W\mathbf{R}_i^C\mathbf{Q}(\boldsymbol{\omega}_{C,i}^C(t - t_i))(t - t_i)\mathbf{n}_{a_C}\end{aligned} \tag{A.23}$$

And finally (A.17):

$$\begin{aligned}\boldsymbol{\delta}\mathbf{r}_W^C(t) &= \boldsymbol{\delta}\mathbf{r}_{W,i}^C + \int_{t_i}^{t}\boldsymbol{\delta}\mathbf{v}_W^C(\tau)d\tau \\ &= \boldsymbol{\delta}\mathbf{r}_{W,i}^C + \boldsymbol{\delta}\mathbf{v}_{W,i}^C(t - t_i) + {}_W\mathbf{R}_i^C\mathbf{S}(\boldsymbol{\omega}_{C,i}^C(t - t_i))(t - t_i)^2\mathbf{n}_{a_C}\end{aligned} \tag{A.24}$$

Appendix B

Robust minimisation with M-estimators

In computer vision as well as in other areas, one is frequently facing the problem of, given a set of data samples $\mathbf{z}_i$, having to estimate some latent parameters expressed by the vector $\mathbf{x}$ which better explains the sampled data through some observation model $\mathbf{h}_i(\mathbf{x})$. This is expressed through the following equation:

$$\mathbf{z}_i = \mathbf{h}_i(\mathbf{x}) + \boldsymbol{\epsilon}_i, \tag{B.1}$$

where $\boldsymbol{\epsilon}_i$ represents noisy perturbations in the data.

One frequent assumption is that these perturbations can be modelled by a zero mean gaussian noise with some variance $\boldsymbol{\Sigma}$, $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. This yields the following probability density function for the data samples:

$$f\left(\mathbf{z}_i|\ \mathbf{x}, \boldsymbol{\Sigma}\right) = \frac{1}{\sqrt{(2\pi)^k\left|\boldsymbol{\Sigma}\right|}} \exp\left(-\frac{1}{2}\left(\mathbf{z}_i - \mathbf{h}_i(\mathbf{x})\right)^T \boldsymbol{\Sigma}^{-1}\left(\mathbf{z}_i - \mathbf{h}_i(\mathbf{x})\right)\right) \tag{B.2}$$

where k denotes the number of dimensions of the samples.

Assuming that the samples are independent and identically distributed we obtain the joint density function:

$$f\left(\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N|\ \mathbf{x}\right) = \prod_{i=1}^{N} f\left(\mathbf{z}_i|\ \mathbf{x}, \boldsymbol{\Sigma}\right). \tag{B.3}$$

Now we want to find the parameter vector $\hat{\mathbf{x}}$ which better explains the obtained set of data samples. In statistical terms we want to compute the Maximum Likelihood Estimator of the likelihood function:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \mathcal{L}\left(\mathbf{x}|\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N\right) = \arg\max_{\mathbf{x}} \prod_{i=1}^{N} f\left(\mathbf{z}_i|\ \mathbf{x}, \boldsymbol{\Sigma}\right) \tag{B.4}$$

To convert the product into a sum we maximise instead the log-likelihood function:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \sum_{i=1}^{N} \log\left(f\left(\mathbf{z}_i|\ \mathbf{x}, \boldsymbol{\Sigma}\right)\right). \tag{B.5}$$

Substituting (B.2) we arrive at a standard least squares minimisation problem:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_{i=1}^{N} \mathbf{e}_i^T(\mathbf{x})\boldsymbol{\Sigma}^{-1}\mathbf{e}_i(\mathbf{x}) \tag{B.6}$$

The main problem of performing least squares minimisation, is that as it has been shown during the previous reasoning, it stems from the assumption that the samples from which we are estimating the latent parameters are assumed to follow a Gaussian distribution. While standard least squares has the property of being a linear minimisation problem (as long as the error function is linear with respect to the latent variable) it is quite sensitive to gross measurement errors which do not fit with the Gaussian model. To address this problem, Huber [Huber, 1964b] proposed alternative Maximum Likelihood Estimators (M-estimators) of the form:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_{i=1}^{N} \rho\left(\mathbf{r}_i(\mathbf{x})\right) \tag{B.7}$$

where $\mathbf{r}_i(\mathbf{x}) = \boldsymbol{\Sigma}^{-\frac{1}{2}}\mathbf{e}_i(\mathbf{x})$. The loss function $\rho(\mathbf{r}) : \mathbb{R}^n \to \mathbb{R}$ is the log-likelihood function of an assumed probability density function f and can be seen as an scalar field generated around the equlibrium point $\mathbf{r} = \mathbf{0}$. The intrinsic idea, is that instead of assuming a Gaussian distribution now the error is assumed to have a new distribution f with heavier tails that allow to accomodate gross outliers. The gradient of the loss function, $\boldsymbol{\psi}^T(\mathbf{r}) = \nabla\rho(\mathbf{r})$, is known as the influence function. The election of different M-estimators with their corresponding loss and influence functions basically affects on how the scalar field varies with the distance to the equilibrium point. However, most of them should fulfill the following conditions:

- It must be a positive-definite function, *i.e.*,

$$\rho(\mathbf{0}) = 0, \tag{B.8}$$
$$\rho(\mathbf{r}) \geq 0 \quad \forall \quad \mathbf{r} \in \mathbb{R}^n \setminus \{\mathbf{0}\} \tag{B.9}$$

- $\rho(\mathbf{r})$ only depends on the distance with respect to the equilibrium point, *i.e.*,

$$\boldsymbol{\psi}^T(\mathbf{r}) = \frac{\partial\rho(\mathbf{r})}{\partial\left|\left|\mathbf{r}\right|\right|}\frac{\partial\left|\left|\mathbf{r}\right|\right|}{\partial\mathbf{r}} = \omega(\mathbf{r})\mathbf{r}^T. \tag{B.10}$$

  where $\omega(\mathbf{r})$ is a scalar weight.

- $\rho(\mathbf{r})$ is monotonically increasing with $||\mathbf{r}||$, *i.e.*,

$$\frac{\partial\rho(\mathbf{r})}{\partial\left|\left|\mathbf{r}\right|\right|} \geq 0 \Rightarrow \boldsymbol{\psi}^T(\mathbf{r})\mathbf{r} \geq 0 \tag{B.11}$$

After having defined the properties of the loss function $\rho(\mathbf{r})$, we proceed to differentiate the cost function in (B.7):

$$\sum_{i=1}^{N} \boldsymbol{\psi}^T \left(\mathbf{r}_i(\mathbf{x})\right) \mathbf{J}_i = \mathbf{0}^T \tag{B.12}$$

where $\mathbf{J}_i = \frac{\partial \mathbf{r}_i(\mathbf{x})}{\partial \mathbf{x}}$. At this point we have a non-linear system of equations $\mathbf{F}(\mathbf{x}) = \mathbf{0}$. The usual way of solving it is by the fixed point iteration method, which consists in algebraically manipulating $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ to express it in the form $\mathbf{x} = \mathbf{G}(\mathbf{x})$. Then $\hat{\mathbf{x}}$ is estimated iteratively by the recursive relation $\mathbf{x}^{(\gamma+1)} = \mathbf{G}(\mathbf{x}^{(\gamma)})$. An easy way of obtaining this relation is applying the Newton-Raphson method, where, taking the first order Taylor expansion of $\mathbf{F}(\mathbf{x})$, we obtain $\mathbf{G}(\mathbf{x}) = -\frac{\mathbf{F}(\mathbf{x})}{\mathbf{F}'(\mathbf{x})} + \mathbf{x}$. However with redescending M-estimators, *i.e.*, those for which $\frac{\partial \boldsymbol{\psi}(\mathbf{r})}{\partial \mathbf{r}}$ is not semidefinite positive for every $\mathbf{r} \in \mathbb{R}^n$ this approach is prone to result in failure in convergence to a solution. This is an important issue because many of the most used M- estimators are of the redescendent type, since this property provides more robustness to outliers.

An alternative to the Newton-Rapson method can be obtained if applying (B.10):

$$\sum_{i=1}^{N} \omega \left(\mathbf{r}_i(\mathbf{x})\right) \mathbf{J}_i^T \mathbf{r}_i(\mathbf{x}) = \mathbf{0} \tag{B.13}$$

And then, to generate the form $\mathbf{x} = \mathbf{G}(\mathbf{x})$, expressing the residual vectors by their first order Taylor expansion.

$$\mathbf{r}_i(\mathbf{x}) = \mathbf{r}_i(\mathbf{x}) + \mathbf{J}_i(\mathbf{x} - \mathbf{x}), \tag{B.14}$$

and substituting in (B.13) we obtain:

$$\sum_{i=1}^{N} \omega \left(\mathbf{r}_i(\mathbf{x})\right) \mathbf{J}_i^T \mathbf{r}_i(\mathbf{x}) + \left(\sum_{i=1}^{N} \omega \left(\mathbf{r}_i(\mathbf{x})\right) \mathbf{J}_i^T \mathbf{J}_i \right) (\mathbf{x} - \mathbf{x}) = \mathbf{0}. \tag{B.15}$$

Note that the scalar weight term is not expanded, since in that case we would be applying again the Newton-Raphson method. Performing some algebraic manipulations we finally obtain the recursive relation:

$$\mathbf{x}^{(\gamma+1)} = - \left(\sum_{i=1}^{N} \omega(\mathbf{r}_i(\mathbf{x}^{(\gamma)})) \mathbf{J}_i^T \mathbf{J}_i \right)^{-1} \left(\sum_{i=1}^{N} \omega(\mathbf{r}_i(\mathbf{x}^{(\gamma)})) \mathbf{J}_i^T \mathbf{r}_i(\mathbf{x}^{(\gamma)}) \right) + \mathbf{x}^{(\gamma)} \tag{B.16}$$

This way of solving a least squares optimisation problem with M-estimators is known as the Iteratively Reweighted Least Squares method [Holland and Welsch, 1977], and is usually presented in the literature with the following cost function:

$$\mathbf{x}^{(\gamma+1)} = \arg\min_{\mathbf{x}} \sum_{i=1}^{N} \omega(\mathbf{r}_i(\mathbf{x}^{(\gamma)})) \frac{\mathbf{r}_i^T(\mathbf{x}) \mathbf{r}_i(\mathbf{x})}{2} \tag{B.17}$$

Ideally the chosen loss function to obtain a maximum likelihood estimate should be that steming from the probability distribution from which the data samples are drawn. Unfourtunately, in practice, the process of sampling depends on a lot of factors and circumstances which are difficult to model by a unique probability distribution. In most cases, then, the choice of a M-estimator can be rather empirical and application-oriented. The tendency is to choose an estimator whose associated probability distribution has heavy tails where large errors can be accommodated, and at the same time shows a good performance under the assumption of gaussian noise in the data sampling process. The measure of this performance with respect to a distribution is expressed in terms of its relative variance with respect to the MLE of the distribution. This is denoted in the literature as Asymptotic Relative Efficiency (ARE) and, considering scalar samples for sake of simplicity, it is defined as:

$$ARE(\psi, f) = \frac{\left(\int_{-\infty}^{\infty} \psi'(r) f(r) dr \right)^2}{\int_{-\infty}^{\infty} \psi^2(r) f(r) dr}, \tag{B.18}$$

where $\psi(r)$ is the influence function of the estimator and $f(r)$ is the probability density function of the assumed distribution for samples. Maximum efficiency of 100%, *i.e.*, that of the estimator with the lowest variance, is achieved when $\psi(r) = \frac{\partial}{\partial r}(-\log(f(r)))$. Many M-estimators include a tunning constant which is usually fitted to get an efficiency of 95% for a gaussian distribution. Some of the most used M-estimators in computer vision are listed below:

Table B.1: Some examples of M-estimators

Estimator	$\rho(r)$	$\psi(r)$
Quadratic	$r^2/2$	r
Huber $\begin{cases} r \le k \\ r > k \end{cases}$	$\begin{cases} r^2/2 \\ k(\lvert r\rvert - k/2) \end{cases}$	$\begin{cases} r \\ r(k/\lvert r\rvert) \end{cases}$
Tukey $\begin{cases} r \le c \\ r > c \end{cases}$	$\begin{cases} c^2/6(1-(1-(r/c)^2)^3) \\ c^2/6 \end{cases}$	$\begin{cases} r(1-(r/c)^2)^2 \\ 0 \end{cases}$
Student(Cauchy)	$((\nu+1)/2)\log(1+r^2/\nu)$	$r(\nu+1/\nu+r^2)$

From these the Huber and Tukey estimators do not derive from some known probability distributions. However it can be verified that Huber loss function is very similar to the log-likelihood function of the logistic distribution, $f(r) = \exp(-r)/1+exp(-r)$, which is quadratic as $r \to 0$ and linear as $r \to \infty$. Also Tukey estimator is close to the log-likelihood function of a mixture model of a gaussian distribution and a uniform distribution with a wide support. The Student estimator derives from the Student's t-distribution with ν degrees of freedom, and it is

frequently denoted in the literature as the Cauchy M-estimator with the substitution $\nu = k^2$. In order to achieve an $ARE = 95\%$ at the Gaussian the constants of Huber, Tukey and Student estimators are set to $k = 1.345$, $c = 4.685$, $\nu = 5.688$.

Bibliography

[Absil et al., 2007] Absil, P.-A., Mahony, R., and Sepulchre, R. (2007). *Optimization Algorithms on Matrix Manifolds*. Princeton University Press.

[Agarwal et al., 2009] Agarwal, S., Snavely, N., Simon, I., Seitz, S. M., and Szeliski, R. (2009). Building rome in a day. In *IEEE Int. Conf. on Computer Vision, (ICCV)*, pages 72–79.

[Aghazadeh et al., 2011] Aghazadeh, O., Sullivan, J., and Carlsson, S. (2011). Novelty detection from an ego-centric perspective. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3297–3304.

[Alcantarilla et al., 2012] Alcantarilla, P. F., Yebes, J. J., Almazán, J., and Bergasa, L. M. (2012). On combining visual slam and dense scene flow to increase the robustness of localization and mapping in dynamic environments. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*.

[Anderson et al., 2014] Anderson, S., Dellaert, F., and Barfoot, T. D. (2014). A hierarchical wavelet decomposition for continuous-time SLAM. In *IEEE Int. Conf. on Robotics and Automation, (ICRA)*.

[Andreasson et al., 2007] Andreasson, H., Treptow, A., and Duckett, T. (2007). Self-localization in non-stationary environments using omni-directional vision. *Robotics and Autonomous Systems (RAS)*, 55(7):541–551.

[Aoki et al., 1999] Aoki, H., Schiele, B., and Pentland, A. (1999). Realtime personal positioning system for wearable computers. In *Int. Symp. on Wearable Computing (ISWC)*, pages 37–43.

[Arulampalam et al., 2002] Arulampalam, M. S., Maskell, S., and Gordon, N. (2002). A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing (TSP)*, 50:174–188.

[Audras et al., 2011] Audras, C., Comport, A., Meilland, M., and Rives, P. (2011). Real-time dense rgb-d localisation and mapping. *Aust. Conf. on Robotics and Automation (ACRA)*.

[Badino and Kanade, 2011] Badino, H. and Kanade, T. (2011). A head-wearable short-baseline stereo system for the simultaneous estimation of structure and motion. In *IAPR Conf. on Machine Vision Applications (MVA)*.

[Baglietto et al., 2011] Baglietto, M., Sgorbissa, A., Verda, D., and Zaccaria, R. (2011). Human navigation and mapping with a 6 dof imu and a laser scanner. *Robotics and Autonomous Systems (RAS)*, 59(12):1060–1069.

[Barreto and Araujo, 2001] Barreto, J. and Araujo, H. (2001). Issues on the geometry of central catadioptric image formation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[Bay et al., 2008] Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding (CVIU)*, 110(3):346–359.

[Beaton and Tukey, 1974] Beaton, A. E. and Tukey, J. W. (1974). The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics*, 16(2):147–185.

[Bell and Cathey, 1993] Bell, B. and Cathey, F. (1993). The iterated kalman filter update as a gauss-newton method. *IEEE Trans. on Automatic Control (TAC)*, 38(2):294–297.

[Bondarev et al., 2013] Bondarev, E., Heredia, F., Favier, R., Ma, L., and de With, P. H. N. (2013). On photo-realistic 3d reconstruction of large-scale and arbitrary-shaped environments. In *IEEE Consumer Communications and Networking Conf. (CCNC)*, pages 621–624.

[Botterill et al., 2012] Botterill, T., Mills, S., and Green, R. (2012). Correcting scale drift by object recognition in single camera slam. *IEEE Trans. on Systems, Man, and Cybernetics–Part B: Cybernetics*, 43(6):1767–1780.

[Boucher, 2006] Boucher, P. (2006). Points on a sphere. http://www.softimageblog.com/archives/115.

[Bradski, 2000] Bradski, G. (2000). The opencv library. *Dr. Dobb's Journal of Software Tools*, 25(11).

[Bylow et al., 2013] Bylow, E., Sturm, J., Kerl, C., Kahl, F., and Cremers, D. (2013). Real-time camera tracking and 3d reconstruction using signed distance functions. In *Robotics: Science and Systems (RSS)*.

[Cadena and Kosecka, 2013] Cadena, C. and Kosecka, J. (2013). Semantic parsing for priming object detection in rgb-d scenes. In *Semantic Perception Mapping and Exploration (SPME) Workshop, held with ICRA*.

[Carlone et al., 2011] Carlone, L., Aragues, R., Castellanos, J. A., and Bona, B. (2011). A linear approximation for graph-based simultaneous localization and mapping. In *Robotics: Science and Systems (RSS)*.

[Carlone and Censi, 2014] Carlone, L. and Censi, A. (2014). From angular manifolds to the integer lattice: Guaranteed orientation estimation with application to pose graph optimization. *IEEE Trans. on Robotics (T-RO)*, 30(2):475–492.

[Castle et al., 2010] Castle, R. O., Klein, G., and Murray, D. W. (2010). Combining monoslam with object recognition for scene augmentation using a wearable camera. *Image and Vision Computing (IVC)*, 28(11):1548–1556.

[Chang et al., 2009] Chang, R., Ieng, S., and Benosman, R. (2009). Auto-organized visual perception using distributed camera network. *Robotics and Autonomous Systems*, 57(11):1075–1082.

[Civera et al., 2008] Civera, J., Davison, A. J., and Montiel, J. M. M. (2008). Inverse depth parametrization for monocular slam. *IEEE Trans. on Robotics*, 24(5):932–945.

[Civera et al., 2010] Civera, J., Grasa, O. G., Davison, A. J., and Montiel, J. M. M. (2010). 1-Point RANSAC for EKF Filtering: application to real-time structure from motion and visual odometry. *J. of Field Robotics*, 27(5):609–631.

[Corke et al., 2004] Corke, P., Strelow, D., and Singh, S. (2004). Omnidirectional visual odometry for a planetary rover. In *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*.

[Couprie et al., 2013] Couprie, C., Farabet, C., Najman, L., and LeCun, Y. (2013). Indoor semantic segmentation using depth information. *CoRR*, abs/1301.3572.

[Cumani et al., 2004] Cumani, S., Denasi, A., Guiducci, A., and Quaglia, G. (2004). Integrating monocular vision and odometry for slam. *WSEAS Trans. on Computers*, 3:625–630.

[Cummins and Newman, 2008] Cummins, M. and Newman, P. (2008). FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *The Int. Journal of Robotics Research (IJRR)*, 27(6):647–665.

[Cummins and Newman, 2011] Cummins, M. and Newman, P. (2011). Appearance-only slam at large scale with fab-map 2.0. *Int. J. of Robotics Research (IJRR)*, 30(9):1100–1123.

[Damen et al., 2012] Damen, D., Gee, A. P., Mayol-Cuevas, W. W., and Calway, A. (2012). Egocentric real-time workspace monitoring using an rgb-d camera. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 1029–1036.

[Davison, 2003] Davison, A. J. (2003). Real-time simultaneous localisation and mapping with a single camera. In *Int. Conf. on Computer Vision (ICCV)*, volume 2, pages 1403–1410.

[Davison et al., 2007] Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O. (2007). Monoslam: Real-time single camera slam. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 29:1052–1067.

[Dong et al., 2014] Dong, H., Figueroa, N., and El Saddik, A. (2014). Towards consistent reconstructions of indoor spaces based on 6d rgb-d odometry and kinectfusion. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 1796–1803.

[Dubbelman and Browning, 2013] Dubbelman, G. and Browning, B. (2013). Closed-form online pose-chain slam. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 5190–5197.

[Dubbelman et al., 2010] Dubbelman, G., Esteban, I., and Schutte, K. (2010). Efficient trajectory bending with applications to loop closure. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 4836–4842.

[Duckett et al., 2002] Duckett, T., Marsland, S., and Shapiro, J. (2002). Fast, on-line learning of globally consistent maps. *Autonomous Robots (AURO)*, 12(3):287–300.

[Endres et al., 2012] Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., and Burgard, W. (2012). An evaluation of the RGB-D SLAM system. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*.

[Endres et al., 2014] Endres, F., Hess, J., Sturm, J., Cremers, D., and Burgard, W. (2014). 3D mapping with an RGB-D camera. *IEEE Trans. on Robotics (T-RO)*, 30(1):177–187.

[Engel et al., 2012] Engel, J., Sturm, J., and Cremers, D. (2012). Camera-based navigation of a low-cost quadrocopter. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robot Systems (IROS)*.

[Engel et al., 2013] Engel, J., Sturm, J., and Cremers, D. (2013). Semi-dense visual odometry for a monocular camera. In *IEEE Int. Conf. on Computer Vision (ICCV)*.

[Eudes et al., 2010] Eudes, A., Lhuillier, M., Naudet-Collette, S., and Dhome, M. (2010). Fast odometry integration in local bundle adjustment-based visual SLAM. In *International Conference on Pattern Recognition*, pages 290–293.

[Fathi et al., 2012] Fathi, A., Hodgins, J. K., and Rehg, J. M. (2012). Social interactions: A first-person perspective. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1226–1233.

[Fathi et al., 2011] Fathi, A., Ren, X., and Rehg, J. M. (2011). Learning to recognize objects in egocentric activities. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3281–3288.

[Felzenszwalb and Huttenlocher, 2004] Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. *Int. Journal of Computer Vision (IJCV)*, 59(2):167–181.

[Finman et al., 2013] Finman, R., Whelan, T., Kaess, M., and Leonard, J. J. (2013). Toward lifelong object segmentation from change detection in dense rgb-d maps. In *Eur. Conf. on Mobile Robotics (ECMR)*, pages 178–185.

[Frese et al., 2005] Frese, U., Larsson, P., and Duckett, T. (2005). A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Trans. on Robotics (T-RO)*, 21(2):196–207.

[Frigo and Johnson, 2005] Frigo, M. and Johnson, S. G. (2005). The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231. Special issue on "Program Generation, Optimization, and Platform Adaptation".

[Gálvez-López and Tardos, 2012] Gálvez-López, D. and Tardos, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Trans. on Robotics (T-RO)*, 28(5):1188–1197.

[Gee and Mayol-Cuevas, 2012] Gee, A. P. and Mayol-Cuevas, W. W. (2012). 6d relocalisation for rgbd cameras using synthetic view regression. In *British Machine Vision Conf. (BMVC)*, pages 1–11.

[Geyer and Daniilidis, 2000] Geyer, C. and Daniilidis, K. (2000). A unifying theory for central panoramic systems and practical applications. In *Eur. Conf. on Computer Vision (ECCV)*, pages 445–461.

[Gordon et al., 1993] Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107–113.

[Grieve and Gear, 1966] Grieve, D. and Gear, R. J. (1966). The relationships between length of stride, step frequency, time of swing and speed of walking for children and adults. *Ergonomics*, 5(9):379–399.

[Grimes et al., 2010] Grimes, M. K., Anguelov, D., and LeCun, Y. (2010). Hybrid hessians for flexible optimization of pose graphs. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 2997–3004.

[Grisetti et al., 2009] Grisetti, G., Stachniss, C., and Burgard, W. (2009). Nonlinear constraint network optimization for efficient map learning. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*, 10(3):428–439.

[Guerrero et al., 2015] Guerrero, J. J., Pérez-Yus, A., Gutiérrez-Gómez, D., Rituerto, A., and López-Nicolás, G. (2015). Human navigation assistance with a RGB-D sensor. In *DRT4ALL 2015 Congreso Internacional de Diseño, Redes de Investigacin y Tecnologa para todos.*

[Gutiérrez et al., 2011] Gutiérrez, D., Rituerto, A., Montiel, J. M. M., and Guerrero, J. J. (2011). Adapting a real-time monocular visual slam from conventional to omnidirectional cameras. In *11th OMNIVIS, held with International Conference on Computer Vision (ICCV).*

[Gutiérrez-Gómez and Guerrero, 2013] Gutiérrez-Gómez, D. and Guerrero, J. J. (2013). Scaled monocular slam for walking people. In *Int. Symp. on Wearable Computing (ISWC).*

[Gutiérrez-Gómez and Guerrero, 2014] Gutiérrez-Gómez, D. and Guerrero, J. J. (2014). Curve-graph odometry: Removing the orientation in loop closure optimisation problems. In *Int. Conf. on Intelligent Autonomous Systems (IAS).*

[Gutiérrez-Gómez and Guerrero, 2015] Gutiérrez-Gómez, D. and Guerrero, J. J. (2015). Curve-graph odometry: Orientation-free error parameterisations for loop closure problems. *Robotics and Autonomous Systems (RAS)*, 74(Part B). Special Issue on Intelligent Autonomous Systems Conf. (IAS-13).

[Gutiérrez-Gómez and Guerrero, 2016] Gutiérrez-Gómez, D. and Guerrero, J. J. (2016). True scaled 6 dof egocentric localisation. *Image and Vision Computing (IVC).* Under Review.

[Gutiérrez-Gómez et al., 2015a] Gutiérrez-Gómez, D., Mayol-Cuevas, W., and Guerrero, J. J. (2015a). Inverse depth for accurate photometric and geometric error minimisation in rgb-d dense visual odometry. In *Proc. IEEE/RSJ Int. Conf. on Robotics and Automation (ICRA).*

[Gutiérrez-Gómez et al., 2015b] Gutiérrez-Gómez, D., Mayol-Cuevas, W., and Guerrero, J. J. (2015b). What should i landmark? entropy of normals in depth juts for place recognition in changing environments using rgb-d data. In *Proc. IEEE/RSJ Int. Conf. on Robotics and Automation (ICRA).*

[Gutiérrez-Gómez et al., 2016] Gutiérrez-Gómez, D., Mayol-Cuevas, W., and Guerrero, J. J. (2016). Dense rgb-d visual odometry using inverse depth. *Robotics and Autonomous Systems (RAS)*, 75(Part B):571 – 583. Special Section on 3D Perception with PCL.

[Gutiérrez-Gómez et al., 2012] Gutiérrez-Gómez, D., Puig, L., and Guerrero, J. J. (2012). Full scaled 3d visual odometry from a single wearable omnidirectional camera. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robot Systems (IROS)*, pages 4276–4281.

[Handa et al., 2014] Handa, A., Whelan, T., McDonald, J., and Davison, A. J. (2014). A benchmark for rgb-d visual odometry, 3d reconstruction and SLAM. In *IEEE Int. Conf. on Robotics and Automation (ICRA).*

[Hartley and Zisserman, 2000] Hartley, R. and Zisserman, A. (2000). *Multiple View geometry in Computer vision.* Cambridge university press, 2nd edition.

[Heikkila et al., 2009] Heikkila, M., Pietikainen, M., and Schmid, C. (2009). Description of interest regions with local binary patterns. *Pattern Recognition*, 42(3):425–436.

[Henry et al., 2010] Henry, P., Krainin, M., Herbst, E., Ren, X., and Fox, D. (2010). RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In *Int. Symp. on Experimental Robotics (ISER)*.

[Henry et al., 2012] Henry, P., Krainin, M., Herbst, E., Ren, X., and Fox, D. (2012). RGB-D mapping: Using kinect-style depth cameras for dense 3D modeling of indoor environments. *Int. Journal of Robotics Research (IJRR)*, 31(5):647–663.

[Herbst et al., 2011] Herbst, E., Henry, P., Ren, X., and Fox, D. (2011). Toward object discovery and modeling via 3-d scene comparison. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2623–2629.

[Herbst et al., 2013] Herbst, E., Ren, X., and Fox, D. (2013). Rgb-d flow: Dense 3-d motion estimation using color and depth. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2276–2282.

[Hertzberg et al., 2013] Hertzberg, C., Wagner, R., Frese, U., and Schröder, L. (2013). Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds. *Information Fusion (INFFUS)*, 14(1):57–77.

[Hesch and Roumeliotis, 2010] Hesch, J. A. and Roumeliotis, S. I. (2010). Design and analysis of a portable indoor localization aid for the visually impaired. *Int. Journal of Robotics Research (IJRR)*, 29(11):1400–1415.

[Hodges et al., 2011] Hodges, S., Berry, E., and Wood, K. (2011). Sensecam: a wearable camera that stimulates and rehabilitates autobiographical memory. *Memory*, 19(7):685–96.

[Holland and Welsch, 1977] Holland, P. W. and Welsch, R. E. (1977). Robust Regression Using Iteratively Reweighted Least-Squares. *Communications in Statistics: Theory and Methods*, A6:813–827.

[Horn and Schunck, 1981] Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence*, 17(1–3):185 – 203.

[Horn et al., 1988] Horn, B. K. P., Hilden, H., and Negahdaripour, S. (1988). Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society America*, 5(7):1127–1135.

[Huang et al., 2011] Huang, A. S., Bachrach, A., Henry, P., Krainin, M., Fox, D., and Roy, N. (2011). Visual odometry and mapping for autonomous flight using an rgb-d camera. In *In Int. Symp. of Robotics Research (ISRR)*.

[Huber, 1964a] Huber, P. J. (1964a). Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35(1):73–101.

[Huber, 1964b] Huber, P. J. (1964b). Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35(1):73–101.

[Jaimez and González-Jiménez, 2015] Jaimez, M. and González-Jiménez, J. (2015). Fast visual odometry for 3-d range sensors. *IEEE Transactions on Robotics*, 31(4):809–822.

[Kaess et al., 2008] Kaess, M., Ranganathan, A., and Dellaert, F. (2008). iSAM: Incremental smoothing and mapping. *IEEE Trans. on Robotics (T-RO)*, 24(6):1365–1378.

[Karpathy et al., 2013] Karpathy, A., Miller, S., and Fei-Fei, L. (2013). Object discovery in 3d scenes via shape analysis. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2088–2095.

[Kerl et al., 2013a] Kerl, C., Sturm, J., and Cremers, D. (2013a). Dense visual slam for rgb-d cameras. In *IEEE/RSJ Conf. on Intelligent Robots and Systems (IROS)*, pages 2100–2106.

[Kerl et al., 2013b] Kerl, C., Sturm, J., and Cremers, D. (2013b). Robust odometry estimation for rgb-d cameras. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3748–3754.

[Khoshelham and Elberink, 2012] Khoshelham, K. and Elberink, S. O. (2012). Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454.

[Kitani et al., 2011] Kitani, K. M., Okabe, T., Sato, Y., and Sugimoto, A. (2011). Fast unsupervised ego-action learning for first-person sports videos. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3241–3248.

[Klein and Murray, 2007] Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *IEEE and ACM Int. Symp. on Mixed and Augmented Reality (ISMAR)*.

[Klose et al., 2013] Klose, S., Heise, P., and Knoll, A. (2013). Efficient Compositional Approaches for Real-Time Robust Direct Visual Odometry from RGB-D Data. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*.

[Kneip and Furgale, 2014] Kneip, L. and Furgale, P. (2014). OpenGV: A unified and generalized approach to real-time calibrated geometric vision. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*.

[Konolige and Bowman, 2009] Konolige, K. and Bowman, J. (2009). Towards lifelong visual maps. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 1156–1163.

[Konolige and Mihelich, 2014] Konolige, K. and Mihelich, P. (2010, [Online; accessed 25-September-2014]). Technical description of kinect calibration. `http://wiki.ros.org/kinect_calibration/technical`.

[Kourogi et al., 2001] Kourogi, M., Kurata, T., and Sakaue., K. (2001). A panorama-based method of personal positioning and orientation and its real-time applications for wearable computers. In *Int. Symp. on Wearable Computing (ISWC)*, pages 107–114.

[Kümmerle et al., 2011] Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). g^{2}o: A general framework for graph optimization. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3607–3613.

[Kuo, 2001] Kuo, A. D. (2001). A simple model of bipedal walking predicts the preferred speed-step length relationship. *Journal of Biomechanical Engineering*, 123:264–269.

[Kümmerle et al., 2009] Kümmerle, R., Steder, B., Dornhege, C., Ruhnke, M., Grisetti, G., Stachniss, C., and Kleiner, A. (2009). On measuring the accuracy of slam algorithms. *Autonomous Robots*, 27(4):387–407.

[Lange et al., 1989] Lange, K. L., Little, R. J. A., and Taylor, J. M. G. (1989). Robust Statistical Modeling Using the t Distribution. *Journal of the American Statistical Association*, 84(408):881–896.

[Latif et al., 2013] Latif, Y., Cadena, C., and Neira, J. (2013). Robust loop closing over time for pose-graph slam. *Int. Journal of Robotics Research (IJRR)*, 32(14):1611–1626.

[Latif and Neira, 2013] Latif, Y. and Neira, J. (2013). Go straight, turn right: Pose graph reduction through trajectory segmentation using line segments. In *Eur. Conf. on Mobile Robots (ECMR)*.

[Liu and Rubin, 1995] Liu, C. and Rubin, D. B. (1995). ML estimation of the t distribution using EM and its extensions, ECM and ECME. *Statistica Sinica*, 5:19–39.

[Lothe et al., 2010] Lothe, P., Bourgeois, S., Royer, E., Dhome, M., and Naudet-Collette, S. (2010). Real-time vehicle global localisation with a single camera in dense urban areas: Exploitation of coarse 3d city models. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 863–870.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision (IJCV)*, 60(2):91–110.

[Lu and Zheng, 2010] Lu, H. and Zheng, Z. (2010). Two novel real-time local visual features for omnidirectional vision. *Pattern Recognition*, 43:3938–3949.

[Lui et al., 2012] Lui, W. L. D., Tang, T. J. J., Drummond, T., and Li, W. H. (2012). Robust egomotion estimation using ICP in inverse depth coordinates. In *IEEE Int. Conf. on Robotics and Automation, (ICRA)*, pages 1671–1678.

[Lupton and Sukkarieh, 2008] Lupton, T. and Sukkarieh, S. (2008). Removing scale biases and ambiguity from 6dof monocular slam using inertial. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3698–3703.

[Mann, 1997] Mann, S. (1997). Smart clothing: The wearable computer and wearcam. *Personal Technologies*, 1(1):21–27.

[Mann et al., 2011] Mann, S., Huang, J., Janzen, R., Lo, R., Rampersad, V., Chen, A., and Doha, T. (2011). Blind navigation with a wearable range camera and vibrotactile helmet. In *ACM Int. Conf. on Multimedia (ICMM)*, pages 1325–1328.

[Martínez et al., 2010] Martínez, J. L., Morales, J., Mandow, A., and García-Cerezo, A. (2010). Incremental closed-form solution to globally consistent 2d range scan mapping with two-step pose estimation. In *IEEE Int. Workshop on Advanced Motion Control (AMC)*, pages 252–257.

[Martinez and Stiefelhagen, 2013] Martinez, M. and Stiefelhagen, R. (2013). Kinect unleashed: Getting control over high resolution depth maps. In *IAPR Int. Conf. on Machine Vision Applications, (MVA)*, pages 247–250.

[Mayol et al., 2003] Mayol, W. W., Davison, A. J., Tordoff, B. J., and Murray, D. W. (2003). Applying active vision and slam to wearables. In *In Int. Symp. on Robotics Research (ISRR)*, pages 325–334.

[Mayol-Cuevas et al., 2009] Mayol-Cuevas, W. W., Tordoff, B. J., and Murray, D. W. (2009). On the choice and placement of wearable vision sensors. *IEEE Trans. on Systems Man and Cybernetics Part A*, 39(2):414–425.

[Mei, 2007] Mei, C. (2007). *Laser-Augmented Omnidirectional Vision for 3D Localisation and Mapping*. PhD thesis, INRIA Sophia Antipolis, Project-team ARobAS.

[Mei and Rives, 2007] Mei, C. and Rives, P. (2007). Single view point omnidirectional camera calibration from planar grids. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*.

[Mei et al., 2010] Mei, C., Sibley, G., Cummins, M., Newman, P., and Reid, I. (2010). RSLAM: A system for large-scale mapping in constant-time using stereo. *Int. Journal of Computer Vision (IJCV)*, 94(2):198–214.

[Meilland and Comport, 2013a] Meilland, M. and Comport, A. (2013a). On unifying key-frame and voxel-based dense visual SLAM at large scales. In *IEEE/RSJ Int. Conf.on Intelligent Robots and Systems (IROS)*.

[Meilland et al., 2011] Meilland, M., Comport, A., and Rives, P. (2011). Dense visual mapping of large scale environments for real-time localisation. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*.

[Meilland and Comport, 2013b] Meilland, M. and Comport, A. I. (2013b). Super-resolution 3d tracking and mapping. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 5717–5723.

[Mitzel and Leibe, 2012] Mitzel, D. and Leibe, B. (2012). Close-range human detection for head-mounted cameras. In *British Machine Vision Conf. (BMVC)*.

[Molton et al., 2004] Molton, N., Davison, A., and Reid, I. (2004). Locally planar patch features for real-time structure from motion. In *British Machine Vision Conf. (BMVC)*.

[Montemerlo et al., 2002] Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. (2002). FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *In AAAI National Conference on Artificial Intelligence*, pages 593–598. AAAI.

[Mur-Artal et al., 2015] Mur-Artal, R., Montiel, J. M. M., and Tardós, J. D. (2015). ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. on Robotics (T-RO)*, 31(5):1147–1163.

[Mur-Artal and Tardós, 2014] Mur-Artal, R. and Tardós, J. D. (2014). Fast relocalisation and loop closing in keyframe-based slam. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*.

[Murillo et al., 2012] Murillo, A. C., Gutiérrez-Gómez, D., Rituerto, A., Puig, L., and Guerrero, J. J. (2012). Wearable omnidirectional vision system for personal localization and guidance. In *2nd IEEE Workshop on Egocentric (First-Person) Vision, held with CVPR*.

[Newcombe et al., 2011a] Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. W. (2011a). Kinectfusion: Real-time dense surface mapping and tracking. In *Int. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 127–136.

[Newcombe et al., 2011b] Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011b). Dtam: Dense tracking and mapping in real-time. In *Int. Conf. on Computer Vision (ICCV)*, pages 2320–2327.

[Nistér et al., 2006] Nistér, D., Naroditsky, O., and Bergen, J. (2006). Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23:3–20.

[Nützi et al., 2010] Nützi, G., Weiss, S., Scaramuzza, D., and Siegwart, R. (2010). Fusion of imu and vision for absolute scale estimation in monocular slam. *Journal of Intelligent Robotic Systems*, 61(1-4):287–299.

[Olson and Agarwal, 2013] Olson, E. and Agarwal, P. (2013). Inference on networks of mixtures for robust robot mapping. *Int. Journal of Robotics Research (IJRR)*, 32(7):826–840.

[Olson et al., 2006] Olson, E., Leonard, J. J., and Teller, S. J. (2006). Fast iterative alignment of pose graphs with poor initial estimates. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2262–2269.

[Park, 1995] Park, F. C. (1995). Distance metrics on the rigid-body motions with applications to mechanism design. *J. of Mechanical Design*, 117(1):48–54.

[Paz et al., 2008] Paz, L. M., Piniés, P., Tardós, J. D., and Neira, J. (2008). Large scale 6 dof slam with stereo-in-hand. *IEEE Trans. on Robotics (T-RO)*, 24(5):946–957.

[Peasley and Birchfield, 2014] Peasley, B. and Birchfield, S. (2014). Fast and accurate poseslam by combining relative and global state spacess. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*.

[Pérez-Yus et al., 2016] Pérez-Yus, A., Gutiérrez-Gómez, D., López-Nicolas, G., and Guerrero, J. J. (2016). Stairs detection with odometry-aided traversal from a wearable RGB-D camera.

[Pirsiavash and Ramanan, 2012] Pirsiavash, H. and Ramanan, D. (2012). Detecting activities of daily living in first-person camera views. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2847–2854.

[Pressley, 2001] Pressley, A. (2001). *Elementary Differential Geometry*. Springer.

[Ren et al., 2012] Ren, X., Bo, L., and Fox, D. (2012). Rgb-(d) scene labeling: Features and algorithms. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2759–2766.

[Ren and Gu, 2010] Ren, X. and Gu, C. (2010). Figure-ground segmentation improves handled object recognition in egocentric video. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3137–3144.

[Rituerto et al., 2010] Rituerto, A., Puig, L., and Guerrero, J. J. (2010). Visual slam with an omnidirectional camera. In *Int. Conf. on Pattern Recognition (ICPR)*, pages 348–351.

[Rosten et al., 2010] Rosten, E., Porter, R., and Drummond, T. (2010). Faster and better: A machine learning approach to corner detection. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 32:105–119.

[Rusu and Cousins, 2011] Rusu, R. B. and Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China.

[Scaramuzza et al., 2007] Scaramuzza, D., Criblez, N., Martinelli, A., and Siegwart, R. (2007). Robust Feature Extraction and Matching for Omnidirectional Images. In *Int. Conf. on Field and Service Robotics* .

[Scaramuzza et al., 2009a] Scaramuzza, D., Fraundorfer, F., Pollefeys, M., and Siegwart, R. (2009a). Absolute scale in structure from motion from a single vehicle mounted camera by exploiting nonholonomic constraints. In *IEEE Int. Conf. on Computer Vision (ICCV)*, pages 1413–1419.

[Scaramuzza et al., 2009b] Scaramuzza, D., Fraundorfer, F., and Siegwart, R. (2009b). Real-time monocular visual odometry for on- road vehicles with 1-point RANSAC. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 4293–4299.

[Silberman et al., 2012] Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *Eur. Conf. on Computer Vision (ECCV)*, pages 746–760.

[Solà et al., 2012] Solà, J., Vidal-Calleja, T., Civera, J., and Montiel, J. M. (2012). Impact of landmark parametrization on monocular EKF-SLAM with points and lines. *Int. Journal of Computer Vision (IJCV)*, 97(3):339–368.

[Steinbrücker et al., 2011] Steinbrücker, F., Sturm, J., and Cremers, D. (2011). Real-time visual odometry from dense rgb-d images. In *IEEE Int. Conf. on Computer Vision Workshops (ICCVW)*, pages 719–722.

[Strasdat, 2012] Strasdat, H. (2012). *Local Accuracy and Global Consistency for Efficient Visual SLAM*. PhD thesis, Department of Computing, Imperial College London.

[Strasdat et al., 2010] Strasdat, H., Montiel, J. M. M., and Davison, A. (2010). Scale drift-aware large scale monocular slam. In *Robotics: Science and Systems (RSS)*.

[Stuckler and Behnke, 2012] Stuckler, J. and Behnke, S. (2012). Integrating depth and color cues for dense multi-resolution scene mapping using rgb-d cameras. In *IEEE Conf. on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 162–167.

[Stückler and Behnke, 2014] Stückler, J. and Behnke, S. (2014). Multi-resolution surfel maps for efficient dense 3d modeling and tracking. *Journal of Visual Communication and Image Representation*, 25(1):137–147.

[Sturm et al., 2012] Sturm, J., Engelhard, N., Endres, F., Burgard, W., and Cremers, D. (2012). A benchmark for the evaluation of rgb-d slam systems. In *IEEE/RSJ Int. Conf. on Intelligent Robot Systems (IROS)*.

[Sünderhauf and Protzel, 2012] Sünderhauf, N. and Protzel, P. (2012). Switchable constraints for robust pose graph slam. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*.

[Svoboda and Pajdla, 2001] Svoboda, T. and Pajdla, T. (2001). Matching in catadioptric images with appropriate windows, and outliers removal. In *Computer Analysis of Images and Patterns (CAIP)*, pages 733–740, London, UK.

[Tang et al., 2012] Tang, S., Wang, X., Lv, X., Han, T. X., Keller, J. M., He, Z., Skubic, M., and Lao, S. (2012). Histogram of oriented normal vectors for object recognition with a depth sensor. In *Assian Conf. on Computer Vision (ACCV)*, pages 525–538.

[Tardif et al., 2008] Tardif, J.-P., Pavlidis, Y., and Daniilidis, K. (2008). Monocular visual odometry in urban environments using an omnidirectional camera. In *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*.

[Targhi et al., 2006] Targhi, A. T., Hayman, E., Eklundh, J.-O., and Shahshahani, M. (2006). The eigen-transform and applications. In *Assian Conf. on Computer Vision (ACCV)*, pages 70–79.

[Thrun et al., 2005] Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. MIT Press.

[Tomasi and Manduchi, 1998] Tomasi, C. and Manduchi, R. (1998). Bilateral filtering for gray and color images. In *Int. Conf. on Computer Vision (ICCV)*, pages 839–846.

[Triebel et al., 2010] Triebel, R., Shin, J., and Siegwart, R. (2010). Segmentation and unsupervised part-based discovery of repetitive objects. In *Robotics: Science and Systems (RSS)*.

[Triggs et al., 2000] Triggs, B., McLauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (2000). Bundle adjustment - a modern synthesis. In *Vision Algorithms: Theory and Practice*, pages 298–372.

[Tykkala et al., 2011] Tykkala, T., Audras, C., and Comport, A. I. (2011). Direct iterative closest point for real-time visual odometry. In *IEEE Int. Conf. on Computer Vision Workshops (ICCVW)*.

[Ulrich and Nourbakhsh, 2000] Ulrich, I. and Nourbakhsh, I. R. (2000). Appearance-based place recognition for topological localization. In *Int. Conf. on Robotics and Automation (ICRA)*, pages 1023–1029.

[Weiss and Siegwart, 2011] Weiss, S. and Siegwart, R. (2011). Real-time metric state estimation for modular vision-inertial systems. In *Int. Conf. on Robotics and Automation (ICRA)*, pages 4531–4537.

[Whelan et al., 2013a] Whelan, T., Johannsson, H., Kaess, M., Leonard, J. J., and McDonald, J. (2013a). Robust real-time visual odometry for dense rgb-d mapping. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*.

[Whelan et al., 2014] Whelan, T., Kaess, M., Finman, R., Fallon, M., Johannsson, H., Leonard, J., and McDonald, J. (2014). Real-time large scale dense rgb-d slam with volumetric fusion. *The Int. Journal of Robotics Research (IJRR)*, 34(4-5):598–626.

[Whelan et al., 2013b] Whelan, T., Kaess, M., Leonard, J. J., and McDonald, J. B. (2013b). Deformation-based loop closure for large scale dense RGB-D SLAM. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*.

[Whelan et al., 2015] Whelan, T., Leutenegger, S., Salas-Moreno, R. F., Glocker, B., and Davison, A. J. (2015). ElasticFusion: Dense SLAM without a pose graph. In *Robotics: Science and Systems (RSS)*.

[Williams et al., 2007] Williams, B., Klein, G., and Reid, I. (2007). Real-time SLAM relocalisation. In *Int. Conf. on Computer Vision (ICCV)*.

[Williams et al., 2011] Williams, B. P., Klein, G., and Reid, I. (2011). Automatic relocalization and loop closing for real-time monocular slam. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 33(9):1699–1712.

[Zarrugh et al., 1974] Zarrugh, M., Todd, F., and Ralston, H. (1974). Optimization of energy expenditure during level walking. *European Journal of Applied Physiology and Occupational Physiology*, 33:293–306.

List of Figures

List of Tables