UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Departamento de Electrónica

Vision Based Localization: From Humanoid Robots to Visually Impaired People

A Thesis submitted for the degree of Doctor of Philosophy

Author

Pablo Fernández Alcantarilla

Supervisor

Dr. D. Luis Miguel Bergasa Pascual

2011

A mi abuela Carmen

Agradecimientos

En primer lugar, quisiera darle las gracias a Luis Miguel Bergasa por la oportunidad que me ofreció para realizar esta tesis doctoral. Por su apoyo constante a lo largo de todos estos años, las innumerables revisiones de artículos y documentos, las importantes reuniones de seguimiento, y por permitirme disfrutar de los congresos, estancias de investigación y un curso de SLAM en ¡Australia!.

A estas alturas, uno se pregunta si la tesis doctoral ha servido para algo. Al menos el doctorado me ha permitido conocer medio mundo (Australia, China, Japón, USA...) y conocer multitud de gente, de los cuáles muchos de ellos ya se han convertido en amigos. Así que únicamente por esto, creo que el doctorado ha merecido la pena. También, gracias al doctorado me he dado cuenta de lo pequeño que es el mundo, y de la cantidad de mundo que aún me queda por conocer.

Tengo muchísimo que agradecer a Frank Dellaert. Sin su ayuda, esta tesis no hubiera sido posible. Frank, me dió la oportunidad de acudir a su laboratorio en Atlanta cuando mis conocimientos en visión y robótica dejaban algo que desear. Desde el primer día me sentí plenamente integrado en su grupo, y trabajar allí fue una gran experiencia. En especial, quería mostrar palabras de agradecimiento a Sang Min y Gian Luca, que me acompañaron durante mis primeros pasos en el mundo de la *visibilidad*. Y a Kai, por seguir acompañándome en el mundo de la visibilidad. También me gustaría agradecer la amabilidad y compañía del resto de integrantes del BORG lab: Michael, Misha, Christian, Grant, Richard, Carlos (aunque sea un fanático del Barcelona) y Alireza (¿Vamos a tomar café?). ¡Y cómo olvidarme del resto de mis amigos por Atlanta! Gracias a la comunidad española de Atlanta, Mari Carmen, Raúl y Guillermo, por hacerme sentir como en casa y a mis amigos de la India, Balaji y Sumit.

Gracias a Andrew Davison, por permitirme visitar su laboratorio en Londres. Trabajar en el Imperial College y en una ciudad tan fantástica como Londres, fue una magnífica experiencia. Quisiera destacar la calidad humana de Andy, así como del resto de gente del grupo por su gran acogida, Adrien, Ankur, *pinche* Gerardo, Margarita, Hatice, Richard, Steve y Hauke. No se que hubiera sido de mi en Londres, sin mi gran amigo Jorge Copa. Gracias por introducirme en el mundo *Pret* y por las buenas fiestas que hemos pasado. Espero que pronto podamos volver de fiesta por Londres.

Gracias a Olivier Stasse y a Eiichi Yoshida, por permitirme visitar el Joint Robotics Laboratory en el AIST de Tsukuba, Japón y trabajar en el fascinante mundo de los robots humanoides. Creo que vivir un tiempo en Japón, ha sido de las mejores experiencias de mi vida y es una experiencia que se la recomiendo a todo el mundo. Gracias al resto de integrantes del JRL: Abder, Francoise, Sebastien, HRP-2, HRP-4, Tokiko, Antoine, Pierre, Gustavo, Wael, Nicola, Torea, Gregoire, Benjamin (*wing man*) y a *mi amigo* Maxime. También quería agradecer los buenos momentos con la comunidad española de Tsukuba, así como la ayuda de mis amigos japoneses para integrarme en la cultura y el idioma japonés. Espero volver algún día pronto por Japón.

A lo largo de estos años, mucha gente ha pasado por el laboratorio del fondo O-34 y por el grupo de investigación RobeSafe. Muchas gracias a todos por hacer las horas de trabajo más amenas y llevaderas. Gracias a Jesús, Almazán, Álvaro, Gavilán, Balky, Sebas, Carlos, Dani Pizarro, Llorca, Nacho, Miguel Ángel Sotelo, Manuel Ocaña, Rafael Barea, Javi Yebes, Sergio, a los alcarreños Álvaro, Rubén y Edu, a Noelia por ser mi compañera de aprendizaje de kanjis, a los Pedros Jiménez y Revenga, a Llamazares, Óscar, *Dirty* Fer, Iván y el amigo *brasileiro* Paulo. Gracias a todos por los buenos momentos que hemos pasado.

En especial quería agradecer a Luis Miguel, Almazán y Javi Yebes, por su colaboración en las grabaciones con usuarios de la ONCE. Del mismo modo, muchísimas gracias a la ONCE, Technosite y a los voluntarios que realizaron las pruebas. Espero que en algún día cercano pueda ofreceros un producto para mejorar vuestra calidad de vida.

Quería agradecer a mis amigos por animarme a terminar la *dichosa* tesis y por hacerme el camino más llevadero. Especialmente quería agradecer a Sergio y Javi por estar siempre ahí desde prácticamente toda la vida, a César *Sueco Loco* por su gran amistad y palabras de ánimo, a David por aguantarme mis rollos sobre la visión artificial, al resto de la panda de Linköping y Madrid, a los amigos de la uni, a Marta, a Javi Blanco y Roberto por los buenos ratos y por las visitas durante mis estancias en Londres y en Japón.

Y por último, a las personas que más tengo que agradecer son mis padres y mi hermano, así como el resto de mi familia. Gracias por el apoyo durante todos estos años de tesis doctoral, por vuestro apoyo en los momentos díficiles y por respetar mis decisiones.

No dejes que tu fuego se extinga, chispa a chispa irremplazables, en los pantanos deshauciados de lo incompleto, del todavía no, del absolutamente no.

No dejes que perezca el héroe que hay en tu alma, en una frustración solitaria por la vida que merecías, pero nunca has podido alcanzar.

El mundo que anhelas puede conseguirse, existe, es real, es posible, y es tuyo.

Ayn Rand.

Resumen

En la actualidad, las aplicaciones 3D presentan un gran interés en diversos campos tales como la robótica, la visión artificial o la realidad aumentada. Mediante el uso de cámaras y técnicas de visión artificial, se pueden obtener modelos 3D precisos en grandes entornos tales como ciudades. Además, las cámaras son unos sensores no invasivos y de bajo coste en comparación con otros sensores tales como el láser y que ofrecen una gran información sobre el entorno.

Una aplicación de gran interés es la localización visual en un mapa 3D. Los robots necesitan realizar tareas en el entorno de manera autónoma, y para la realización de estas tareas es necesario conocer en que posición se encuentran dentro un mapa de manera precisa. Del mismo modo, proporcionar información de posición y orientación puede ser de mucha utilidad para personas ciegas o con problemas de visión. La movilidad o capacidad de desplazarse de forma independiente y segura tanto en entornos conocidos, como en entornos desconocidos, puede llegar a ser un gran reto para las personas que presentan ceguera o algún tipo de deficiencia visual. Los sistemas comerciales de ayuda a la movilidad de personas invidentes, están basados en tecnología de posicionamiento por satélite GPS. Sin embargo, esta tecnología no es fiable en entornos urbanos para la comunidad de personas invidentes, ya que presenta errores de localización elevados del orden de varios metros y otros problemas asociados a la tecnología GPS como pérdida de la señal o escasa visibilidad de satélites. La tecnología GPS no funciona si no existe un número mínimo de satélites visibles. Por consiguiente, esta tecnología no puede ser utilizada en entornos de interiores. Por lo tanto, es necesario investigar nuevos métodos de localización más precisos y robustos.

En esta tesis se desarrollan diversos algoritmos para obtener una localización visual precisa y en tiempo real a partir de un mapa 3D conocido. Para obtener una localización robusta es necesario calcular previamente un mapa 3D del entorno. Para calcular dicho mapa 3D, se utilizan técnicas conocidas como Simultaneous Localization and Mapping (SLAM) o Structure from Motion (SfM). En esta tesis se presenta un sistema de SLAM utilizando una cámara estéreo como único sensor que nos permite obtener reconstrucciones 3D precisas del entorno. El sistema de SLAM propuesto es capaz de detectar posibles objetos en movimiento en un rango cercano a la cámara de aproximadamente 5 metros, gracias a un módulo desarrollado de detección de objetos en movimiento. Los objetos en movimiento se detectan gracias a una representación densa conocida como *scene flow* que nos permite obtener información sobre la velocidad de los puntos 3D del entorno. Este módulo resulta muy eficaz en entornos muy dinámicos en los que suelen existir una gran cantidad de objetos dinámicos tales como peatones. A partir del módulo de detección de objetos en movimiento se evita incorporar puntos 3D erróneos al proceso de SLAM, obteniendo mejores resultados de reconstrucción 3D. Desde nuestro conocimiento, es la primera vez que se aplica la técnica de *scene flow* denso y detección de objetos en movimiento en el contexto de SLAM visual para entornos complejos y dinámicos, tales

como los que se presentan en esta Tesis.

Tanto en las técnicas de SLAM como en los algoritmos de localización visual, los puntos 3D del mapa se identifican mediante descriptores de apariencia. A partir de estos descriptores, se realiza la asociación de datos de un punto 3D con una característica 2D detectada en la imagen. En esta tesis se ha desarrollado una familia nueva de descriptores de apariencia llamada Gauge-Speeded Up Robust Features (G-SURF), los cuáles se basan en el uso de las coordenadas *gauge*. A partir de este tipo de representación, para cada píxel en la imagen se define un nuevo sistema de coordenadas basado en la estructura local alrededor del píxel de interés. Dicho sistema de coordenadas se define a partir del vector gradiente y la dirección perpendicular a este en el píxel de interés. Se ha realizado una evaluación experimental detallada en aplicaciones de matching, reconocimiento de categorías visuales y aplicaciones de reconstrucción 3D que demuestran la utilidad y mejores resultados de los descriptores G-SURF con respecto a otras propuestas en el estado del arte tales como los descriptores Scale Invariant Feature Transform (SIFT) o SURF.

En las aplicaciones de localización visual, uno de los pasos que presentan una mayor carga computacional es la asociación de datos entre un mapa grande de puntos 3D y las características 2D detectadas en la imagen. Los métodos tradicionales normalmente basan esta asociación de datos únicamente en información de apariencia. Estos algoritmos pueden llevar una carga computacional elevada y en entornos con texturas repetitivas, dicha asociación de datos puede dar lugar a corresponencias erróneas. En esta tesis se ha desarrollado un algoritmo para la predicción de la visibilidad de puntos 3D utilizando técnicas de aprendizaje sobre una reconstrucción 3D previa. Gracias a estas técnicas de aprendizaje, se obtiene una mejor y más rápida asociación de datos gracias a la predicción de la visibilidad de los puntos 3D para una pose de cámara.

Se han desarrollado y evaluado algoritmos de SLAM y localización visual utilizando información de una sola cámara y un mapa 3D previo para dos aplicaciones diferentes de gran interés: robots humanoides y personas con deficiencia visual. En el caso de los robots humanoides, se ha evaluado el algoritmo desarrollado de localización visual monocular con predicción de visibilidad en distintos escenarios y diversos tipos de secuencias tales como trayectorias rectangulares, circulares, con personas moviéndose en el entorno, cambios de iluminación, etc. Se ha realizado una comparativa del error del sistema de localización y mapeado con respecto a un sistema preciso de captura de movimiento, que demuestra que los errores son del orden de pocos centímetros. También se ha comparado el sistema de localización visual con el algoritmo Parallel Tracking and Mapping (PTAM), obteniendo mejores resultados con el sistema de localización visual propuesto en esta tesis. Respecto a la aplicación de localización de personas con deficiencia visual, se ha evaluado un sistema de localización visual monocular en secuencias de interiores de tipo oficina. También, se ha evaluado el sistema de visual SLAM con detección de objectos de movimiento en pruebas reales con usuarios invidentes considerando entornos interiores muy dinámicos tales como el interior de la estación de trenes de Atocha (Madrid, España) y en la ciudad de Alcalá de Henares (Madrid, España). Los resultados obtenidos demuestran que los algoritmos desarrollados puede ser de gran interés para aplicaciones de localización de usuarios invidentes en grandes entornos.

Abstract

Nowadays, 3D applications have recently become a more and more popular topic in robotics, computer vision or augmented reality. By means of cameras and computer vision techniques, it is possible to obtain accurate 3D models of large-scale environments such as cities. In addition, cameras are low-cost, non-intrusive sensors compared to other sensors such as laser scanners. Furthermore, cameras also offer a rich information about the environment.

One application of great interest is the vision-based localization in a prior 3D map. Robots need to perform tasks in the environment autonomously, and for this purpose, is very important to know precisely the location of the robot in the map. In the same way, providing accurate information about the location and spatial orientation of the user in a large-scale environment can be of benefit for those who suffer from visual impairment problems. A safe and autonomous navigation in unknown or known environments, can be a great challenge for those who are blind or are visually impaired. Most of the commercial solutions for visually impaired localization and navigation assistance are based on the satellite Global Positioning System (GPS). However, these solutions are not suitable enough for the visually impaired community in urban-environments. The errors are about of the order of several meters and there are also other problems such GPS signal loss or line-of-sight restrictions. In addition, GPS does not work if an insufficient number of satellites are directly visible. Therefore, GPS cannot be used for indoor environments. Thus, it is important to do further research on new more robust and accurate localization systems.

In this thesis we propose several algorithms in order to obtain an accurate real-time vision-based localization from a prior 3D map. For that purpose, it is necessary to compute a 3D map of the environment beforehand. For computing that 3D map, we employ well-known techniques such as Simultaneous Localization and Mapping (SLAM) or Structure from Motion (SfM). In this thesis, we implement a visual SLAM system using a stereo camera as the only sensor that allows to obtain accurate 3D reconstructions of the environment. The proposed SLAM system is also capable to detect moving objects especially in a close range to the camera up to approximately 5 meters, thanks to a moving objects detection module. This is possible, thanks to a dense scene flow representation of the environment, that allows to obtain the 3D motion of the world points. This moving objects detection module seems to be very effective in highly crowded and dynamic environments, where there are a huge number of dynamic objects such as pedestrians. By means of the moving objects detection module we avoid adding erroneous 3D points into the SLAM process, yielding much better and consistent 3D reconstruction results. Up to the best of our knowledge, this is the first time that dense scene flow and derived detection of moving objects has been applied in the context of visual SLAM for challenging crowded and dynamic environments, such as the ones presented in this Thesis.

In SLAM and vision-based localization approaches, 3D map points are usually de-

scribed by means of appearance descriptors. By means of these appearance descriptors, the data association between 3D map elements and perceived 2D image features can be done. In this thesis we have investigated a novel family of appearance descriptors known as Gauge-Speeded Up Robust Features (G-SURF). Those descriptors are based on the use of *gauge* coordinates. By means of these coordinates every pixel in the image is fixed separately in its own local coordinate frame defined by the local structure itself and consisting of the gradient vector and its perpendicular direction. We have carried out an extensive experimental evaluation on different applications such as image matching, visual object categorization and 3D SfM applications that show the usefulness and improved results of G-SURF descriptors against other state-of-the-art descriptors such as the Scale Invariant Feature Transform (SIFT) or SURF.

In vision-based localization applications, one of the most expensive computational steps is the data association between a large map of 3D points and perceived 2D features in the image. Traditional approaches often rely on purely appearence information for solving the data association step. These algorithms can have a high computational demand and for environments with highly repetitive textures, such as cities, this data association can lead to erroneous results due to the ambiguities introduced by visually similar features. In this thesis we have done an algorithm for predicting the visibility of 3D points by means of a memory based learning approach from a prior 3D reconstruction. Thanks to this learning approach, we can speed-up the data association step by means of the prediction of visible 3D points given a prior camera pose.

We have implemented and evaluated visual SLAM and vision-based localization algorithms for two different applications of great interest: humanoid robots and visually impaired people. Regarding humanoid robots, a monocular vision-based localization algorithm with visibility prediction has been evaluated under different scenarios and different types of sequences such as square trajectories, circular, with moving objects, changes in lighting, etc. A comparison of the localization and mapping error has been done with respect to a precise motion capture system, yielding errors about the order of few cm. Furthermore, we also compared our vision-based localization system with respect to the Parallel Tracking and Mapping (PTAM) approach, obtaining much better results with our localization algorithm. With respect to the vision-based localization approach for the visually impaired, we have evaluated the vision-based localization system in indoor and cluttered office-like environments. In addition, we have evaluated the visual SLAM algorithm with moving objects detection considering test with real visually impaired users in very dynamic environments such as inside the Atocha railway station (Madrid, Spain) and in the city center of Alcalá de Henares (Madrid, Spain). The obtained results highlight the potential benefits of our approach for the localization of the visually impaired in large and cluttered environments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Vision-based 3D applications, such as reconstructions of cities or localization have recently become a more and more popular topic in computer vision and robotics. Cameras are an appealing sensor for 3D applications. They are small, light-weight, cheaper than other sensors and can also provide range information. With the introduction of handheld devices equipped with cameras, accelerometers or Global Positioning System (GPS) sensors (e.g. mobile phones), extending these 3D applications to such devices is very much desired. However, most of the approaches are designed for offline processing due to their high computational cost [Agarwal et al., 2009; Furukawa et al., 2010]. Therefore, it is necessary to develop new algorithms that can provide robust real-time localization and can cope with large maps of 3D points. For example, Figure 1.1 depicts different topics where 3D vision-based applications can have a tremendous impact in the next future.

(a) iPhone (b) HRP-2 humanoid robot (c) A visually impaired user

Figure 1.1: Examples of different scenarios that would benefit from real-time 3D vision-based applications.

For a robust vision-based localization, it is necessary to compute an accurate 3D map of the environment. Normally, at the same time the map of the environment is computed, the camera trajectory is also estimated simultaneously. This is the well-known Simultaneous Localization and Mapping (SLAM) [Durrant-White and Bailey, 2006] problem in the robotics community. In computer vision, this problem is also known as Structure from Motion (SfM) [Hartley and Zisserman, 2000]. The final goal of visual SLAM and SfM is obtaining an accurate and consistent 3D representation of the environment. Then, this

map can be used for long-term localization or navigation purposes. Another alternative for vision-based localization is using a prior 3D map of the environment. In this case, instead of computing simultaneously the localization and the 3D map, the map is fixed and only the location and orientation of the camera is estimated, speeding-up the localization process. Figure 1.2 depicts a graphical example of the visual SLAM and vision-based localization with a prior 3D map problems.

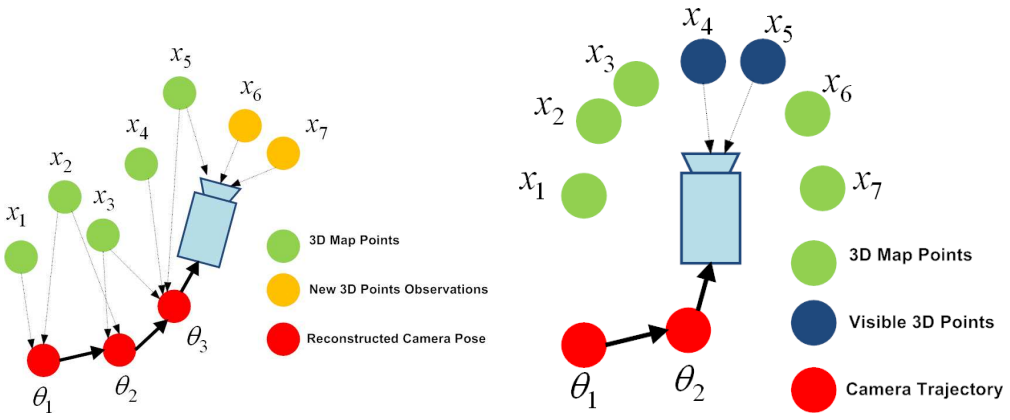


Figure 1.2: Graphical example of the problems of visual SLAM and vision-based localization with a prior map. (a) Visual SLAM: In this case the camera trajectory and a 3D map of the environment is estimated simultaneously, while the camera is discovering the environment. The new observations are incorporated in the previous map (b) Vision-based localization with a prior map: In this case, the map is fixed and the position of the camera in the map is updated by matching the visible map points and the image observations given a camera pose prior. Best viewed in color.

In most of 3D reconstruction frameworks, each 3D point is also associated with a descriptor vector that encodes local appearance information over an area of interest centered on the detected 2D image feature. Common feature detectors and descriptors are the Scale Invariant Feature Transform (SIFT) [Lowe, 2004] or Speeded Up Robust Features (SURF) [Bay et al., 2008]. For example, in one of the most successful projects about city-scale 3D reconstructions [Snavely et al., 2006], SIFT features are used both in the detection and the description steps. One of the main drawbacks of these kind of local image descriptors is the high dimensionality of the descriptors (e.g. 128 dimensions for SIFT). This high dimension can become a computational bottleneck in the matching step of certain vision-based localization applications, where we have to match a large map of 3D points and associated descriptors with respect to the detected 2D image features. Therefore, it is important to reduce the number of matching operations between descriptors and also to obtain low-dimensional descriptors robust to different image transformations such as changes in lighting, viewpoint, noise, image blur, etc.

One of the most computationally expensive steps in vision-based localization with large maps is data association, in which matching candidates between a large map of 3D points and 2D features are retrieved and then usually validated by geometric constraints using RANdom SAmple Consensus (RANSAC) [Fischler and Bolles, 1981]. For environments with highly repetitive textures, such as cities, the traditional methods mainly depend on the appearance information, which results in a very large number of matching candidates due to the ambiguities introduced by visually similar features [Schindler et al.,

2008]. Therefore, it is necessary to develop algorithms that can speed-up this data association step reducing the matching ambiguities and performing an efficient and fast data association. In this way, we can solve efficiently the pose estimation problem given a set of correspondences between 3D points and detected 2D image features.

In this thesis we present several computer vision algorithms towards an efficient and accurate visual SLAM and vision-based localization applications. Now, we summarize the main lines we want to contribute in this thesis:

- We present a stereo visual SLAM algorithm that can build accurate 3D reconstructions of the environment. For this purpose we combine stereo visual odometry [Nistér et al., 2004; Kaess et al., 2009] techniques and a local Bundle Adjustment (BA) [Triggs et al., 1999] procedure for refining simultaneously the motion and structure in a hierarchical way.

- Most of SLAM systems in the literature [Konolige and Agrawal, 2008; Mei et al., 2010] need to assume static features in the environment and that a dominant part of the scene changes only with the camera motion. As a result, these approaches are prone to failure in crowded scenes with many independently moving objects. Even though some of the outliers can be detected by geometric constraints validation, one needs to take special care about not introducing any outlier in the 3D reconstruction process or otherwise the estimated map and camera trajectory can diverge considerably from the real solution. In this thesis, we propose a stereo visual SLAM algorithm that can deal with moving objects in extremely crowded environments thanks to a dense scene flow [Vedula et al., 1999] representation.

- We study a novel family of image-based descriptors that are based on second-order multiscale gauge derivatives [ter Haar Romeny, 2003]. While the standard derivatives used to build a SURF or SIFT descriptor are all computed relative to a single or few chosen orientations, gauge derivatives are evaluated relative to the gradient direction at every pixel offering extra matching robustness due to the extra invariance offered by gauge coordinates. We present extensive experimental results on different standard local descriptors evaluations that show the benefits of our approach. In particular we show experimental results on image matching against different image transformations [Mikolajczyk and Schmid, 2005], 3D SfM applications [Brown et al., 2011], loop closure detection [Cummins and Newman, 2008] and visual image categorization [Csurka et al., 2004]. The obtained results show that our family of descriptors G-SURF can be used efficiently in 3D SfM applications such as for example vision-based localization or pure image matching experiments.

- In order to speed-up the data association process between a large map of 3D points and 2D features perceived by a new camera in vision-based applications, in this thesis we describe a novel memory based learning approach for predicting the visibility of 3D points given a prior camera pose. *Visibility prediction* exploits all the geometric relationships between camera poses and 3D map points in the prior 3D reconstruction. Then, during vision-based localization experiments we can speed-up tremendously the data association and pose estimation by predicting only the most highly visible 3D points given a prior on the camera pose. In this way, we can perform an extremely fast prediction of visible 3D points, yielding a correct data association and solving efficiently the pose estimation or localization problem given a prior 3D map of the environment.

- We validate our visual SLAM and vision-based localization algorithms considering different applications of great interest in robotics and computer vision. The first of the applications is vision-based localization for **humanoid robots** in indoor environments. The second one is related to visual SLAM and vision-based localization for the **visually impaired**. We performed several vision-based localization experiments given a prior 3D map in indoor office-like environments. In addition, we also validate our visual SLAM with moving objects detection in extremelly challenging environments with many independent moving objects such as inside the Atocha railway station (Madrid, Spain) or in the city center of Alcalá de Henares (Madrid, Spain) with real visually impaired users.

1.1 Organization of the Dissertation

The remainder of this dissertation is organized as follows: Related work is discussed separately in each of the five main chapters. Chapter 2 presents our stereo visual SLAM algorithm that provides an incremental and accurate 3D reconstruction of the environment, capable of dealing with outliers in dynamical scenarios. Chapter 3 introduces our family of local-invariant descriptors known as Gauge-SURF (G-SURF) that exhibit a higher recall compared with previous approaches. We show that these descriptors can be used efficiently in image matching problems (e.g. loop closure detection) and in SLAM or SfM applications. Chapter 4 describes our visibility learning framework for large-scale urban environments. In addition, we show experimental results of the algorithm considering large city-scale 3D reconstructions. Chapter 5 describes the vision-based localization algorithm for humanoid robots. Chapter 6 describes our visual SLAM and vision-based localization algorithms for visually impaired users in indoor environments. Finally, Chapter 7 concludes this dissertation summarizing the main contributions of this thesis and discussing future work directions.

Chapter 2

Visual Simultaneous Localization and Mapping

The great progress of robotics, computer vision and computing hardware in the last decades has increased the demand for localization and mapping applications. Simultaneous Localization and Mapping (SLAM) has a key role in robotics and computer vision applications. In computer vision, this problem is also known as Structure from Motion (SfM). The final goal of SLAM and SfM methods is obtaining an accurate and persistent 3D representation of the environment that can be used efficiently for long-term localization or navigation purposes.

Ever since the seminal work by Broida et al. in the early nineties [Broida et al., 1990; Broida and Chellappa, 1991], visual SLAM has captured the attention of researchers and the interest of using cameras as sensors has grown considerably due to mainly three reasons: cameras are cheaper than commonly used scan-lasers, they provide rich visual information about scene elements and they are easy to adapt for wearable systems. According to this, the range of SLAM based applications has spread to atypical robotic environments such as non-invasive surgery [Mountney et al., 2006; Grasa et al., 2011], augmented reality applications [Klein and Murray, 2007; Chekhlov et al., 2007] and road vehicle localization [Milford and Wyeth, 2008; Schleicher et al., 2009].

SLAM is of extreme importance in robotics, since robots need to build a map of the surrounding environment in order to perform autonomous navigation or interaction tasks. In addition, visual SLAM applications can be very interesting for visually impaired users, providing information about their current position and orientation and/or guiding them to their destination through diverse sensing modalities [Walker and Lindsay, 2006]. Moreover, vision systems can also provide scene understanding [Li et al., 2009] allowing visually impaired users to have a more effective navigation through space.

One of the main problems in SLAM is the integration of the new observations (landmarks and camera poses) into the whole optimization problem. In visual SLAM approaches, camera poses are usually represented by means of a 6-Degrees of Freedom (DoF) representation, three degrees for camera translation and three degrees for camera orientation. Landmarks usually describe 3D points of the environment that can be tracked successfully during several frames by means of visual features. Although a higher level structure representation such as lines or planes can be also used as landmarks and incorporated into the SLAM framework [Gee et al., 2008]. When new observations arrive, these new observations must be locally and globablly consistent with the previous reconstruction. However, in incremental SLAM or SfM approaches some drift is inevitably accumulated over time. This drift needs to be corrected by means of global relaxation or optimization

methods that take into account all the variables in the reconstruction. The accumulated drift can be only estimated when the camera re-visits an area that was previously mapped with a lower uncertainty. This situations can be identified by means of vision-based loop closure detection algorithms [Angeli et al., 2008; Cummins and Newman, 2008] that are usually based on pure appearance information and popular visual vocabularies schemes [Sivic and Zisserman, 2003; Nistér and Stewénius, 2006].

Nowadays in computer vision, Bundle Adjustment (BA) is the traditional method to optimize simultaneously all the camera parameters and 3D map points involved in the reconstruction. BA is a very popular and well-known technique used in computer vision, and in particular for SfM problems [Agarwal et al., 2010; Byröd and Åström, 2010]. A complete survey on BA methods can be found in [Triggs et al., 1999]. The main problem of BA-based methods is that their speed can be very low when the number of parameters is high, since for solving the full optimization problem it is necessary to perform the inversion of several linear systems whose size is proportional to the number of estimated parameters. Therefore, it is necessary to obtain BA-based solutions that can cope with large-scale environments. In this thesis, we propose to use a combination of local BA and global BA to obtain accurate 3D maps with respect to a global coordinate frame. For optimizing simultaneously the set of camera poses and 3D map points, we use the incremental local BA approach described in [Mouragnon et al., 2009]. This approach uses a sliding window BA of a fixed number of keyframes, optimizing only a subset of the camera poses and the 3D map points involved in the whole reconstruction. In this way, 3D points and camera poses are refined simultaneously through the sequence by means of local BA, and when a loop closure is detected, the residual error in the reconstruction can be corrected by means of global BA adding the loop closure constraints.

Most of SLAM systems in the literature [Konolige and Agrawal, 2008; Mei et al., 2010] assume static features in the environment. Even though some of the outliers can be detected by geometric constraints validation, one needs to take special care about not introducing any outlier in the SLAM process or otherwise the estimated map and camera trajectory can diverge considerably from the real solution. In this chapter, we will present a stereo visual SLAM algorithm that can deal with moving objects in extremely cluttered environments such as real-world crowded scenarios. This is possible, since by means of stereo vision we can obtain dense disparity maps (between the two images of the stereo rig) and dense 2D optical flow estimates (between two consecutive frames). Therefore, a dense scene flow [Vedula et al., 1999] description of the scene can be obtained, describing the 3D motion of the world points. With the 3D motion of the world points and the estimated camera motion, those 3D points that are located on moving objects can be detected and deleted from the SLAM process.

In this chapter we will describe the main components of our stereo visual SLAM system that builds an accurate 3D representation of the environment that can be used later for visibility learning and an efficient real-time vision-based localization. In the map computation process we dedicate more computational resources in order to obtain the best 3D map as possible for an efficient posterior real-time vision-based localization. Therefore, the map computation is run in an off-line or batch mode, although it can be easily adapted to real-time demands by means of some code optimization and the use of modern Graphic Processing Units (GPUs) that are widely used nowadays in computer vision problems as for example in [Newcombe and Davison, 2010; Lovegrove and Davison, 2010].

The remainder of the chapter is organized as follows: In Section 2.1 we will describe

the related work in monocular and stereo visual SLAM approaches and also some SLAM applications in dynamic environments. In Section 2.2 an overall overview of the main components of our stereo visual SLAM system is done. Then, in the following sections, each of the main components is explained in more detail. In this thesis we will validate the stereo visual SLAM system in two different applications: humanoid robots and visually impaired people. These two applications will be described in more detail including extensive experimental results in Chapters 5 and 6 respectively. Finally, some conclusiones and future work are described in Section 2.11.

2.1 Related Work

In this section, we will describe the related work about visual SLAM focusing on monocular and stereo-based approaches. In addition, we will also describe different approaches related to SLAM in dynamic environments.

2.1.1 Monocular SLAM

In the recent years, there has been much progress in vision-based SLAM. Among the different modalities, monocular SLAM is of special interest for several reasons. Visual SLAM with a single camera is more challenging than when using stereo vision where the 3D geometry of the world can be recovered more easily. In contrast, in monocular visual SLAM the 3D geometry must be inferred from multiple view images. This is because is not possible to recover the absolute scale of a 3D point due to the inherent observability problems in recovering 3D information from 2D projections using a single camera as the only sensor. However, monocular visual SLAM is still an active field of research. Satisfactory monocular visual SLAM solutions can have a big impact on many application domains, since a single camera is cheaper than a stereo rig, they are not so sensitive to calibration parameters as stereo cameras and can be more easily integrated into wearable systems or Advanced Driver Assistance Systems (ADAS).

Probably one the first monocular visual SLAM works is the one described by Broida et al. in the early nineties [Broida et al., 1990; Broida and Chellappa, 1991]. This seminal work describes a recursive algorithm to a sequence of images of a moving object to estimate both its structure and kinematics. The recursive estimation is done using an Iterated Extended Kalman Filter (IEKF) for the features and motion. Years later, one of the most successful real-time monocular SLAM systems named *MonoSLAM* was introduced by Davison et al. [2007]. In this approach, camera poses and an incremental sparse map of 3D landmarks are computed simultaneously using a standard Extended Kalman Filter (EKF) framework. Another important work is [Eade and Drummond, 2007], where the authors proposed a real-time monocular SLAM system for small office scale environments that comprises of different nodes combined in a larger graph that is fully optimized by non-linear optimization techniques. Ever since, EKF-SLAM strategies have been widely used and have been improved significantly in different aspects such as: undelayed initialization of features thanks to an inverse depth parametrization of landmarks [Civera et al., 2008], automatic re-localization [Williams et al., 2007] or large-scale mapping applications [Clemente et al., 2007].

The main drawback of EKF-based approaches is the limited number of 3D points that can be tracked, apart from divergence from the true solution due to linearization errors. As shown in several works [Dellaert and Kaess, 2006; Strasdat et al., 2010a] non-linear

optimization techniques such as BA or Smoothing and Mapping (SAM) are superior in terms of accuracy to filtering based methods, and allow to track many hundreds of features between consecutive frames. According to this, several monocular SLAM approaches based on non-linear optimization techniques such as BA have been presented in the recent years, outperforming considerably the performance and accuracy of EKF-based approaches. Some examples of these works are as follows: In [Royer et al., 2005], the authors presented one of the first monocular SLAM systems based on BA for large-scale environments. Then, Klein and Murray [2007] presented the highly popular Parallel Tracking and Mapping (PTAM) approach that combines local and global BA for map optimization. The PTAM approach was designed for small Augmented Reality (AR) applications. Recently, Strasdat et al. [2010b] proposed a PTAM-inspired keyframe SLAM system for large-scale monocular applications and scale drift optimization.

2.1.2 Stereo SLAM

As mentioned before, one of the main limitations of monocular SLAM approaches, is recovering the true scale of a 3D scene, due to the observability problems in recovering 3D information from 2D projections. Stereo cameras are an appealing alternative, since they directly provide the scale of a point using the information from the two camera views. Therefore, most successful and accurate visual SLAM systems are based on stereo vision approaches [Konolige and Agrawal, 2008; Mei et al., 2010].

EKF-SLAM strategies have been also applied with success to stereo vision sensors in large-scale environments, as for example in the following two works [Paz et al., 2008; Schleicher et al., 2009]. These two works adapted in a different way the MonoSLAM approach [Davison et al., 2007] for the stereo vision setup. In order to extend EKF solutions to large-scale environments, both works consider submapping techniques that divide the whole map into different local conditionally independent submaps.

In [Paz et al., 2008], the authors proposed a 6-DoF Stereo EKF-SLAM system with a stereo camera carried in hand for large indoor and outdoor environments. The inverse depth parametrization proposed in [Civera et al., 2008] for the MonoSLAM approach is adapted to the stereo SLAM version. In this way, landmarks provide distance and orientation information. Point features are extracted from the images and are classified as 3D features if the disparity is higher than a fixed threshold, or stored as inverse depth features otherwise. Their visual SLAM algorithm generates conditionally independent local maps and finally, the full map is obtained using a conditionally independent divide and conquer algorithm, which allows constant time operation most of the time [Paz et al., 2007].

In contrast, Schleicher et al. [2009] proposed a real-time hierarchical topological-metric representation of the environment for road-vehicle SLAM applications by combining stereo vision and Global Positioning System (GPS) information. The metric level is based on stereo EKF-SLAM techniques, whereas the topological level is based on the use of the appearance-based Scale Invariant Feature Transform (SIFT) [Lowe, 2004] *fingerprints* that identify each node in terms of appearance in the whole topological graph. When a loop closure is detected, a global rectification of the map in 2D is perfomed by means of the MultiLevel Relaxation (MLR) approach [Frese et al., 2008]. Figure 2.1 depicts the system architecture of Schleicher et al. mounted on a commercial car, fusing the information from a stereo vision system and a low cost GPS.

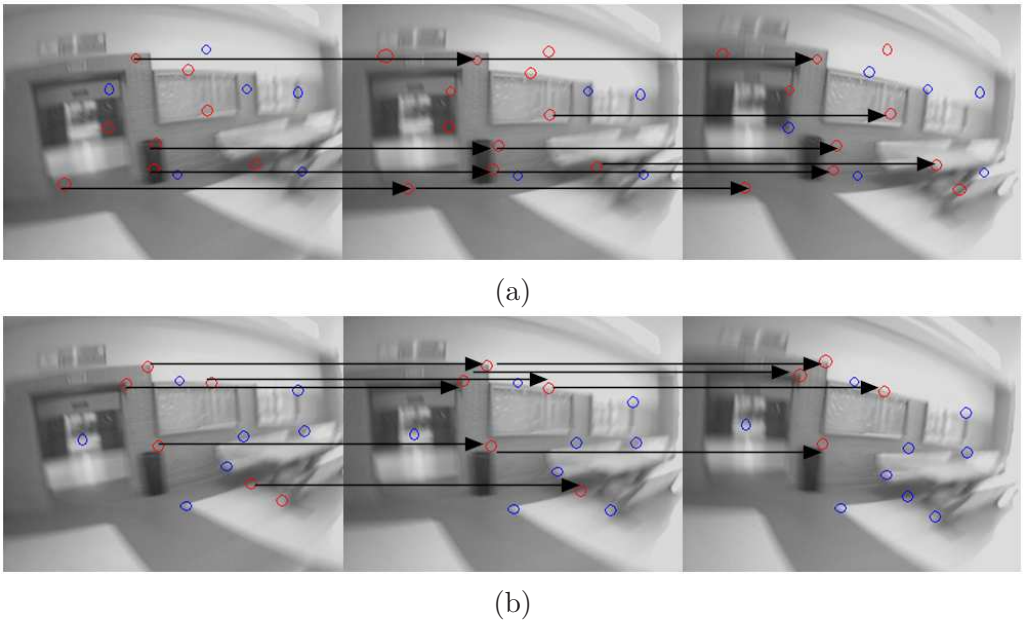
Another important drawback of standard visual EKF-SLAM techniques is the assumption of a general camera motion model. Usually this motion model has been implemented

Figure 2.1: System architecture mounted on a commercial car. (Top left) Stereo vision system and low-cost GPS. (Top right) Ground truth RTK-GPS used for system validation.

in the literature as a constant linear and angular velocity model [Davison et al., 2007]. Because of this, most approaches cannot deal with sudden camera movements, causing them to lose accurate camera pose estimates and leading to a corrupted 3D scene map. For example, the mentioned stereo based approaches [Paz et al., 2008; Schleicher et al., 2009] cannot handle sudden camera motions since both of them use a constant velocity model. We hypothesize that results can be improved considerably if a better motion model is used.

In order to cope with sudden camera motions in visual EKF-SLAM frameworks, the use of visual odometry priors was proposed in [Alcantarilla et al., 2010a] for the stereo vision case and in [Williams and Reid, 2010] for the monocular version. Visual odometry priors considers how well-known visual odometry schemes [Nistér et al., 2004; Kaess et al., 2009] can be used in conjuction with EKF-SLAM frameworks, in order to provide accurate camera pose priors that can be incorporated into the EKF process. For example, Figure 2.2 depicts a comparison between a constant velocity model (a) and visual odometry priors (b) from the work described in [Alcantarilla et al., 2010a]. The figure depicts feature tracking in three consecutive frames where a fast camera rotation is performed. Ellipses in red color means that the feature has been correctly matched (high 2D templates correlation value) whereas blue color means that the feature has not been matched correctly. As can be observed, with visual odometry priors the feature tracking search is correctly performed whereas with the constant velocity model the search areas are not in the correct position due to the fast camera rotation yielding bad features estimates that corrupt the pose and the map.

In a similar way as monocular approaches, stereo visual SLAM techniques based on BA also outperform considerably the performance and accuracy of EKF-based approaches. By means of stereo vision, powerful non-linear optimization techniques such as BA and highly engineered SfM pipelines, stereo visual SLAM systems can obtain precisions down to a few meters over distances of a few kilometers. Probably, the two most representative works in the state of the art of stereo visual SLAM techniques are the works of Konolige



(a)

(b)

Figure 2.2: Fast camera rotation in three consecutive frames. (a) Constant velocity motion model (b) Visual odometry priors. Best viewed in color.

and Agrawal [2008] and Mei et al. [Mei et al., 2010].

In [Konolige and Agrawal, 2008], the authors presented a real-time visual SLAM approach that includes accurate stereo visual odometry and global map rectification. In order to scale with large-scale environments, the number of keyframes in the map is limited building a *skeleton* or a reduced form of the whole graph. The obtained constraints from measurements are transformed into probability distributions over keyframe transformations. Then, the skeleton system is solved in a non-linear minimization approach and the rest of the graph is updated by means of the previous probability distributions over keyframe transformations. In contrast, Mei et al. [2010] proposed a stereo visual SLAM system using a relative map representation. Their system, named Relative Simultaneous Localization and Mapping (RSLAM) allows to map large-scale environments in real-time by means of a relative map representation and high performance appearance-based loop closure detection. Their relative map representation allows to perform relative BA updates in a local neighbourhood of a certain camera pose. Recently in [Clipp et al., 2010], the authors presented a highly-optimized stereo visual SLAM system that makes extensive use of parallelism both on the graphics processor and through multiple CPU threads.

2.1.3 SLAM in Dynamic Environments

Almost all the works in stereo visual SLAM need to compute a good initialization of the structure and motion. This initialization needs to be close to the real solution, otherwise BA or EKF methods will not converge to an optimal solution. In visual SLAM approaches, this initialization is normally obtained by means of visual odometry [Nistér et al., 2004; Kaess et al., 2009] or other pose estimation procedures [Mei et al., 2008]. Visual odometry is as its heart a pose estimation problem and involves the tracking of features between (at least) two consecutive frames. Then, the set of candidate matches is usually validated by means of a RANdom SAmple Consensus (RANSAC) framework [Bolles and Fischler, 1981].

Visual odometry and most of SLAM systems in the literature assume static features in the environment and that a dominant part of the scene changes only with the camera motion. However, most of those systems have not been tested under realistic highly crowded scenes. As a result, these approaches are prone to failure in crowded scenes with many independent moving objects. Even though some of the outliers can be detected by geometric constraints validation or RANSAC-type frameworks, in SLAM one needs to take special care about not introducing outliers in the reconstruction process or otherwise the estimated map and camera trajectory can diverge considerably from the real solution.

In robotics, there have been several approaches related to SLAM in dynamic environments, but most of them are focused on range data [Hähnel et al., 2003; Wang et al., 2007; Bibby and Reid, 2007]. Regarding, vision-based approaches there have been only limited attempts. In [Wangsiripitak and Murray, 2009], the authors presented an extended version of the MonoSLAM algorithm that includes a 3D object tracker. By using the information from the object tracker, features that belong to moving objects can be detected and deleted from the SLAM estimation. However, their method is only useful for indoor small workspace scenarios and only simple 3D objects with a prior shape knowledge can be tracked. Ess et al. [2009] showed impressive results of pedestrian detection and tracking from a movable platform using stereo vision. In this way, visual odometry estimation can be benefited from the detection of moving objects.

Another important advantage of stereo vision with respect to monocular systems, is that we can exploit the information from four images at once, obtaining dense disparity maps and dense 2D optical flow [Lucas and Kanade, 1981] estimates between two consecutive frames. Since for every detected point of interest we know its 3D position (with respect to the camera coordinate frame) and the associated 2D optical flow, a dense scene flow [Vedula et al., 1999] description of the scene can be obtained, describing the 3D motion of the world points. Hence, by knowing the camera motion and the motion of 3D world points, 3D moving objects can be identified and eliminated from the SLAM process. In this Chapter, we will describe how by means of dense scene flow techniques, moving objects can be detected and deleted from the SLAM process, yielding a more robust 3D reconstruction.

2.2 Stereo Visual SLAM System Overview

In this section, we will explain a general overview of the main components of our visual stereo SLAM system. Figure 2.3 depicts an overview of the main components of our system. Notice, that in this work we are mainly interested in using stereo visual SLAM for computing a robust and accurate 3D map, that will be used later for a posterior visibility learning and long-term real-time vision-based localization.

Our stereo visual SLAM system is based on different components. Firstly, in Section 2.3 we will describe the steps for performing a calibration of the stereo rig and a posterior rectification of the images. This calibration and rectification process is a fundamental step for obtaining accurate 3D measurements. In Section 2.4, we will analyze the uncertainty in the 3D measurements and depth estimates that we can obtain considering a calibrated stereo rig. Then in Section 2.5 we will describe the main components of the stereo visual odometry algorithm, which plays a key role in the whole visual SLAM system. By means of visual odometry, we can estimate the egomotion of a moving camera by tracking features between two consecutive frames. In this way, we can obtain accurate priors on the camera poses and also on the location of the map 3D points. Both camera

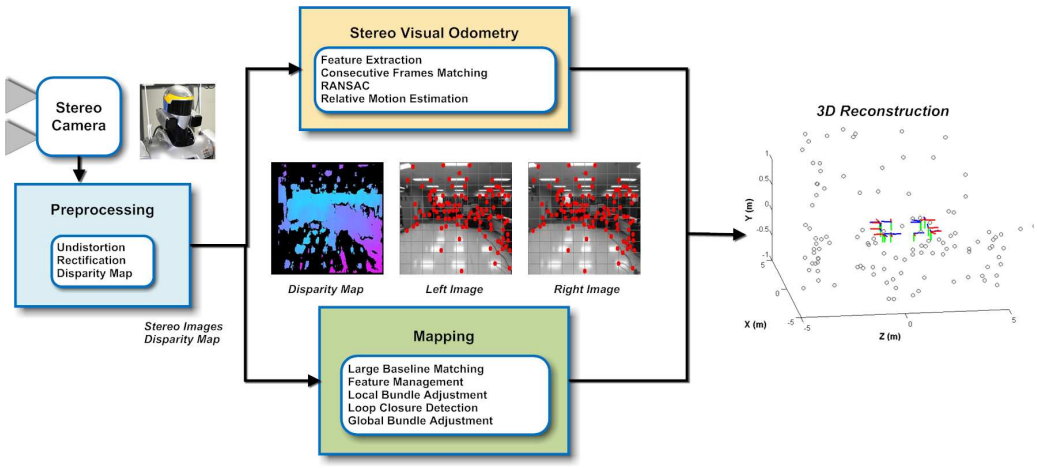


Figure 2.3: Stereo visual SLAM system overview: Firstly, we undistort and rectify the stereo images given the stereo rig calibration parameters. Then, for the left image of the stereo pair, we detect 2D features of interest and an associated descriptor vector that encodes appearance information. By performing stereo visual odometry between two consecutive frames, we estimate the relative camera motion between frames. Finally, this relative camera motion is translated into a global coordinate frame, and a set of selected camera poses (keyframes) and 3D points are refined in a local bundle adjustment procedure.

poses and 3D map points will be refined later in a BA process. Section 2.6 and 2.7 describe the local BA and the map management modules respectively, whereas loop closure and global map optimization are discussed in Section 2.8. In Section 2.9, we will describe a method to detect possible moving objects based on scene flow techniques. This method prevents the SLAM system from bad visual odometry estimates and from avoiding adding erroneous observations in the whole estimation process. In addition, in Section 2.10 we describe how by means of the described SLAM pipeline, we can obtain dense mapping representations of the environment without high computational demands. Even though, the final map that will be used later for vision-based localization applications is based on highly accurate sparse 3D points, a dense scene representation of the environment can be of interest in different applications such as for example navigation, planning or just simply 3D world modelling for visualization purposes.

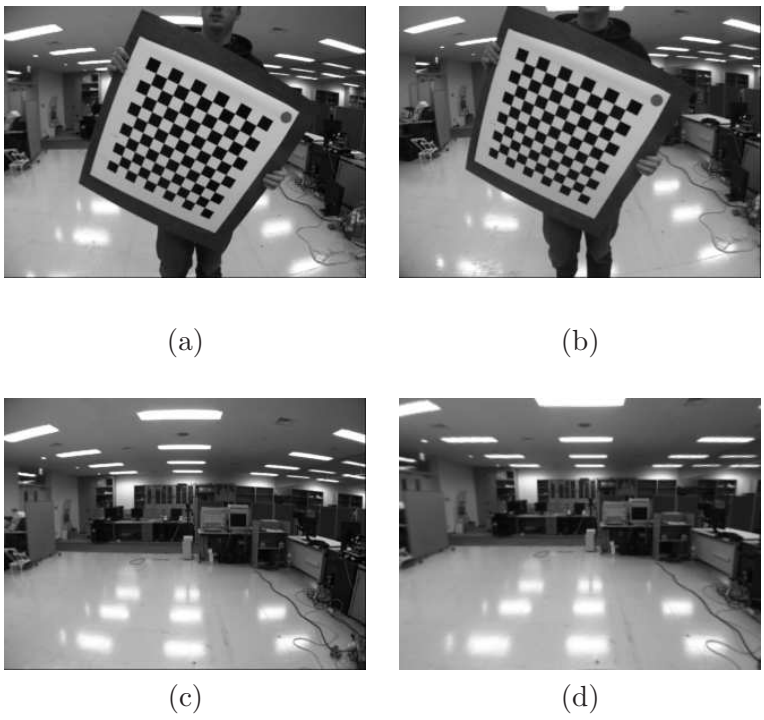
2.3 Stereo Rig Calibration and Rectification

In order to obtain accurate localization and mapping results, a prior stereo rig calibration process is necessary. The stereo rig calibration problem involves the estimation of the *intrinsic* parameters and *distortion* parameters of each of the cameras, and the *extrinsic* parameters (rotation, translation) between cameras. In this thesis, we used a chessboard pattern of known dimensions as a calibration object, and around twenty pairs of images were taken for the calibration. Figures 2.4(a,b) depict one example of a pair of images used in the stereo rig calibration process. Both cameras were calibrated independently using the Camera Calibration Toolbox for Matlab [Bouguet, 2008b]. In this way, we can obtain the intrinsics calibration matrix for each of the cameras:

$$\mathbf{K} = \begin{pmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.1}$$

where f_x and f_y are the focal lengths and (u_0, v_0) are the coordinates of the principal point of the camera. Radial (k_1, k_2) and tangential (p_1, p_2) distortion parameters are modelled by means of polynomial approximations [Heikkila and Silven, 1997]. After the calibration of each of the cameras, the extrinsics parameters of the stereo rig are estimated. The extrinsics parameters comprise of a rotation matrix R_{LR} and a translation vector T_{LR} between the left and right cameras of the stereo rig.

Once we have obtained the intrinsics and extrinsics of the stereo rig, we can correct the distortion of the images and perform stereo rectification [Hartley, 1999]. Stereo rectification simplifies considerably the stereo correspondences problem and allows to compute dense disparity or depth maps. We perform stereo rectification by means of the algorithm described in [Bouguet, 2008a]. Bouguet's rectification algorithm minimizes the amount of change reprojection produces for each of the two images while maximizing common viewing area between both images. Figure 2.4(c) depicts an example of a left raw image acquired by the camera and (d) depicts the resulting image after rectification and distortion correction.



(a) (b)

(c) (d)

Figure 2.4: Stereo Rig Calibration. One example of a pair of images of the chessboard pattern used for the calibration: (a) Left image (b) Right image. (c) An example of a raw image acquired by the robot during localization and the respective rectified image (d).

After stereo rectification, we obtain a new camera matrix $\mathbf{K}$, where the left and right camera have the same focal lengths f and principal point (u_0, v_0). The rotation matrix between cameras R_{LR} is the identity matrix, and the translation vector T_{LR} encodes the baseline B of the rectified stereo rig. Now, considering an ideal stereo system, the depth of one 3D point can be determined by means of the following equation:

$$Z \;=\; f \cdot \frac{B}{u_R - u_L} \;=\; f \cdot \frac{B}{d_u} \tag{2.2}$$

where d_u is the horizontal disparity or the difference in pixels between the horizontal image projections of the same point in the right and left images. Given the depth of the

3D point Z, and the stereo image projections of the same point in both images (u_L, u_R, v) (notice that in a rectified stereo $v_L = v_R = v$) the rest of the coordinates of the 3D point with respect to the camera can be determined as:

$$X = \frac{Z \cdot (u_L - u_0)}{f} \tag{2.3}$$

$$Y = \frac{Z \cdot (v - v_0)}{f} \tag{2.4}$$

2.3.1 Pre-Processing

Firstly, the raw stereo images captured by the stereo camera are rectified and undistorted in a pre-processing stage. The undistortion and rectification is performed in a fast way by means of using precomputed Look up Tables (LUTs). This is possible, since stereo rig calibration parameters are obtained in a previous offline calibration process as described in Section 2.3. Once, the images are undistorted and rectified, the disparity map is computed by the method proposed in [Konolige, 1997].

The stereo rectification and disparity map computation can be on ocassions a computational burden for some limited computing hardware applications. With stereo vision, it is necessary to undistort the images and to find stereo correspondences between the left and the right view for every processed frame. However, there are some fast stereo implementations for obtaining dense disparity maps such as [Hirschmuller, 2006; Tola et al., 2010] that are amenable to GPUs implementations. In addition, many commercial stereo sensors provide disparity map and rectification implementations on-chip. One example is the popular Bumblebee2 stereo camera sold by Point Grey Research [1]. This commercial stereo rig provides highly accurate camera calibration parameters and also stereo rectification and dense depth map generation on-chip.

2.4 Analysis of Stereo Vision Errors and Uncertainties

In this section, we will analyze the errors and uncertainties in the 3D reconstruction that we can expect using a calibrated stereo rig. The reconstructed 3D scene from a calibrated stereo camera is error-prone due to measurement noise. According to the stereo geometry equations described in Section 2.3, we can obtain expressions for the uncertainty on the location of a reconstructed 3D point from a calibrated stereo rig. The resulting error of the 3D point position can be computed by means of linear error propagation. The covariance $\mathbf{P_{3D}}$ of the location of a 3D point $h_i = (x, y, z)^t$ with respect to the camera coordinate frame can be computed as follows:

$$\mathbf{P_{3D}}_{[3,3]} = \mathbf{J_{3D}} \cdot \mathbf{S_i} \cdot \mathbf{J_{3D}}^t \tag{2.5}$$

where $\mathbf{J_{3D}}$ is the Jacobian of the 3D point location with respect to the measurement paremeters vector $\mathbf{U} = (u_L, u_R, v)^t$, and $\mathbf{S_i}$ is the measurement noise matrix. This matrix measures the uncertainties in the location of a point in the image plane for both the left and right camera views. We will consider this matrix to be a diagonal matrix with standard deviations of ± 1 pixels. The Jacobian $\mathbf{J_{3D}}$ can be derived as follows:

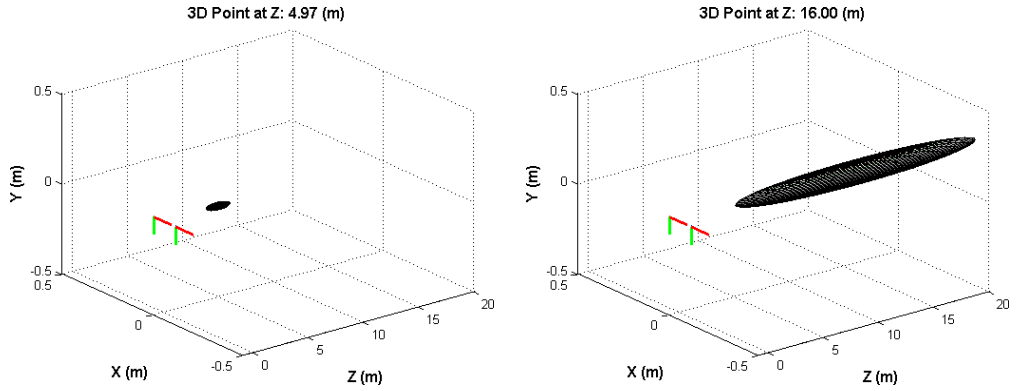
[1] For more information, please check: http://www.ptgrey.com/products/stereo.asp

$$\mathbf{J_{3D}}_{[3,3]} = \frac{\partial h_i}{\partial \mathbf{U}} = \begin{pmatrix} \frac{\partial x}{\partial u_L} & \frac{\partial x}{\partial u_R} & \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial u_L} & \frac{\partial y}{\partial u_R} & \frac{\partial y}{\partial v} \\ \frac{\partial z}{\partial u_L} & \frac{\partial z}{\partial u_R} & \frac{\partial z}{\partial v} \end{pmatrix} \tag{2.6}$$

From the stereo geometry Equations 2.2, 2.3, 2.4, we can obtain the expressions for the partial derivatives that are involved in the previous Jacobian expression:

$$\mathbf{J_{3D}}_{[3,3]} = \frac{\partial h_i}{\partial \mathbf{U}} = \begin{pmatrix} \frac{B}{u_R-u_L} + \frac{B\cdot(u_L-u_0)}{(u_R-u_L)^2} & -\frac{B\cdot(u_L-u_0)}{(u_R-u_L)^2} & 0 \\ \frac{B\cdot(v-v_0)}{(u_R-u_L)^2} & -\frac{B\cdot(v-v_0)}{(u_R-u_L)^2} & -\frac{B\cdot(v-v_0)}{(u_R-u_L)^2} \\ \frac{f\cdot B}{(u_R-u_L)^2} & -\frac{f\cdot B}{(u_R-u_L)^2} & 0 \end{pmatrix} \tag{2.7}$$

From the covariance $\mathbf{P_{3D}}$ of the location of a 3D point (Equation 2.5) we can obtain an estimation of the 3D error in the reconstruction from the resulting covariance ellipsoids. This error grows quadratically with the depth of the 3D point. For example, Figure 2.5 depicts the associated covariance ellipsoids for two 3D points located at a depth of 4.97 m (a) and 16.00 m (b), considering a stereo rig of 12 cm baseline. As it can be observed, for those points that are located close to the stereo rig the 3D error due to measurement noise is low, but as long as the depth increases the error grows quadratically with the depth of the 3D point.



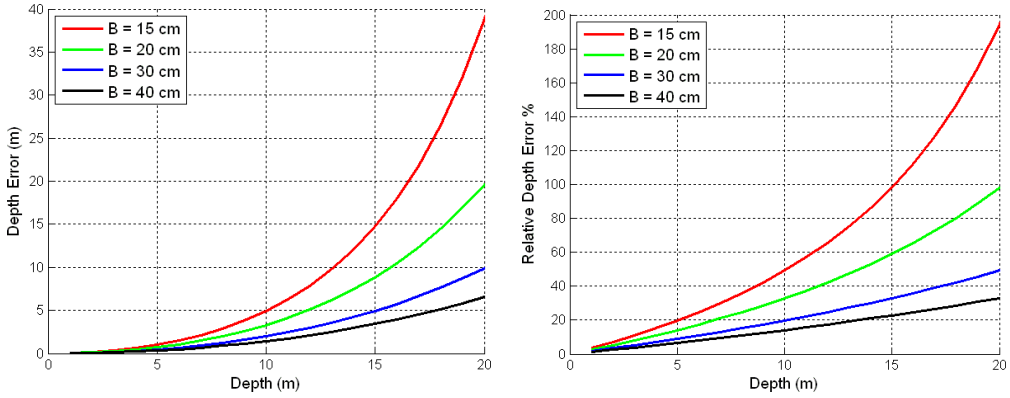
(a) (b)

Figure 2.5: Associated covariance ellipsoids for two 3D points considering a stereo rig of 12 cm baseline. (a) 3D point located at a depth distance of 4.97 m (b) 3D point located at a depth distance of 16.00 m

Now, we will show another representation of the 3D error from stereo reconstructions. Given the baseline B of the stereo rig, the focal length in pixels f and the image size (width (W), height (H)), we can estimate the stereo error from the maximum disparity $d_{uMAX} = W - 1$ (minimum depth) to the minimum disparity $d_{uMIN} = 1$ in incremental steps of one pixel as follows:

$$\Delta Z_i = Z_i - Z_{i-1} = f \cdot B \left(\frac{1}{d_{u_i} - 1} - \frac{1}{d_{u_i}} \right) = f \cdot B \cdot \frac{1}{d_{u_i}^2 - d_{u_i}} \tag{2.8}$$

Equation 2.8 shows the relationship between the depth accuracy and stereo rig parameters f, B and image size (W,H). Figure 2.6 depicts the depth accuracy for different stereo baselines B considering fixed focal length $f = 202$ pixels and image size $(W = 320, H = 240)$.



(a) (b)

Figure 2.6: (a) Absolute and (b) relative depth estimation errors for a stereo rig considering a focal length $f = 202$ pixels and image size 320×240, for different stereo rig baselines. Best viewed in color.

The type of graphs shown in Figure 2.6 have been proposed in [Llorca et al., 2010] to show the errors for depth estimates using a stereo rig. As it can be observed, depending on the baseline, the error in the depth estimate can be very high. In general, the higher the baseline the lower the error in the depth estimation. For example, for a typical wearable stereo rig device with a baseline of 15 cm, the error in depth if we try to estimate the 3D coordinates of a point located at a real distance of 10 m from the stereo rig, the relative error in depth ΔZ will be higher than 40% or more than 15 m in absolute terms.

Even though we can obtain high 3D reconstruction errors for some 3D points, especially for the ones that are located far away from the stereo rig, in visual SLAM approaches these 3D errors are reduced by using the information from the measurements of the points from multiple view images. BA optimizes simultaneously the structure and motion parameters that are involved in the reconstruction, minimizing a defined cost function which usually comprises of the squared difference between the 2D reprojections of the 3D map points onto the different camera views and the obtained 2D measurements. The BA procedure will be explained in more detail in Section 2.6.

2.5 Stereo Visual Odometry

Visual odometry [Nistér et al., 2004; Kaess et al., 2009] is as its heart a pose estimation problem, that allows to estimate the relative camera motion between two consecutive frames. In addition, visual odometry can be implemented very efficiently in real-time and can be used to obtain good priors for the camera poses and 3D points that can be optimized later in a BA procedure.

We estimate the relative camera motion by matching detected features between two consecutive frames. Features are detected by means of the widely used Harris corner detector [Harris and Stephens, 1988] at different scale levels. We detect features only for the left image of the stereo pair. Then, we find the correspondences of the 2D features

in the right image by accessing the disparity map and compute the 3D coordinates of the point by means of the stereo geometry equations (see Equations 2.2, 2.3, 2.4). Finally, we have a set of stereo features $\mathcal{F}_t = \{(u_L, u_R, v)_i\}$, where (u_L, v) is the location of the feature in the left image and (u_R, v) is the corresponding location in the right image. In addition, we also store for each stereo feature $\mathcal{F}_t$ the 3D coordinates of the reconstructed point h_i with respect to the camera coordinate frame at that time instant t.

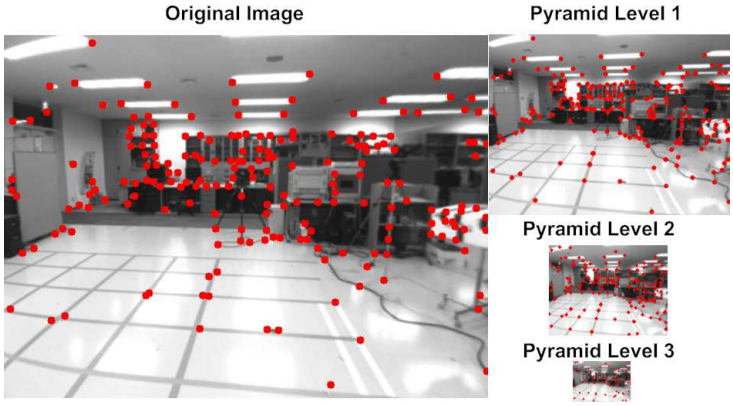


Figure 2.7: Multiscale Harris corner detector: From the original image, we compute three different pyramid levels and perform Harris corner detector at each pyramid level. Best viewed in color.

For each detected 2D feature in the left image we also extract a descriptor vector that encodes the appearance information of a local area centered on the point of interest. Similar to Speeded Up Robust Features (SURF) [Bay et al., 2008] descriptors, for a detected feature at a certain scale, we compute a unitary descriptor vector. This descriptor vector is computed in a very fast way thanks to the use of integral images [Viola and Jones, 2004]. Our descriptors named Gauge-SURF (G-SURF) are based on second-order multiscale gauge derivatives, and as will be shown in Chapter 3, exhibit much better performance and recall than other state of the art descriptors [Lowe, 2004; Bay et al., 2008]. We use the upright version of the descriptors (no invariance to rotation) since for common visual odometry and visual SLAM approaches, rotation invariance is not necessary. In addition, as found in [Gil et al., 2009], the upright version of the descriptors performs best in scenarios where the camera only rotates around its vertical axis, which is often the typical case of visual odometry applications.

Once we have computed the features descriptors, we find the set of putatives between the stereo features from the current frame $\mathcal{F}_t$ and the previous one $\mathcal{F}_{t-1}$ by matching their associated list of descriptors vectors. In order to reduce matching ambiguities we only try to match descriptors between consecutive frames in a circular area of fixed radius centered on the detected feature in the current frame. In our experiments, a fixed radius of 15 pixels is enough for finding the set of putatives between two consecutive frames.

After finding the set of putatives between two consecutive frames we estimate the relative camera motion using a standard two-point algorithm in a RANSAC setting by minimizing the following cost function:

$$\underset{R^t_{t-1}, \mathbf{t}^t_{t-1}}{\arg\min} \sum_i \left\| z_{i,t} - \Pi\left(R^t_{t-1}, \mathbf{t}^t_{t-1}, h_{i,t-1}\right) \right\|_2 \tag{2.9}$$

where $z_{i,t} = \{(u_L, u_R, v)_i\}$ are the set of 2D measurements of a stereo feature at time t and Π is a function that projects a 3D point $h_{i,t-1}$ (referenced to the camera coordinate

frame at time $t-1$) to the image coordinate frame at time t. This projection function Π involves a rotation R_{t-1}^{t} and a translation $\mathbf{t}_{t-1}^{t}$ of 3D points between both coordinate frames and a projection onto the image plane by means of the stereo rig calibration parameters. The resulting relative camera motion is transformed to a global coordinate frame (usually referenced to the first frame of the sequence) and then is used by the mapping management module. We use the Levenberg-Marquardt (LM) [Marquardt, 1963] algorithm for all the non-linear optimizations.

2.6 Bundle Adjustment

By means of stereo visual odometry, we can estimate the relative camera motion between two consecutive frames. When the accumulated motion in translation or rotation is higher than a fixed threshold we decide to create a new keyframe. This keyframe, will be optimized later in an incremental local BA procedure. While initializing a new keyframe, we store its pose with respect to a global coordinate frame, the detected 2D features, associated appearance descriptors and respective 3D points location. In addition, we also store its visibility information, i.e. the list of 3D points that are visible from that keyframe. This information will be used later in the visibility learning procedure as will be explained in Chapter 4. In our experiments, we add a new keyframe when the accumulated translation or rotation are higher than 0.25 m and 15° respectively.

BA provides an iterative optimization of the camera poses and 3D points involved in the reconstruction. Roughly speaking, BA is a non-linear least squares problem and consists in the minimization of the sum of squared reprojection errors. Furthermore, if the noise in the image error is Gaussian, then BA is the *Maximum Likelihood Estimator* yielding the optimal least squares solution. In general, BA has a $\Theta\left(N^3\right)$ time complexity, being N the number of variables involved in the optimization problem [Hartley and Zisserman, 2000]. This time complexity becomes a computational bottleneck for incremental SfM or visual SLAM approaches that have real-time constraints. Therefore another alternatives that can reduce this time complexety are necessary. In addition, it is also important to have an initial estimate of the parameters close to the real solution, or in other case BA can diverge from an optimal solution [Schweighofer and Pinz, 2006]. In our work we obtain robust initial estimates of the reconstruction by means of the stereo visual odometry algorithm described in Section 2.5.

For optimizing simultaneously the set of camera poses and 3D points in real-time, we use the incremental local BA approach described in [Mouragnon et al., 2009]. We use a sliding window over the last N_k keyframes, optimizing only the camera parameters of the last n_k cameras. With respect to the 3D points, only those 3D points that are visible in the last n_k cameras are optimized. In this way, 3D points and camera poses are refined simultaneously through the sequence. The optimization process is a sparse LM minimization of the cost function $f_{t_i}\left(\Theta_{t_i}, X_{t_i}\right)$, where Θ_{t_i} and X_{t_i} are respectively the motion parameters (cameras translation and rotation) and the 3D structure for the time instant t_i. The idea of the local BA is to reduce the complexity by optimizing only a subset of the keyframes and 3D points from the whole 3D reconstruction. Only the motion parameters of the last n_k cameras are optimized at each local BA execution taking into account the 2D reprojections in the last N_k keyframes, being $N_k \geq n_k$. Thus, $\Theta_{t_i} = \{\theta_{i-n_k+1} \ldots \theta_i\}$ and X_{t_i} contains all the 3D points projected on cameras Θ_{t_i}. The cost function f_{t_i} is the sum of the reprojection errors of the 3D point cloud X_{t_i} in the last frames from θ_{i-N_k+1} to θ_i:

$$\underset{\theta_{i-N_k+1}\dots\theta_i, X_{ti}}{\arg\min} f_{t_i}\left(\Theta_{t_i}, X_{t_i}\right) = \underset{\theta_{i-N_k+1}\dots\theta_i, X_{ti}}{\arg\min} \sum_{\theta_k \in \left\{\theta_{i-N_k+1}\dots\theta_i\right\}} \sum_{x_j \in X_{t_i}} \left\|\epsilon_j^k\right\|_2 \tag{2.10}$$

where $\left\|\epsilon_j^k\right\|_2$ is the square of the Euclidean distance between the estimated projections of the 3D point x_j through the camera pose θ_k and the observed stereo measurements $z_{j,k} = (u_L, u_R, v)$ from the camera pose θ_k. Figure 2.8 depicts a graphical local BA example when a new camera pose is added.

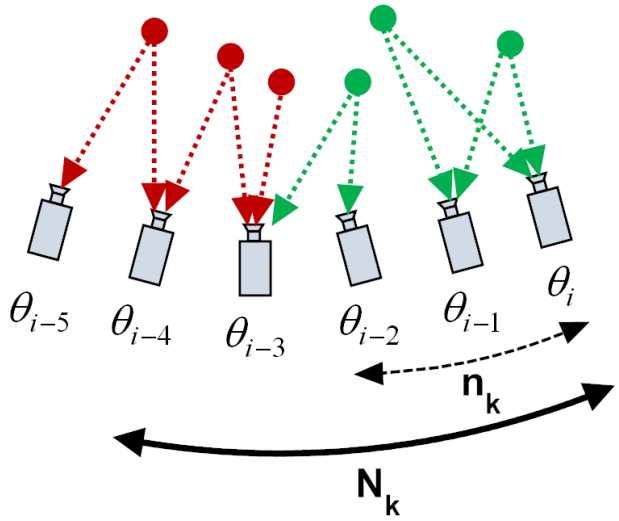


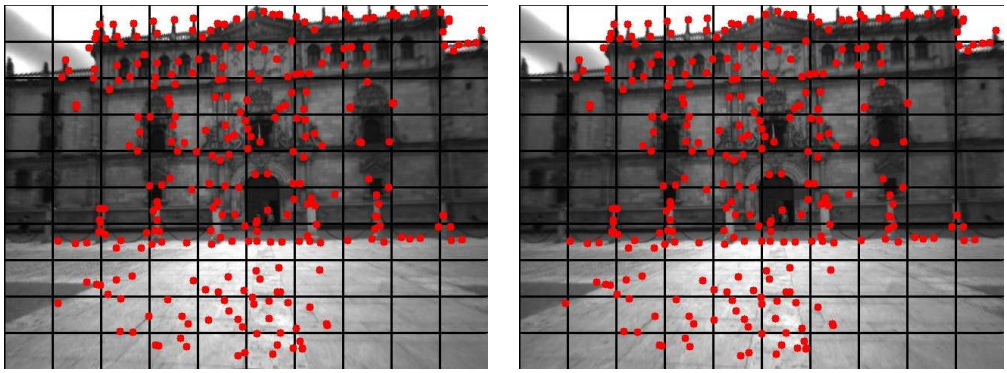
Figure 2.8: Local Bundle Adjustment when the new camera pose θ_i is added. Only the last n_k cameras are optimized in a sliding window of N_k cameras. Regarding the structure, only those 3D points that are visible by the last n_k cameras (in green) are optimized whereas the rest of 3D points in the reconstruction (in red) are fixed. Best viewed in color.

Optimal values for the involved parameters in the sliding window BA n_k, N_k are typically $n_k = 3$ and $N_k = 10$ (see [Mouragnon et al., 2009] for more details). In this work, we use the Sparse Bundle Adjustment (SBA) package [Lourakis and Argyros, 2009] as the basis for our local BA implementation. SBA exploits the inherent sparsity structure of the BA problem and is widely used in the computer vision community [Snavely et al., 2006; Agarwal et al., 2009].

2.7 Map Management

In order to build the map incrementally, we need to define a criteria for adding new features and deleting those ones whose tracking was poor during previous frames. We perform an *intelligent management* of features into the map in order to produce an equal distribution of feature locations over the image. Figure 2.9 depicts an example of the two images from the stereo camera and the tracked features. The left image is divided in a rectangular grid of 10×10 cells, and stereo correspondences are found in the right image by means of the disparity map. To ensure an equal distribution of feature locations over the image, only 5 features can exist on each grid cell. While adding a new feature to the map, we also store its associated appearance descriptor and 3D point location. Then, we try to match the feature descriptor against detected new 2D features on a new keyframe by matching their associated descriptors in a high probability search area. In this way,

we can create for a map element, *feature tracks* that contain the information of the 2D measurements of the feature (both in left and right views) in several keyframes. Then, this information is used as an input in the local BA procedure. Features are deleted from the map when the mean reprojection error in the 3D reconstruction is higher than a fixed threshold (e.g. 3 pixels).



(a) (b)

Figure 2.9: Points detected using a stereo camera. The images are divided in a rectangular grid of 10×10 cells. To produce an equal distribution of feature locations over the image, we only allow 5 features per cell. (a) Left Image (b) Right Image. Best viewed in color.

When adding a new feature into the map, we use the extracted set of appearance descriptors from the visual odometry module. The choice of a proper descriptor dimension highly depends on the application of interest. In Chapter 3 we will evaluate the matching capabilities of our G-SURF descriptors considering different dimensions. Notice here, that we are mainly interested in using the stereo visual SLAM system for a posterior robust and fast vision-based localization. For example, for the humanoid robotic experiments described in Chapter 5 we used a descriptor dimension of 16, since we were interested in very fast vision-based localization. Even though, this descriptor dimension may seem relatively small, matching is robust enough for obtaining accurate and fast vision-based localization as we will show in our experimental results section.

2.8 Loop Closure and Map Correction

By means of appearance based methods, loop closure situations can be detected. We try to match the set of descriptors from the current image to the stored descriptors from previous keyframes, but only taking into account those keyframes that are inside a small uncertainty area around the current camera location. We also check for geometric consistency by means of epipolar geometry. This geometric consistency check is very important and almost guarantees that there will be no false positives, even using a very low inlier threshold [Konolige and Agrawal, 2008]. Even simple, our method can detect very efficiently loop closure situations although *Bag of Visual Words* methods can be also used [Cummins and Newman, 2008; Angeli et al., 2008].

Once a loop closure is detected, the residual error in the 3D reconstruction can be corrected by a global BA procedure. Typical humanoid robot laboratory-based scenarios are relatively small, and therefore the accumulated drift or error at the end of a sequence is very small. In those scenarios, just few iterations are necessary in the global BA step. In contrast, for the large-scale scenarios such as the ones we are interested for visually

impaired visual SLAM and vision-based localization applications, the accumulated drift or error can be very high, about the order of several m. In those cases, some of the previous 3D points will exhibit high reprojection errors and since we are far away from the global minimum, we need to initialize the structure and motion closer to the global minimum before a global BA procedure, or otherwise BA can get stuck into a local minimum.

One solution is to optimise over relative constraints between camera poses using pose-graph optimization techniques [Dellaert and Kaess, 2006; Kaess et al., 2008; Grisetti et al., 2010]. These non-linear least squares techniques minimize a cost function that considers all the relative constraints between camera poses in the whole pose-graph. Once the pose-graph optimization is done, the set of 3D points are corrected with respect to the corrected keyframes. Finally, the whole map and set of reconstructed keyframes poses can be further optimised in a global BA setup.

2.9 Dense Scene Flow and Detection of Moving Objects

One of the advantages of stereo vision against monocular vision, is that we can exploit the information from four images at once, obtaining dense disparity maps (between the left and right stereo views at each frame) and dense 2D optical flow correspondences (between two consecutive frames). Since for every pixel that has a valid disparity value we know its 3D position (with respect to the camera coordinate frame) and the associated dense 2D optical flow (between two consecutive images), a dense scene flow [Vedula et al., 1999] representation can be obtained, describing the 3D motion of the world points. Figure 2.10 depicts an example of the four images that can be used in order to compute a dense scene flow representation of the environment.

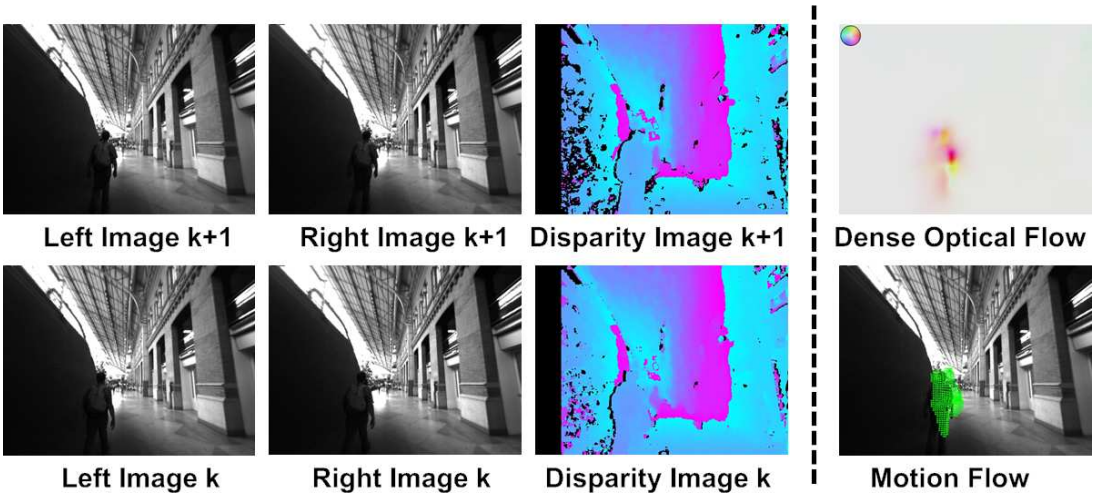


Figure 2.10: From each pair of images at each time step, a dense disparity map can be obtained. In this way, for every pixel in the image we know its 3D position with respect to the camera coordinate frame. By means of dense optical flow, a set of 2D correspondences between points from a reference image (e.g. left) of the stereo rig can be obtained between two consecutive frames. Then, using the set of 2D optical flow correspondences and the dense disparity values a scene flow representation of the environment can be obtained. Best viewed in color.

Scene flow was introduced in [Vedula et al., 1999], and should be considered as an essential algorithm for studying 3D motion in scenes. Scene flow describes the 3D motion of the points in the scene, whereas optical flow describes the 2D motion of the pixels

in the image. One possible alternative to compute scene flow, is by using the information from different camera views and optical flow estimates as proposed in [Vedula et al., 2005]. Although another alternatives exist for computing scene flow, as for example in [Devernay et al., 2006], where the authors proposed to compute scene flow by tracking 3D points and surface elements in a multi-camera setup.

Recently, scene flow techniques have been proposed for intelligent vehicles applications [Lenz et al., 2011; Wedel et al., 2011]. The work of Wedel et al. [2011] can be considered as the main reference for computing dense scene flow from stereo images. In this work, the authors proposed a variational framework for estimating dense stereo correspondences and dense 2D optical flow correspondences between consecutive images and also how dense scene flow estimates can be used for moving objects segmentation. In [Lenz et al., 2011], a sparse scene flow representation of the scene is obtained in order to detect moving objects in road-traffic urban environments. Those adjacent points that describe a similar scene flow are clustered and considered to belong to the same rigid object.

However, scene flow computation considering a moving stereo pair with 6-DoF and crowded scenes with many independent moving objects is more challenging than for common ADAS applications. In these kind of intelligent vehicles applications, it is possible to use the information from inertial sensors to compensate for camera egomotion. Despite of this, some approaches neglect the effect of camera rotation [Wedel et al., 2009] or do not perform any kind of egomotion compensation [Lenz et al., 2011]. In addition, typical stereo baselines that are used in intelligent vehicles applications (approximately 30-40 cm) are two or three times higher than the stereo baselines that can be used in wearable visual SLAM applications, where the stereo baseline is usually small (approximately 10 cm). For example, in [Nedevschi et al., 2009], where the authors presented a stereo-based pedestrian detection system for collision-avoidance applications, the baseline of the stereo rig was 32 cm. In contrast, in the work of Paz et al. [2008], where a stereo camera carried in hand was used for visual SLAM applications, the stereo rig baseline was 12 cm. Considering a higher stereo baseline yields lower uncertainties in the 3D reconstrution and consequently a better detection of possible moving objects at far depth ranges.

In stereo visual SLAM applications, it is necessary to consider the camera rotation and translation for egomotion compensation for a reliable scene flow computation. SLAM and scene flow computation from a stereo camera in highly crowded environments can be a difficult task. These scenarios are extremelly challenging since we can have fast camera motion, changes in lighting conditions, motion blur and many independent moving objects such as pedestrians that on occasions can almost cover the entire image view. For example, Figure 2.11 depicts few samples of typical environments where visually impaired users have to deal with during navigation tasks.

By means of dense scene flow estimates, we can derive motion likelihoods that can be used to segment moving objects, aiding the visual SLAM process. Up to the best of our knowledge, this is the first time that dense scene flow has been used in the context of visual SLAM for dealing with moving objects in crowded and highly dynamic scenarios. In Section 2.9.1 we will derive the set of equations that are necessary in order to obtain a dense scene flow representation. Finally, we will show in Section 2.9.2, how to obtain motion likelihoods by means of scene flow and how these likelihoods can be used to identify moving objects in the image. In this way, we obtain more robust visual odometry estimates and delete those features located on moving objects from the SLAM process, yielding superior 3D reconstruction results.

Figure 2.11: These three images depict some examples of the difficult challenging scenes that we can have in real-world crowded environments.

2.9.1 Scene Flow Computation

Given dense disparity maps between the two images of a stereo rig and dense optical flow estimates between two consecutive images, we can estimate a dense 3D scene flow. Using this information and considering that the images are rectified and undistorted, the 3D motion vector associated to two correspondent points can be computed considering the following equations:

$$\begin{pmatrix} X_{t+1} \\ Y_{t+1} \\ Z_{t+1} \end{pmatrix} = \frac{B}{u'_R - u'_L} \cdot \begin{pmatrix} u'_L - u_0 \\ v' - v_0 \\ f \end{pmatrix} \tag{2.11}$$

$$\begin{pmatrix} X_t \\ Y_t \\ Z_t \end{pmatrix} = \frac{B}{u_R - u_L} \cdot \mathbf{R} \cdot \begin{pmatrix} u_L - u_0 \\ v - v_0 \\ f \end{pmatrix} - \mathbf{T} \tag{2.12}$$

where Equation 2.11 describes the coordinates of a 3D point at time instant $t+1$, and Equation 2.12 describes the 3D coordinates of a 3D point at time t referenced to the camera coordinate frame at time $t+1$. $\mathbf{R}$ and $\mathbf{T}$ are respectively the rotation matrix and the translation vector of the camera between the two time steps. Notice that if the camera is stationary, the rotation matrix is equal to an identity matrix and the translation vector components are zero.

Considering the above two equations, the 3D translation or motion vector $\mathbf{M}$ can be expressed as follows:

$$\mathbf{M} = \left[\begin{pmatrix} X_{t+1} \\ Y_{t+1} \\ Z_{t+1} \end{pmatrix} - \begin{pmatrix} X_t \\ Y_t \\ Z_t \end{pmatrix} \right] = \left[\frac{B}{u'_R - u'_L} \begin{pmatrix} u'_L - u_0 \\ v' - v_0 \\ f \end{pmatrix} - \frac{B}{d_u} \cdot \mathbf{R} \cdot \begin{pmatrix} u_L - u_0 \\ v - v_0 \\ f \end{pmatrix} + \mathbf{T} \right] \tag{2.13}$$

In this work, we have used the dense optical flow method described in [Farnebäck, 2003], for obtaining dense 2D correspondences between consecutive frames $\{(u_L, v) \rightarrow (u'_L, v')\}$. This algorithm computes the 2D motion between consectutive frames by means of minimizing a cost function that approximates each neighborhood of both frames by quadratic polynomials. In addition, this algorithm is included in the OpenCV library [2] and exhibits good performance. Notice here that other advanced variational optical flow methods could have been employed [Pock et al., 2007; Müller et al., 2011], but the derivation of the scene flow and residual motion likelihoods remain the same. Of special interest seems to be the

[2] Available from http://sourceforge.net/projects/opencvlibrary/

recent work of Müller et al. [2011], where a total variation optical flow is aided by means of stereo and features correspondences information.

2.9.2 Detection of Moving Objects by Motion Likelihoods

Once we have computed the 3D motion vector for each pixel in the image, it is necessary to take into account the uncertainties of the scene flow vector in order to derive robust motion likelihoods. If we try to segment objects based on the modulus of the 3D motion vector, the segmentation is prone to errors due to measurement noise and depth uncertainty in the stereo reconstruction process. Therefore, it is much more robust to take all the uncertainties of the problem into account and derive a metric based on the Mahalanobis distance [Mahalanobis, 1936]. By means of this Mahalanobis distance, a metric can be derived in order to identify possible moving objects in the scene [Lenz et al., 2011; Wedel et al., 2011]. In addition, this metric can be used to perform a more detailed image segmentation by means of graph-cuts segmentation methods [Wedel et al., 2009], increasing considerably the processing time per frame.

First, we need to define the uncertainties of our measurements. Then, the resulting error of the 3D motion vector can be computed by linear error propagation and the Jacobian of the motion vector with respect to the measurements. Let us denote the scene flow vector of measurements $\mathbf{z_{SF}}$ as:

$$\mathbf{z_{SF}} = (u'_L, u'_R, v', u_L, u_R, v, t_x, t_y, t_z, q_x, q_y, q_z)^t \tag{2.14}$$

where (u'_L, u'_R, v') are the image coordinates for a given stereo point at time instant $t+1$ and (u_L, u_R, v) are the image coordinates for the same corresponding point at time instant t. The set of parameters (t_x, t_y, t_z) and (q_x, q_y, q_z) represent respectively the 3D translation vector and the rotation matrix parametrized by means of a unit quaternion between the two time instants. This translation vector and rotation matrix can be obtained directly from the visual odometry procedure described in Section 2.5. The Jacobian of the scene flow with respect to the vector of measurements $\mathbf{z_{SF}}$ can be obtained as:

$$\mathbf{J_{SF\ [3,12]}} = \frac{\partial \mathbf{M}}{\partial \mathbf{z_{SF}}} =$$

$$= \begin{pmatrix} \frac{\partial M_x}{\partial u'_L} & \frac{\partial M_x}{\partial u'_R} & \frac{\partial M_x}{\partial v'} & \frac{\partial M_x}{\partial u_L} & \frac{\partial M_x}{\partial u_R} & \frac{\partial M_x}{\partial v} & \frac{\partial M_x}{\partial t_x} & \frac{\partial M_x}{\partial t_y} & \frac{\partial M_x}{\partial t_z} & \frac{\partial M_x}{\partial q_x} & \frac{\partial M_x}{\partial q_y} & \frac{\partial M_x}{\partial q_z} \\ \frac{\partial M_y}{\partial u'_L} & \frac{\partial M_y}{\partial u'_R} & \frac{\partial M_y}{\partial v'} & \frac{\partial M_y}{\partial u_L} & \frac{\partial M_y}{\partial u_R} & \frac{\partial M_y}{\partial v} & \frac{\partial M_y}{\partial t_x} & \frac{\partial M_y}{\partial t_y} & \frac{\partial M_y}{\partial t_z} & \frac{\partial M_y}{\partial q_x} & \frac{\partial M_y}{\partial q_y} & \frac{\partial M_y}{\partial q_z} \\ \frac{\partial M_z}{\partial u'_L} & \frac{\partial M_z}{\partial u'_R} & \frac{\partial M_z}{\partial v'} & \frac{\partial M_z}{\partial u_L} & \frac{\partial M_z}{\partial u_R} & \frac{\partial M_z}{\partial v} & \frac{\partial M_z}{\partial t_x} & \frac{\partial M_z}{\partial t_y} & \frac{\partial M_z}{\partial t_z} & \frac{\partial M_z}{\partial q_x} & \frac{\partial M_z}{\partial q_y} & \frac{\partial M_z}{\partial q_z} \end{pmatrix} \tag{2.15}$$

Then, by means of the previous Jacobian and linear propagation of the errors, the covariance of the scene flow $\mathbf{\Sigma_{SF}}$ is obtained as follows:

$$\mathbf{\Sigma_{SF}} = \mathbf{J_{SF}} \cdot \mathbf{S_{SF}} \cdot \mathbf{J_{SF}}^t \tag{2.16}$$

where $\mathbf{S_{SF}}$ is the measurement noise matrix. We consider a pixelic standard deviation of ± 1 pixel for all the pixelic values that are involved in the measurement scene flow $\mathbf{z_{SF}}$. Regarding the translation and orientation variances, these quantities can be obtained from the visual odometry estimation. As shown in Section 2.5, we formulated visual odometry as a non-linear least squares minimization. When the sum of squares represents the

goodness of fit of a non-linear model to observed data, there are several approximations to obtain the covariance matrix of the estimated regression coefficients. These approximations usually approximate the Hessian of a function in the neighbourhood of a solution by means of the Jacobian product $J(x)^t \cdot J(x)$ being $J(x)$ the Jacobian matrix of the function $f(x)$, thereby avoiding to compute or approximate any second-order derivatives. For more information about how to compute these covariances estimates we recommend the reader to check the following works [Wolberg, 1967; Bard, 1974; Gill et al., 1981].

For a given pixel in the image (u, v), we can evaluate the associated Mahalanobis distance of the 3D motion vector in order to compute a residual motion likelihood:

$$\xi_{motion}(u, v) \;=\; \sqrt{(\mathbf{M}^t \cdot {\mathbf{\Sigma_{SF}}}^{-1} \cdot \mathbf{M})} \tag{2.17}$$

Assuming a stationary world and Gaussian error propagation, Equation 2.17 can be used to identify possible outliers or moving points. Stationary points will exhibit low residual motion likelihoods, whereas moving points will yield higher deviations from zero. Then, by thresholding on the residual motion likelihood we can identify those parts in the scene that are static or that belong to moving objects. In this way, we can identify those points in the image that are not static, deleting them from the SLAM process yielding more robust 3D reconstruction results. Figure 2.12 depicts some examples obtained with the proposed residual motion likelihood as shown in Equation 2.17.

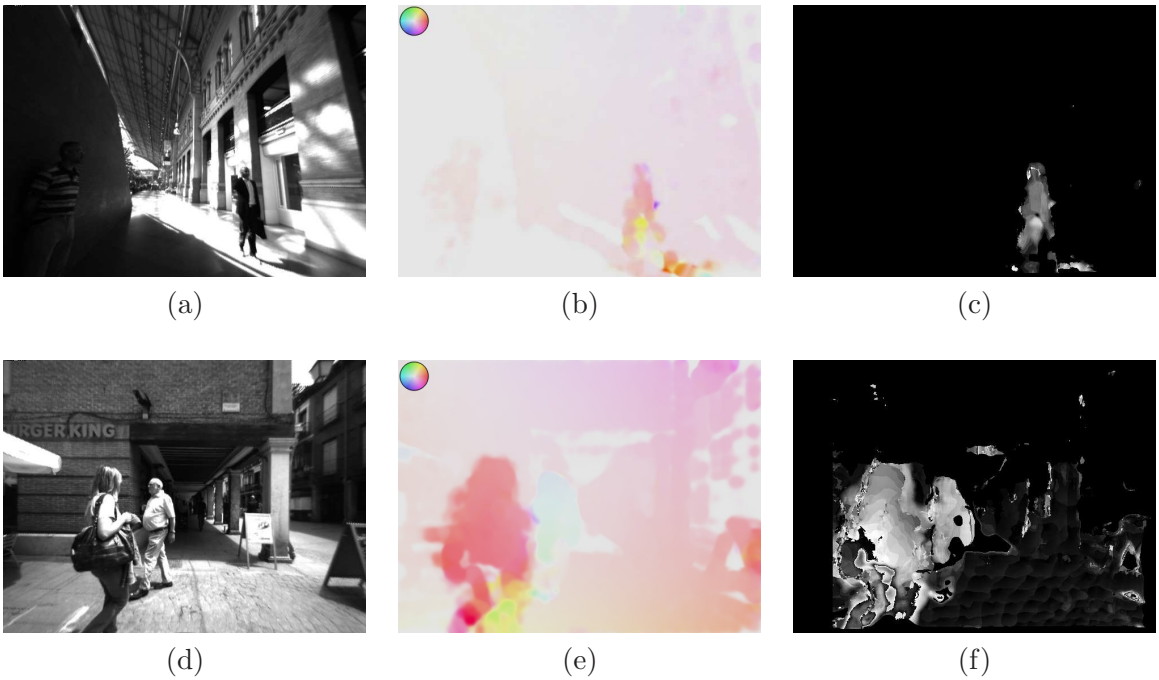


Figure 2.12: The first row of images (a-c) depict optical flow and motion likelihood results from one sequence conducted inside the Atocha railway station. (a) Original image (b) Dense Optical flow image (c) Motion likelihood results. The second row of images (d-e) depict optical flow and motion likelihood results from one sequence conducted in Alcalá de Henares city center. (d) Original image (e) Dense Optical flow image (f) Motion likelihood results. For the optical flow images, the color encodes the direction and the saturation encodes the magnitude of the optical flow. For the residual metric motion results, (black↔white) represents (low↔high) likelihood that the point is moving. Best viewed in color.

The squared Mahalanabois distance $\xi_{motion}(u, v)$ follows a χ^2 distribution, and outliers

can be identified by thresholding according to this distance. Assuming a stationary world, Gaussian error propagation and that the measurement variances are correct, one can use the assumed quantiles of the χ^2 distribution to find a proper threshold value for the residual motion likelihood. For example, considering three degrees of freedom, the 95% quantile of the distribution is 7.81. That means that a point is moving with a probability of 95% if the residual motion likelihood is higher than 7.81. We empirically found that a threshold value of 4.8, which corresponds to the 80% quantile of the distribution, gives satisfactory results for our set of experiments.

2.10 Dense 3D Reconstruction

In this section, we will show how to create a dense 3D representation of the scene in a simple and efficient way. Even though, a dense 3D reconstruction of the environment is not the main goal of our visual SLAM system, we describe a very simple method to obtain dense 3D point clouds, that can be used for a better visualization or different tasks such as planning or navigation. Our method is very simple and exploits all the previous information from the SLAM process and the dense disparity maps. We use a similar approach as the one described in [Geiger et al., 2011b]. When we add a new keyframe into the system, we reproject all the 3D points from the previous keyframe into the image plane of the new keyframe and check those 3D point reprojections that fall onto valid disparity values. We can perform this reprojection because we know the camera egomotion between these two keyframes thanks to the information provided by the SLAM algorithm. Moreover, we also check for photoconsistency discarding those correspondences between points that their difference in image intensity is higher than a fixed threshold in a rectangular area of interest. In addition, we also discard those 3D point reprojections that fall onto pixels that belong to moving objects. For those points that are added in the dense scene, we fuse both 3D points by computing their 3D mean reducing measurement noise errors. Figure 2.13 depicts a graphical example of the dense disparity maps fusion between two consecutive keyframes.

When dealing with large-scale sequences the memory requirements of storing the whole dense 3D reconstruction of the scene can become prohibitive for most computer architectures. For example, just adding one disparity map into the whole dense reconstruction yields $640 \times 480 \times 4 bytes = 1.17$ MB, considering an image resolution of 640×480 and that each disparity value can be encoded by means of a float value. If we have a high number of keyframes storing that amount of information in memory can become a computational bottleneck. Therefore, what we do in practice is just to keep a small dense 3D reconstruction from a set of selected window of keyframes over the lastest one. The rest of information from previous keyframes is saved to disk. Once the sequence is finished, we save the final 3D dense scene by reading the information stored in disk and computing the 3D position of the points considering the resulting optimized camera poses.

Figure 2.14, depicts one example of a dense 3D reconstruction in one outdoor experiment in the city of Alcalá de Henares. Figure 2.14(a) depicts an image view of the Cervantes house area, whereas Figure 2.14(b) depicts the reconstructed 3D point cloud.

In the next figures we can observe some dense 3D reconstruction results from different experiments performed with the humanoid robot HPR-2 in a laboratory environment in Toulouse, France. Figure 2.15(a) depicts the 3D dense reconstruction from a bird's eye view camera viewpoint. In this sequence the robot performed a $3\ m \times 3\ m$ square trajectory. Figure 2.15(b) depicts one area of the dense 3D reconstruction from a circular

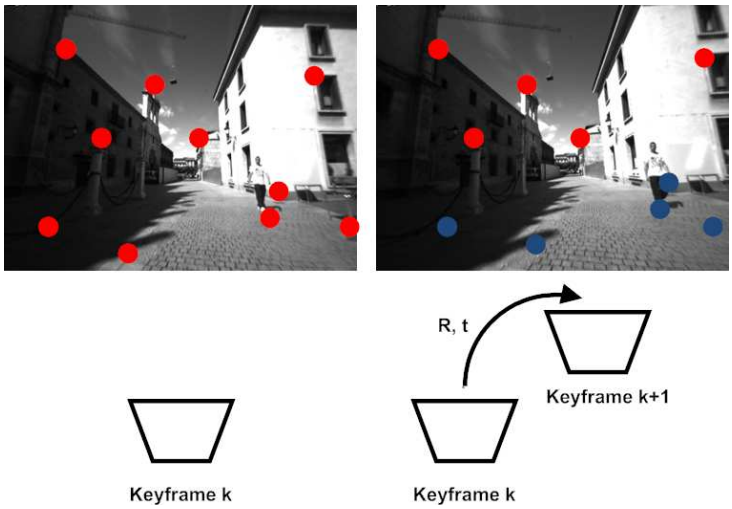
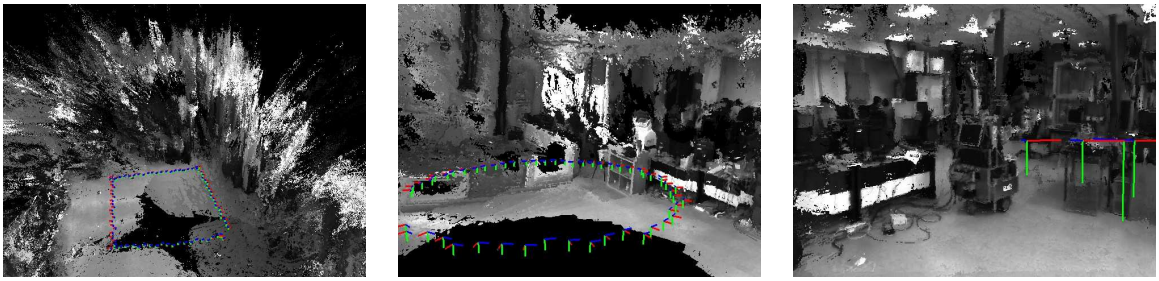


Figure 2.13: Fusion of disparity maps: The points from the previous keyframe are reprojected into the image plane of the current keyframe. Only those points that the reprojections fall onto a valid disparity value (depicted in red), have similar grey intensity and do not belong to moving objects areas are considered and added to the dense 3D scene. The rest of 3D points reprojections (depicted in blue) are discarded. Best viewed in color.

(a) (b)

Figure 2.14: Example of a dense 3D reconstruction in the city of Alcalá de Henares. (a) Image view of the Cervantes house area (b) Reconstructed dense 3D point cloud.

trajectory. Finally, Figure 2.15(c) depicts some details of the laboratory, where we can appreciate the dense 3D reconstruction of a wheeled robot. In all the images the coordinates axis of the cameras are depicted by means of a standard RGB color representation.



(a) (b) (c)

Figure 2.15: Some examples of dense 3D reconstructions in a humanoid robotics laboratory. (a) Bird's eye view of a square $3m \times 3m$ trajectory (b) Some details of a circular trajectory (c) Dense 3D reconstruction of a wheeled robot. Best viewed in color.

2.11 Conclusions and Future Works

In this chapter, we have described a stereo visual SLAM system that allows to obtain an accurate trajectory estimation and 3D maps of the environment. These 3D maps can be used later for an efficient vision-based localization. In addition, in order to deal with extremely crowded and dynamic environments, we exploit all the information from the stereo camera building a dense scene flow representation of the environment. From a dense scene flow representation, we can obtain residual motion likelihoods that can be used for the detection of moving objects in the image. Furthermore, we have also shown how a dense 3D reconstruction of the environment can be obtained considering low computational demands.

Our stereo visual SLAM system can be used for small and large-scale environments. Constant time operation is achieved by means of an incremental local BA approach, using a sliding window of a fixed number of keyframes, optimizing only a subset of the camera poses and the 3D map points involved in the whole reconstruction. When a loop closure is detected, the residual drift is corrected by means of pose-graph optimization techniques. Then, at the end of the sequence the map and the set of camera poses can be further optimized by means of global BA.

In the next future, we are interested in improving the capabilities of visual SLAM and SfM approaches in order to deal with moving objects in the scene. We think that the combination of a robust dense scene flow representation plus the use of well-understood pedestrian detectors [Nedevschi et al., 2009; Enzweiler and Gavrila, 2009] and the tracking of the moving objects [Ess et al., 2009] can yield a very robust visual SLAM method that can be used in extremelly challenging and crowded environments. In addition, we think that a more detailed 3D scene understanding [Geiger et al., 2011a] can be of benefit for visual SLAM approaches.

Chapter 3

Gauge SURF Descriptors

Given two images of the same scene, image matching is the problem of establishing correspondence and is a core component of all sorts of computer vision algorithms, particularly in classic problems such as Structure from Motion (SfM) [Agarwal et al., 2009], visual categorization [Csurka et al., 2004] or object recognition [Lowe, 1999]. There has been a wealth of work in particular on matching image keypoints, and the key advances have been in multiscale feature detectors and invariant descriptors which permit robust matching even under significant changes in viewing conditions.

We have studied the use of gauge coordinates [ter Haar Romeny, 2003] for image matching and SfM applications and incorporated them into a Speeded-Up Robust Features (SURF) [Bay et al., 2008] descriptor framework to produce a family of descriptors of different dimensions which we named Gauge-SURF (G-SURF) descriptors. With gauge coordinates, every pixel in the image is described in such a way that if we have the same 2D local structure, the description of the structure is always the same, even if the image is rotated. This is possible since multiscale gauge derivatives are rotation and translation invariant. In addition, gauge derivatives play a key-role in the formulation of non-linear diffusion processes, as will be explained in Section 3.2.1. By using gauge derivatives, we can make blurring locally adaptive to the image itself, without affecting image details.

The G-SURF descriptors are very related to non-linear diffusion processes in image processing and computer vision [Perona and Malik, 1990; Álvarez et al., 1992]. In the typical Gaussian scale-space framework [Lindeberg, 1998], details are blurred during evolution (i.e. the convolution of the original image with Gaussian kernels of increasing standard deviation). The advantage of blurring is the removal of noise, but relevant image structures like edges are blurred and drift away from their original locations during evolution. In general, a good solution should be to make the blurring locally adaptive to the image yielding the blurring of noise, while retaining details or edges. Instead of local first-order spatial derivatives, G-SURF descriptors measure per pixel information about image blurring and edge or detail enhancing, resulting in a more discriminative descriptors.

We have obtained notable results in an extensive image matching evaluation using the standard evaluation framework of Mikolajczyk and Schmid [2005]. In addition, we have tested our family of descriptors in large-scale 3D SfM datasets [Brown et al., 2011] and visual categorization experiments [Csurka et al., 2004] with satisfactory results. Our results show that G-SURF descriptors outperform or approximate state of the art methods in accuracy while exhibiting low computational demands making it suitable for real-time applications.

We are interested in robust multiscale feature descriptors, to reliably match two images

in real-time for visual odometry [Nistér et al., 2004] and large-scale 3D SfM [Brown et al., 2011] applications. Image matching here, is in fact a difficult task to solve due to the large motion between frames and the high variability of camera movements. For this purpose, we need desciptors that are fast to compute and at the same time exhibit high performance.

In addition, we have ellaborated an open-source library called *OpenGSURF* that contains all the family of G-SURF descriptors and we plan to make it publicly available. This family of descriptors comprises of several descriptors of different dimensions based on second-order multiscale gauge derivatives. Depending on the application some descriptors may be preferred instead of others. For example, for real-time applications a low-dimensional descriptor should be preferred instead of a high-dimensional one, whereas for image-matching applications considering severe image transformations one can expect a higher recall by using high-dimensional descriptors. Up to the best of our knowledge, this is the first open source library that allows the user to choose between different dimensional descriptors. Current open source descriptors libraries [Evans, 2009; Vedaldi and Fulkerson, 2008] just have implementations for the standard SURF and Scale Invariant Feature Transform (SIFT) [Lowe, 2004] descriptors default dimensions (64 and 128 respectively). This can be a limitation and a computational bottleneck for some real-time applications that do not necessarily need those default descriptor dimensions.

The rest of the chapter is organized as follows: Related work is discussed in Section 3.1. Gauge coordinates are introduced in Section 3.2 and the importance of gauge derivatives in non-linear diffusion schemes is reviewed in Section 3.2.1. Then we briefly discuss SURF based descriptors in Section 3.3. The overall framework of our family of descriptors is explained in Section 3.4. We show extensive experimental results in image matching, large-scale 3D SfM and visual categorization applications in Section 3.5. Finally, we describe main conclusions and future work in Section 3.6.

3.1 Related Work

The highly influential SIFT [Lowe, 2004] features have been widely used in applications from mobile robotics to object recognition, but are relatively expensive to compute and are not suitable for some applications with real-time demands. Inspired by SIFT, Bay et al. [2008] proposed the SURF features both detector and descriptor. SURF features exhibit better results than previous schemes with respect to repeatability, distinctiviness and robustness, but at the same time can be computed much faster thanks to the use of integral images [Viola and Jones, 2004]. Recently, Agrawal et al. [2008] proposed some modifications of SURF in both the detection and description steps. They introduced Center Surround Extremas (CenSurE) features and showed that they outperform previous detectors and have better computational characteristics for real-time applications. Their variant of the SURF descriptor, Modified-SURF (M-SURF), efficiently handles the descriptor boundaries problem and uses a more intelligent two-stage Gaussian weighting scheme in contrast to the original implementation which uses a single Gaussian weighting step.

All the mentioned approaches rely on the use of the Gaussian scale-space [Lindeberg, 1998] framework to extract features at different scales. An original image is blurred by convolution with Gaussian kernels of successively large standard deviation to identify features at increasingly large scales. The main drawback of the Gaussian kernel and its set of partial derivatives is that both interesting details and noise are blurred away to the same degree. It seems to be more appropriate in feature description to make blurring

locally adaptive to the image data so that noise will be blurred, while at the same time details or edges will remain unaffected. In this way, we can increase distinctiveness when describing an image region at different scale levels. In spirit, non-linear diffusion shares some similarities with respect to the *geometric blur* approach [Berg and Malik, 2001], in where the the amount of Gaussian blurring is proportional to the distance from the point of interest.

From their definition, gauge derivatives are local invariants. Matching by local invariants has previously been studied in the literature. In [Schmid and Mohr, 1997], the family of local invariants known as the *local jet* [Florack et al., 1993] was used for image matching applications. Their descriptor vector contained 8 invariants up to third order for every point of interest in the image. This work supposed a step-forward over previous invariant recognition schemes [Rothwell et al., 1992]. In [Mikolajczyk and Schmid, 2005], the performance of the *local jet* (with invariants up to third order) was compared against other descriptors such as steerable filters [Freeman and Adelson, 1991], image moments [Gool et al., 1996] or SIFT descriptor. In their experiments the local jet exhibits poor performance compared to SIFT. We hypothesize that this poor performance is due to the fixed settings used in the experiments, such as a fixed image patch size and a fixed Gaussian derivative scale. In addition, invariants of high order are more sensitive to geometric and photometric distortions than first-order methods. In [Platel et al., 2006], the local jet (with invariants up to third order) was again used for matching applications, and they showed that even a descriptor vector of dimension 6 can outperfom SIFT descriptor performance for small perspective changes. By a suitable scaling and normalization, the authors obtained invariance to spatial zooming and intensity scaling. Although these results were encouraging, a more detailed comparison with other descriptors would have been desirable. However, this work motivated us to incorporate gauge invariants into the SURF descriptor framework.

Brown et al. [2011], proposed a framework for learning discriminative local dense image descriptors from training data. The training data was obtained from large-scale real 3D SfM scenarios, and accurate ground truth correspondences were generated by means of multi-view stereo matching techniques [Goesele et al., 2006, 2007] that allow to obtain very accurate correspondences between 3D points. They describe a set of building blocks for building discriminative local descriptors that can be combined together and jointly optimized to minimize the error of a nearest-neighbor classifier. In this thesis, we use the evaluation framework of Brown et al. [2011] to evaluate the performance of multiscale gauge derivatives under real large-scale 3D SfM scenarios.

3.2 Gauge Coordinates and Multiscale Gauge Derivatives

Gauge coordinates are a very useful tool in computer vision and image processing. Using gauge coordinates, every pixel in the image is described in such a way that if we have the same 2D local structure, the description of the structure is always the same, even if the image is rotated. This is possible since every pixel in the image is fixed separately in its own local coordinate frame defined by the local structure itself and consisting of the gradient vector $\vec{w}$ and its perpendicular direction $\vec{v}$:

$$\begin{aligned} \vec{w} &= \left(\frac{\partial L}{\partial x}, \frac{\partial L}{\partial y}\right) = \frac{1}{\sqrt{L_x^2+L_y^2}} \cdot (L_x, L_y) \\ \vec{v} &= \left(\frac{\partial L}{\partial y}, -\frac{\partial L}{\partial x}\right) = \frac{1}{\sqrt{L_x^2+L_y^2}} \cdot (L_y, -L_x) \end{aligned} \tag{3.1}$$

In Equation 3.1, L denotes the convolution of the image I with a 2D Gaussian kernel $g(x, y, \sigma)$, where σ is the kernel's standard deviation or scale parameter:

$$L(x, y, \sigma) \;=\; I(x, y) * g(x, y, \sigma) \tag{3.2}$$

Derivatives can be taken up to any order and at multiple scales for detecting features of different sizes. Raw image derivatives can only be computed in terms of the Cartesian coordinate frame x and y, so in order to obtain gauge derivatives we need to use directional derivatives with respect to a fixed gradient direction (L_x, L_y). The $\vec{v}$ direction is tangent to the isophotes or lines of constant intensity, whereas $\vec{w}$ points in the direction of the gradient, thus $L_v = 0$ and $L_w = \sqrt{L_x^2 + L_y^2}$. If we take derivatives with respect to first-order gauge coordinates, since these are fixed to the object, irrespective of rotation or translation, we obtain the following interesting results:

1. Every derivative expressed in gauge coordinates is an orthogonal invariant. The first-order derivative $\frac{\partial L}{\partial \vec{w}}$ is the derivative in the gradient direction, and in fact the gradient is an invariant itself.

2. Since $\frac{\partial L}{\partial \vec{v}} = 0$, this implies that there is no change in the luminance if we move tangentially to the constant intensity lines.

By using gauge coordinates, we can obtain a set of invariant derivatives up to any order and scale that can be used efficiently for image description and matching. Of special interest, are the second-order gauge derivatives L_{ww} and L_{vv}:

$$L_{ww} \;=\; \frac{L_x^2 L_{xx} + 2 \cdot L_x L_{xy} L_y + L_y^2 L_{yy}}{L_x^2 + L_y^2} \tag{3.3}$$

$$L_{vv} \;=\; \frac{L_y^2 L_{xx} - 2 \cdot L_x L_{xy} L_y + L_x^2 L_{yy}}{L_x^2 + L_y^2} \tag{3.4}$$

These two gauge derivatives can be obtained as the product of gradients in $\vec{w}$ and $\vec{v}$ directions and the 2×2 second-order derivatives or Hessian matrix.

$$L_{ww} \;=\; \frac{1}{L_x^2 + L_y^2} \begin{pmatrix} L_x & L_y \end{pmatrix} \begin{pmatrix} L_{xx} & L_{xy} \\ L_{yx} & L_{yy} \end{pmatrix} \begin{pmatrix} L_x \\ L_y \end{pmatrix} \tag{3.5}$$

$$L_{vv} \;=\; \frac{1}{L_x^2 + L_y^2} \begin{pmatrix} L_y & -L_x \end{pmatrix} \begin{pmatrix} L_{xx} & L_{xy} \\ L_{yx} & L_{yy} \end{pmatrix} \begin{pmatrix} L_y \\ -L_x \end{pmatrix} \tag{3.6}$$

L_{vv} is often used as a ridge detector. Ridges are elongated regions of approximately constant width and intensity, and at these points the curvature of the isophotes is high. L_{ww} gives information about gradient changes in the gradient direction.

Figure 3.1(a) illustrates first-order gauge coordinates. Unit vector $\vec{v}$ is always tangential to lines of constant image intensity (isophotes), while unit vector $\vec{w}$ is perpendicular and points in the gradient direction. Figure 3.1(b) depicts an example of the resulting second-order gauge derivative L_{ww} on one of the images from the Mikolajczyk and Schmid's dataset [2005].

According to [Schmid and Mohr, 1995], where the authors explicitly describe the set of second-order invariants used in the local jet, we can find two main differences between the second-order gauge derivatives L_{ww}, L_{vv} and the local jet. The first difference is that by definition gauge derivatives are normalized with respect to the modulus of the gradient at

(a) (b)

Figure 3.1: (a) Local first-order gauge coordinates (b) Resulting gauge derivative L_{ww} applied on the first image of the Leuven dataset, at a fixed scale $\sigma = 2$ pixels.

each pixel. Although this normalization can be also included in the local jet formulation as shown in [Platel et al., 2006]. The second difference and the most important one, is that the invariant L_{vv} is not included in the set of second-order derivatives of the local jet. The invariant L_{vv} plays a fundamental role in non-linear diffusion processes [Álvarez et al., 1992, 1993]. Typically, Equation 3.4 is used to evolve the image in a way that locally adapts the amount of blurring to differential invariant structure in the image in order to perform edge-preserving smoothing [ter Haar Romeny, 2003].

3.2.1 Importance of Gauge Derivatives in Non-Linear Diffusion Schemes

In this section we aim to throw some more light on our decision to use gauge derivatives in a feature descriptor by briefly reviewing non-linear image diffusion, and highlighting the important role of gauge derivatives in these schemes. Koendenrik [1984] and Lindeberg [1998] showed that the Gaussian kernel and its set of partial derivatives provide the unique set of operators for the construction of linear scale-space under certain conditions. Some examples of algorithms that rely on the Gaussian scale-space framework are SIFT [Lowe, 2004] and SURF [Bay et al., 2008] invariant features.

However, to repeat, details are blurred in Gaussian scale-space during evolution. The advantage of blurring is the removal of noise, but relevant image structures like edges are blurred and drift away from their original locations during evolution. In general, a good solution should be to make the blurring locally adaptive to the image yielding the blurring of noise, while retaining details or edges.

In the early nineties, several Partial Differential Equations (PDEs) were proposed for dealing with the mentioned Gaussian scale-space problem. Some famous examples are the Perona-Malik equation [Perona and Malik, 1990] and the Mean Curvature Motion (MCM) [Álvarez et al., 1992]. Note that in general, non-linear diffusion approaches perform better than linear diffusion schemes [ter Haar Romeny, 2003; Kuijper, 2009]. Recently, Kuijper [2009] showed that the evolution of an image can be expressed as a linear combination of the two different second-order gauge derivatives L_{ww} and L_{vv}. According to this, we can conclude that non-linear approaches steer between blurring L_{ww} and edge regularising L_{vv}. Some examples of practical applications of L_{ww} flow are image impaiting [Caselles et al., 1998]. For L_{vv} flow, one example is the cited MCM [Álvarez et al., 1992].

Based on this, we can think about a local invariant descriptor that takes into account the information encoded in the two gauge derivatives L_{vv} and L_{ww} while the image evolves

according to a scale σ. Notice that in our family of descriptors we just replace the first-order local derivatives L_x and L_y for the gauge derivatives L_{vv} and L_{ww} and do not perform any image evolution through a non-linear scale space. That is, our descriptors will measure information about blurring (L_{ww}) and edge enhancing (L_{vv}) for different scale levels.

Another difference between first-order local derivatives and gauge ones, is that gauge derivatives are intrisically weighted with the strength of the gradient L_w. That is, the weighting is intrinsically related to the image structure itself, and no artificial weighting such as Gaussian weighting is needed. This is an important advantage over other descriptors, such as for example SURF, where different Gaussian weighting schemes [Agrawal et al., 2008] have been proposed to improve the performance of the original descriptor.

(a) (b) (c)

(d) (e) (f)

Figure 3.2: Gaussian Scale Space versus Non-Linear Diffusion schemes. The first row depicts the evolution of the sixth image from the Mikolajczyk and Schmid's Bikes dataset considering a Gaussian scale space of increasing scale σ in pixels. (a) $\sigma = 2$ (b) $\sigma = 4$ (c) $\sigma = 8$. The second row depicts the evolution of the same reference image but considering the MCM non-linear diffusion flow. (d) $\sigma = 2$ (e) $\sigma = 4$ (f) $\sigma = 8$. Notice how with non-linear diffusion schemes, details are enhanced and noise is removed, whereas for the Gaussian scale-space, details and noise are blurred in the same degree.

3.3 SURF Based Descriptors

Agrawal et al. [2008] proposed the Modified Upright-SURF descriptor (MU-SURF), which is a variant of the original U-SURF descriptor. MU-SURF handles descriptor boundary effects and uses a more robust and intelligent two-stage Gaussian weighting scheme. For a detected feature at scale s, Haar wavelet responses L_x and L_y of size $2s$ are computed over a $24s \times 24s$ region. This region is divided into $9s \times 9s$ subregions with an overlap of $2s$. The Haar wavelet responses in each subregion are weighted with a Gaussian ($\sigma_1 = 2.5s$) centered on the subregion center and summed into a descriptor vector $d_v = (\sum L_x, \sum L_y, \sum |L_x|, \sum |L_y|)$. Then, each subregion vector is weighted using a

Gaussian ($\sigma_2 = 1.5s$) defined over a mask of 4×4 and centered on the interest keypoint. Finally, the descriptor vector of length 64 is normalized into a unit vector to achieve invariance to contrast. Figure 3.3(a) depicts the involved regions and subregions in the MU-SURF descriptor building process.

The main differences between the MU-SURF and U-SURF descriptor is that the size of the region is reduced to $20s \times 20s$ divided into $5s \times 5s$ subregions without any overlap between subregions. In addition, Haar wavelet responses in each subregion are weighted by a Gaussian ($\sigma = 3.3s$) centered at the interest keypoint. This is a very small standard deviation considering that the square grid size is $20s \times 20s$. Figure 3.3(b) depicts a normalized 2D Gaussian kernel considering a standard deviation $\sigma = 3.3$. Notice how this weighting scheme smoothes completely the contribution of far points from the point of interest. Therefore, only points within a distance of ± 5 pixels have a significant influence in the whole descriptor.

The upright version of SURF-based descriptors (U-SURF) is faster to compute and usually exhibits higher performance (compared to its corresponding rotation invariant version, SURF) in applications where invariance to rotation is not necessary. Some examples of these applications are 3D reconstruction [Bay et al., 2008] or face recognition [Dreuw et al., 2009]. Although the MU-SURF descriptor is not invariant to rotation, it can be easily adapted for this purpose by interpolating Haar wavelet responses according to a dominant orientation in the same way as is done in the orginal SURF descriptor.

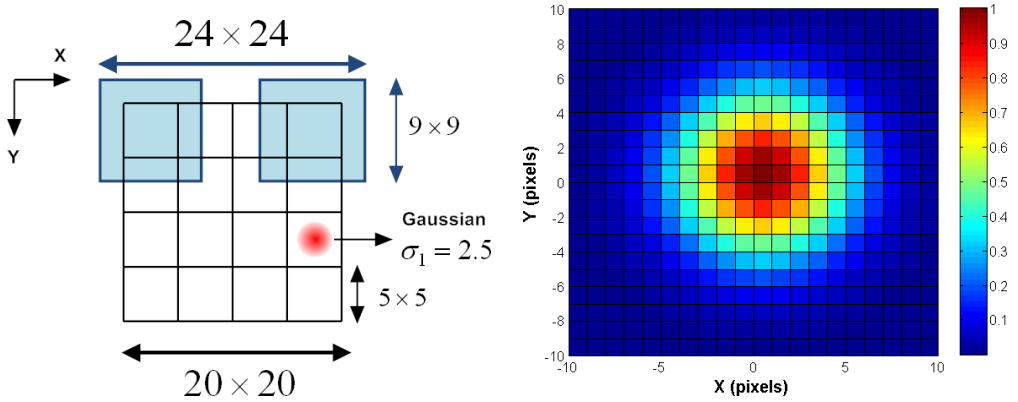


Figure 3.3: (a) MU-SURF descriptor building process. All sizes are relative to the scale of the feature s (b) The single Gaussian weighting scheme proposed in the original SURF descriptor. Normalized 2D gaussian kernel values considering a Gaussian kernel of standard deviation $\sigma = 3.3$ centered at the interest keypoint. Best viewed in color.

3.4 Gauge-SURF Descriptors

Our family of G-SURF descriptors are based on the original SURF descriptor. However, instead of using the local first-order derivatives L_x and L_y, we replace these two derivatives by the second-order gauge derivatives L_{ww} and L_{vv}. For computing multiscale gauge derivatives, we always need to compute the derivatives first in the Cartesian coordinate frame (x, y), and then fix the gradient direction (L_x, L_y) for every pixel. After these computations, we can obtain invariant gauge derivatives up to any order and scale with respect to the new gauge coordinate frame $(\vec{w}, \vec{v})$.

From the definition of gauge coordinates in Equation 3.1, it can be observed that these coordinates are not defined at pixel locations where $\sqrt{L_x^2 + L_y^2} = 0$, i.e. at saddle points and extrema of the image. In practice this is not a problem as ter Haar Romeny [2003] states, since we have a small number of such points, and according to Morse theory [Damon, 1995] we can get rid of such singularities by infinitesimally small local changes in the intensity landscape. What we do in practice is to not sum the contributions of these points into the final descriptor vector.

Now, we will describe the building process of a GU-SURF descriptor of dimension 64. For a detected feature at scale s, we compute first and second-order Haar wavelet responses $L_x, L_y, L_{xx}, L_{xy}, L_{yy}$ over a $20s \times 20s$ region. We call L_x the Haar wavelet response in the horizontal direction and L_y the response in the vertical direction. The descriptor window is divided into 4×4 regular subregions without any overlap. Within each of these subregions Haar wavelets of size $2s$ are computed for 25 regularly distributed sample points. Once we have fixed the gauge coordinate frame for each of the pixels, we compute the gauge invariants $|L_{ww}|$ and $|L_{vv}|$. Each subregion yields a four-dimensional descriptor vector $d_v = (\sum L_{ww}, \sum L_{vv}, \sum |L_{ww}|, \sum |L_{vv}|)$. Finally, the total length of the unitary descriptor vector is 64.

Figure 3.4 depicts an example of the GU-SURF descriptor building process. For simplicity reasons, we only show one gauge coordinate frame for each of the 4×4 subregions. Note that if we want to compute a descriptor which is invariant to rotation, we do not need to interpolate the value of the invariants L_{ww} and L_{vv} according to a dominant orientation as in SURF or M-SURF. Due to the rotation invariance of gauge derivatives, we only have to rotate the square grid.

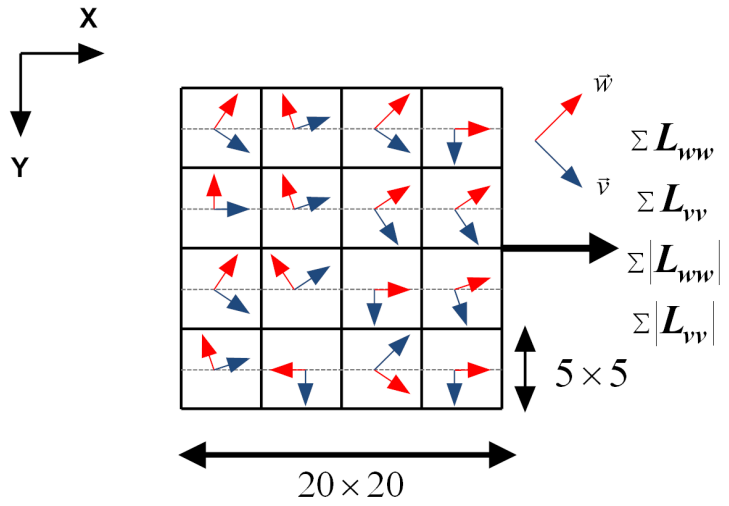


Figure 3.4: GU-SURF descriptor building process. Note that for the rotationally-invariant version of the descriptor we just have to rotate the square grid. Best viewed in color.

In the same way as proposed in SURF, we use box-filters to approximate first and second-order Gaussian derivatives. These box-filters are constructed through the use of integral images [Viola and Jones, 2004], which allows the approximation of Gaussian derivatives with low computational demands.

In Section 3.4.1, we describe the rest of descriptors of the G-SURF family included in the OpenGSURF library and the notation of the descriptors we will use throughout the rest of the chapter.

3.4.1 Descriptors Notation

Similar to [Bay et al., 2008], we can modify the number of divisions of the square grid and the size of each subregion in Figure 3.4 to obtain descriptors of different dimensions. The descriptor size has a major impact on the matching speed and recall rates. We also tested the extended version of the descriptors [Bay et al., 2008]. This option is included in the OpenGSURF library, however this descriptor version is not evaluated in this thesis. As shown in [Bay et al., 2008], the overall effect of the extended descriptor is minimal.

Now, we will describe the notation for the set of descriptors we use throughout the rest of the chapter, with the number of dimensions of the descriptors in parenthesis. For the SURF-based descriptors the default dimension is 64, whereas for SIFT the default dimension is 128.

- **SURF (64)**: Original SURF implementation as described in [Bay et al., 2006b, 2008] that uses a single Gaussian weighting scheme of a standard deviation $\sigma = 3.3s$ centered at the interest keypoint and a square grid of $20s \times 20s$.

- **M-SURF (64)**: Modified-SURF descriptor as described in [Agrawal et al., 2008]. This descriptor uses a square grid of $24s \times 24s$ considering an overlap of Haar wavelets responses and two Gaussian weighting steps.

- **G-SURF (64)**: Gauge-SURF descriptor, that uses second-order multiscale gauge derivatives and a square grid of $20s \times 20s$ without any additional Gaussian weighting step.

- **MG-SURF (64)**: Modified Gauge-SURF descriptor, that uses the same scheme as the M-SURF but replacing first-order local derivatives (L_x, L_y) for second-order gauge ones (L_{ww}, L_{vv}).

- **NG-SURF (64)**: No Gaussian Weighting-SURF descriptor. This descriptor is exactly the same as the original SURF descriptor, with the difference that no Gaussian weighting step is applied. In this way, we can perform a fair comparison between gauge derivatives and first-order local derivatives based descriptors without any additional weighting scheme.

- **SIFT (128)**: The SIFT descriptor as described in [Lowe, 2004]. This descriptor has a dimension of 128.

For all the mentioned above descriptors, we denote the *upright* version of the descriptors (not invariant to rotation) adding the prefix U to the name of the descriptor. For example, GU-SURF is the upright version of the G-SURF descriptor. By modifying the number of divisions of the square grid and the size of each of the subregions, we can obtain descriptors of different dimensions. Now, we will describe the number of divisions of the square grid and the size of each subregion for each of the descriptor sizes we evaluate in this chapter. The first number in parenthesis indicates the dimension of the descriptor with the new square grid and subregion size.

- **(36)**: Square grid of size $18s \times 18s$ yielding 3×3 subregions each of size $6s \times 6s$.

- **(144)**: Square grid of size $24s \times 24s$ yielding 6×6 subregions each of size $4s \times 4s$.

3.5 Results and Discussion

In this section, we present extensive experimental image matching results obtained on the standard evaluation set of Mikolajczyk and Schmid [2005], large-scale 3D SfM applications [Brown et al., 2011] and visual categorization experiments [Csurka et al., 2004]. In addition, we introduce a new dataset named *Iguazu* that consist of a series of six images with the addition of increasing random Gaussian noise levels with respect to the first image of the dataset. In some research areas such as medical imaging, RADAR or astronomy, images are usually corrupted by different types of random noise. Therefore, we think that the evaluation of local descriptors in these kind of datasets is of interest.

Our family of G-SURF descriptors implementation is based on the OpenSURF library[1]. OpenSURF is an open source C++ based library with detailed documentation and a reference paper [Evans, 2009]. To our knowledge, this library is widely used in the computer vision and robotics community and exhibits good perfomance, while having speed similar to the original SURF library which is only available as a binary. Currently, OpenSURF uses by default the M-SURF descriptor, since perfomance is much higher than when using the single weighting Gaussian scheme. We think, that OpenSURF is a good open source library for perfoming a fair evaluation and comparison of a set of descriptors that are all based on the same source code framework.

We also show comparison results with respect to SIFT descriptor, using Vedaldi's implementation [Vedaldi and Fulkerson, 2008]. In all SIFT experiments we used the default magnification factor $m = 3.0$, i.e. each spatial bin of the histogram has support of size $m \cdot \sigma$ where σ is the scale of the point of interest. This parameter has an important effect in the descriptor performance. See [Vedaldi, 2007] for more details.

We have compared G-SURF descriptors to SURF, M-SURF, NG-SURF (all based on OpenSURF implementation) and SIFT (based on Vedaldi's implementation), in both standard and upright forms. Agrawal et al. [2008] claim that M-SURF's performance is similar to the original SURF library, although their implementation is much faster than the original one. Like Agrawal et al., we also noticed that the standard single Gaussian weighting scheme as proposed in the original SURF algorithm [Bay et al., 2008] gives poor results. However, we also include in our comparison the standard SURF method based on the OpenSURF implementations, since this single Gaussian scheme is still used in practically all of the open source libraries that include the SURF algorithm, such as OpenCV or dlib C++ [2]. In addition, in Section 3.5.2 we also show some comparison results with respect to the OpenCV SURF implementation, since this library has become a de facto standard for fast-to-compute descriptors.

The rest of the experimental results and discussion section is organized as follows: In Section 3.5.1 we show extensive image matching experiments based on the standard evaluation framework of Mikolajczyk and Schmid [2005], with the addition of a new dataset for evaluating descriptor performance under different image noise settings. Then, in Section 3.5.3 we evaluate the performance of G-SURF descriptors in large-scale 3D SfM scenarios. In Section 3.5.4 we show some results on visual categorization applications, and finally in Section 3.5.5 we describe some implementation details and timing evaluation results.

[1] Available from http://code.google.com/p/opensurf1/
[2] Available from http://dclib.sourceforge.net/

3.5.1 Image Matching Experiments

We tested our descriptors using the image sequences and testing software provided by Mikolajczyk [3]. We used OpenSURF's Fast Hessian to extract the keypoints in every image and then compute the descriptors, setting the number of octaves and number of intervals to 4 and 2 respectively.

The standard dataset includes several image sets (each sequence generally contains 6 images) with different geometric and photometric transformations such as image blur, lighting, viewpoint, scale changes, zoom, rotation and JPEG compression. In addition, the ground truth homographies are also available for every image transformation with respect to the first image of every sequence. We show results on eight sequences of the dataset. Table 3.1 gives information about the datasets and the image pairs we evaluated for each of the selected sequences. We also provide the number of keypoints detected for each image and the Hessian threshold value to permit reproduction of our results. Figure 3.5 depicts some of the selected image pairs from the standard local descriptors evaluation dataset of Mikolajczyk and Schmid.

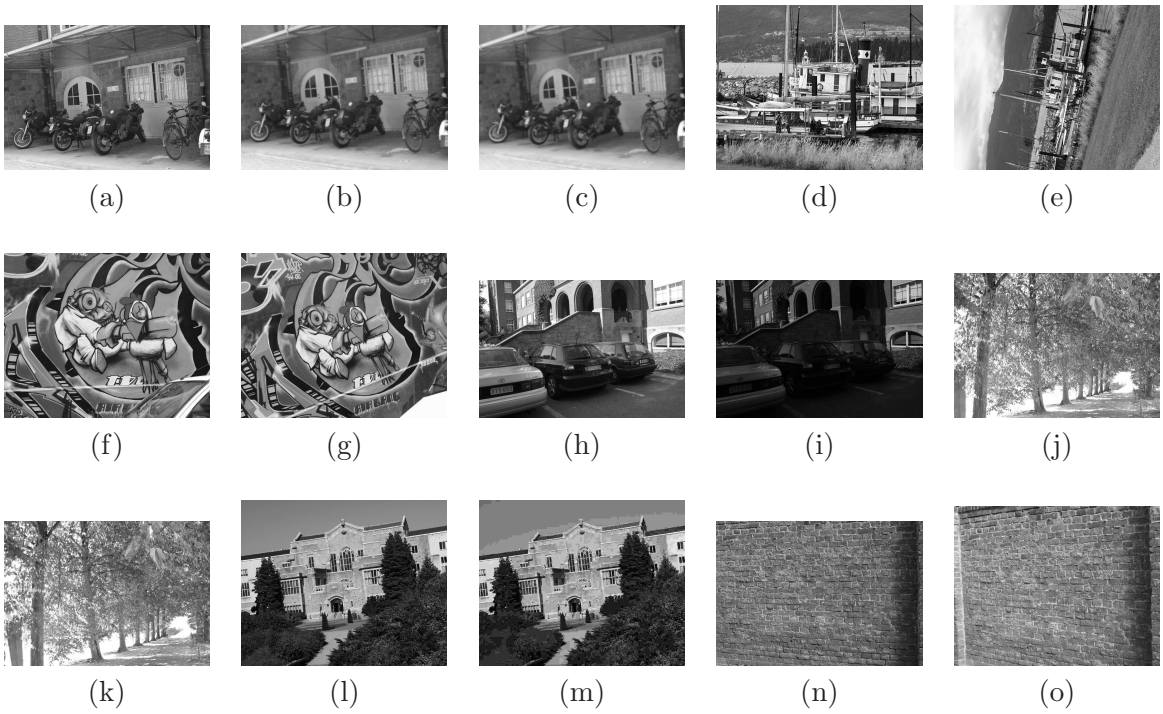


Figure 3.5: Image matching pairs from the Mikolajczyk and Schmid dataset. (a,b,c) Bikes dataset: Images 1,4,5. Image blurring (d,e) Boat dataset: Images 1,4. Zoom+Rotation (f,g) Graffiti dataset: Images 1,2. Changes in viewpoint (h,i) Leuven dataset: Images 1,5. Changes in lighting (j,k) Trees dataset. Images 1,3. Image blurring (l,m) UBC dataset: Images 1,5. JPEG compression (n,o) Wall dataset: Images 1,3. Changes in viewpoint.

Descriptors are evaluated by means of *recall versus 1 - precision* graphs as proposed in [Mikolajczyk and Schmid, 2005]. This criterion is based on the number of correct

[3] Available from http://www.robots.ox.ac.uk/$\tilde{\text{v}}$gg/research/affine/

matches and the number of false matches obtained for an image pair:

$$\begin{aligned} recall &= \frac{\#correct\ matches}{\#correspondences} \\ 1 - precision &= \frac{\#false\ matches}{\#all\ matches} \end{aligned} \tag{3.7}$$

The number of correct matches and correspondences is determined by the overlap error. Two regions (A, B) are deemed to correspond if the overlap error ϵ_0, defined as the error in the image area covered by the regions, is sufficiently small, as shown in Equation 3.8:

$$\epsilon_0 < 1 - \frac{A \cap H^T \cdot B \cdot H}{A \cup H^T \cdot B \cdot H} \tag{3.8}$$

In [Mikolajczyk and Schmid, 2005] there were shown some examples of the error in relative point location and recall considering different overlap errors. They found that for overlap errors smaller than 20% one can obtain the maximum number of correct matches. In addition, they showed that recall decreases with increasing overlap errors. Larger overlap errors result in a large number of correspondences and general low recall. Based on this, we decided to use an overlap error threshold of $\epsilon_0 < 20\%$, since we think this overlap error is reasonable for SfM applications, where you are only interested on very accurate matches. Furthermore, as in [Mikolajczyk and Schmid, 2004] we also impose that the error in relative point location for two corresponding regions has to be less than 2.5 pixels: $\|x_a - H \cdot x_b\| < 2.5$, where H is the homography between the images. Due to space limitations, we only show results on similarity threshold based matching, since this technique is better suited for representing the distribution of the descriptor in its feature space [Mikolajczyk and Schmid, 2005].

Dataset	**Image Change**	**Image N**	**# Keypoints Image 1**	**# Keypoints Image N**	**Hessian Threshold**
Bikes	Blur	4	2275	1538	0.0001
Bikes	Blur	5	2275	1210	0.0001
Boat	Zoom+Rotation	4	2676	1659	0.0001
Graffiti	Viewpoint	2	1229	1349	0.001
Leuven	Illumination	5	2705	2009	0.00001
Trees	Blur	3	3975	4072	0.0001
UBC	JPEG Compression	5	2106	2171	0.0001
Van Gogh	Rotation	10	864	782	0.00005
Van Gogh	Rotation	18	864	855	0.00005
Wall	Viewpoint	3	3974	3344	0.0001
Iguazu	Gaussian Noise	3	1603	2820	0.0001
Iguazu	Gaussian Noise	4	1603	3281	0.0001
Iguazu	Gaussian Noise	5	1603	3581	0.0001

Table 3.1: Sequences and image pairs used for image matching experiments: Image change, image number, keypoints number and Hessian threshold value.

Figure 3.6 depicts *recall versus 1-precision* graphs for the selected pairs of images. This figure suggests the following conclusions:

- In general, among the upright evaluation of the descriptors, GU-SURF descriptors perform much better than its competitors, especially for high precision values, with sometimes more than 20% improvement in recall for the same level of precision with

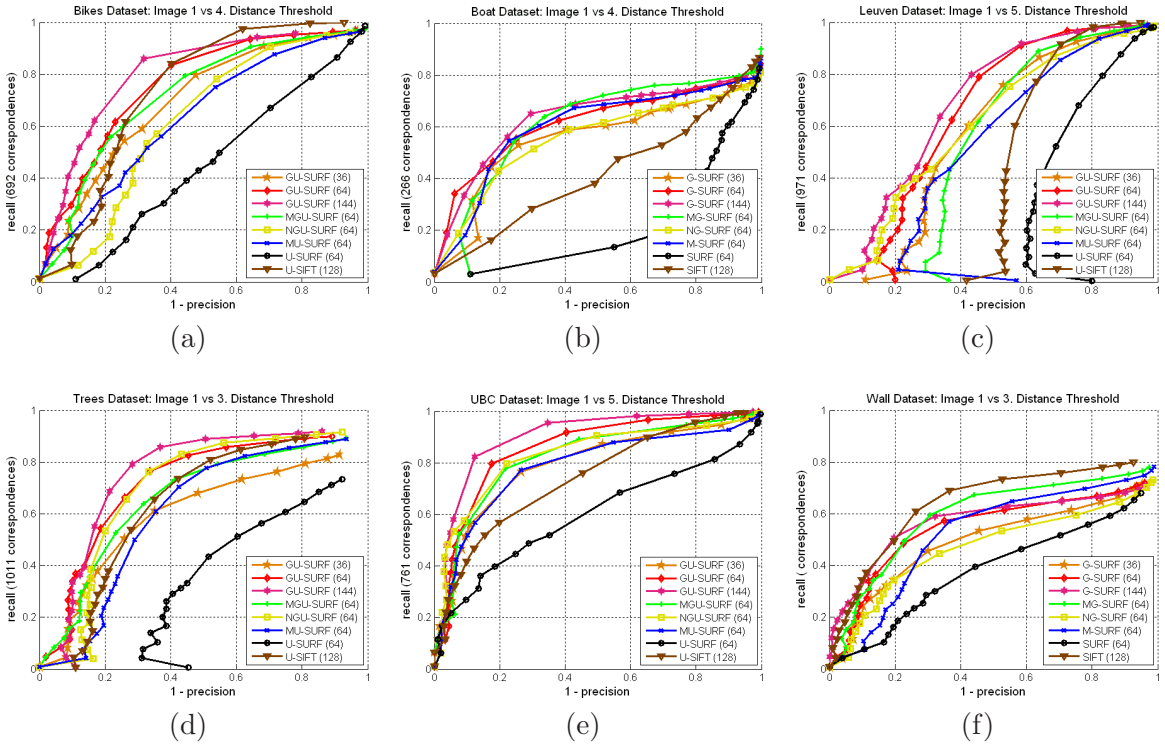


Figure 3.6: Image matching experiments: Recall versus 1-precision graphs, Similarity threshold based matching. (a) Bikes 1 vs 4 (b) Boat 1 vs 4 (c) Leuven 1 vs 5 (d) Trees 1 vs 3 (e) UBC 1 vs 5 (f) Wall 1 vs 3. Best viewed in color.

respect to MU-SURF (64) and U-SIFT (128) (e.g. Leuven, Bikes and Trees datasets), and even much more improvement with respect to U-SURF (64). GU-SURF (144) was the descriptor that normally achieved the highest recall for all the experiments, followed close by GU-SURF (64). GU-SURF (36) also exhibits good performance, on occasions even better than higher dimensional descriptors such as U-SIFT (128) or MU-SURF (64).

- In the upright evaluation of the descriptors, one can obtain higher recall rates by means of descriptors that do not have any kind of Gaussian weighting or subregions overlap. For example, we can observe this effect between NGU-SURF (64) and U-SURF (64), where the only difference between both descriptors is the Gaussian weighting step. Furthermore, we can see that between GU-SURF (64) and MGU-SURF (64), GU-SURF (64) obtained higher recall values than when using the modified version of the descriptors.

- With respect to the rotation invariant version of the descriptors, in these cases, the modified descriptor version plays a more important role. The use of two Gaussian weighting steps and subregions overlap, yield a more robust descriptor against large geometric deformations and non-planar rotations. In addition, the Gaussian weighting helps in reducing possible computation errors when interpolating Haar wavelets responses according to a dominant orientation. This interpolation of the responses, is not necessary in the case of gauge derivatives, since by definition they are rotation invariant. We can observe that MG-SURF (64) obtained slightly better results compared to M-SURF (64) and SIFT (128) for the Boat dataset (Zoom+Rotation). For the Wall dataset (changes in viewpoint), SIFT (128) was the descriptor that

obtained better results, and MG-SURF (64) obtained better results compared to M-SURF (64), especially for high precision values.

- When comparing gauge-based descriptors and first-order local derivatives descriptors, we can observe that gauge-based descriptors always obtained higher recall values, both in the standard and upright form of the descriptors. We can observe this behaviour between G-SURF (64) versus NG-SURF (64), and MG-SURF (64) versus M-SURF (64) and also considering the upright version of the descriptors. One of the reasons why gauge derivatives obtained better performance is because they are intrinsically weighted by the strength of the gradient L_w per pixel, and thus the resulting decriptor exhibits a higher discriminative power.

- In all the sequences the worst results were obtained by the OpenSURF's SURF implementation, which uses the single Gaussian weighting scheme that gives poor results.

Evaluation under image noise transformations

In this section, we evaluate the performance of the descriptors under image noise transformations. For this purpose, we created a new dataset named *Iguazu*. This dataset consists of 6 images, and the image transformation in this case is the progressive addition of random Gaussian noise. For each pixel of the transformed images, we add random Gaussian noise with increasing variance considering grey scale value images. The noise variances for each of the images are the following: Image 2 $\pm$2.55, Image 3 $\pm$12.75, Image 4 $\pm$15.00, Image 5 $\pm$51.0 and Image 6 $\pm$102.00, considering that the grey value of each pixel in the image ranges from 0 to 255. Noisy images are very common in fields such as biomedical imaging [ter Haar Romeny, 2003] and other research areas such as Synthetic Aperture RADAR imaging (SAR) [Liu and Wang, 2009]. We think that for these applications, a descriptor which is robust to different noise settings is very desirable. Figure 3.7 depicts three images of the Iguazu dataset for image random noise transformations, and the *recall versus 1-precision* for three image pairs of the sequence.

According to the graphs, we can observe than for this dataset, the difference between gauge-derivatives and first-order local derivatives based descriptors is much more important than for the previous image transformations evaluation. The best results were obtained again with the GU-SURF (144) descriptor. In this experiment, U-SIFT (128) obtained also good results, with higher recall values than MU-SURF (64), U-SURF (64) and NGU-SURF (64). Notice that in these experiments, GU-SURF (36) obtained better results for the three image pairs than MU-SURF (64), U-SURF (64) and NGU-SURF (64). This is remarkable, due to the low dimension of the descriptor, and this clearly stands out the disciminative properties of gauge derivatives against first-order ones. The main reason why G-SURF descriptors exhibit good performance against image noise settings and higher recall rates compared to first-order local derivatives methods, is because G-SURF descriptors measure information about the amount of blurring (L_{ww}) and details or edge enhancing (L_{vv}) in the image at different scale levels.

Evaluation under pure rotation sequences

One of the nicest properties of gauge derivatives, is their invariance against rotation. In this section, we compare G-SURF descriptors against first-order local derivatives descriptors, to stand out the rotation invariance properties of gauge derivatives. For this

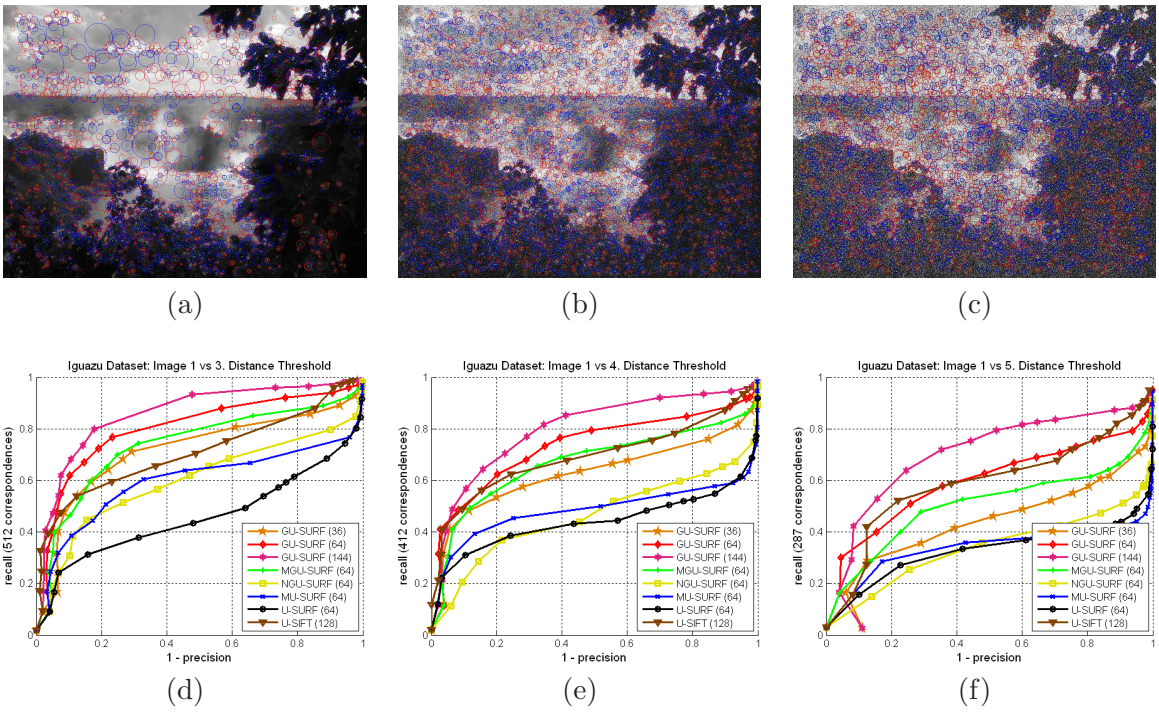


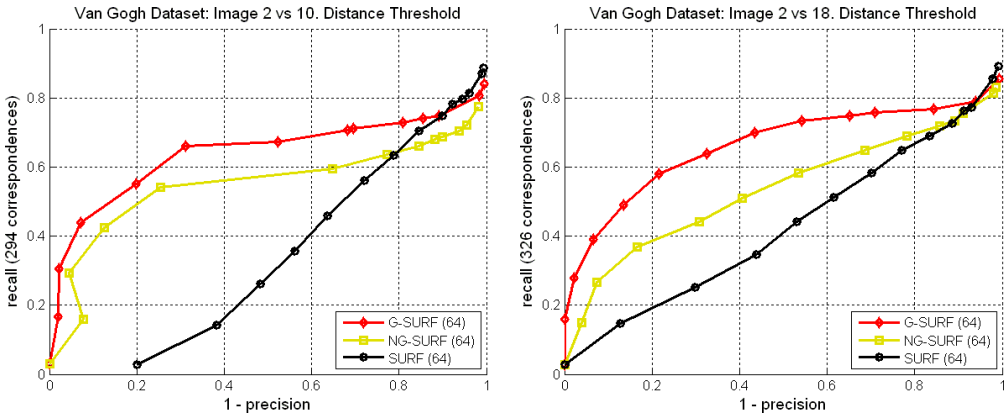
Figure 3.7: In the first row (a,b,c), we show some images from the Iguazu dataset, with incremetally increasing random Gaussian noise values per image. Notice that when severe random noise is added to the image, the number of detected blobs increases, mainly at small scales. The detected keypoints are shown in red or blue depending on the sign of the Laplacian. (a) Iguazu 1 (b) Iguazu 3 (c) Iguazu 5. In the second row (d,e,f), Image matching experiments: Recall versus 1-precision graphs, Similarity threshold based matching. (d) Iguazu 1 vs 3 (e) Iguazu 1 vs 4 (f) Iguazu 1 vs 5. Best viewed in color.

purpose, we decided to use the Van Gogh sequence that consists on pure rotation image transformations. This sequence and the ground truth homographies relating the images can be downloaded from Mykolajczyk's older webpage[4]. In order to show the performance of G-SURF descriptor under pure rotation transformation, we evaluated two image pairs from the Van Gogh sequence. Figure 3.8 depicts the reference image and the rest two images that are related by a pure rotation of 90° and 180° with respect to the reference image.

(a) Image 2 (b) Image 10 (c) Image 18

Figure 3.8: Van Gogh rotation dataset: Images 2 and 10 are related by a pure rotation of 90°, whereas Images 2 and 18 are related by a pure rotation of 180°.

Figure 3.9 depicts the *recall versus 1-precision* for the selected image pairs from the Van Gogh dataset. In this experiment, we compared only G-SURF (64) versus NG-SURF (64) and SURF (64). According to the results, we can observe that for some points in the graphs, by using G-SURF (64), there is an improvement in recall about the 20% with respect to NG-SURF (64) and approximately the double, 40%, with respect to SURF (64) for the same precision values. These results stand out the effect of the nice rotation invariance property of gauge-derivatives in the matching capabilities of the descriptors.



(a) (b)

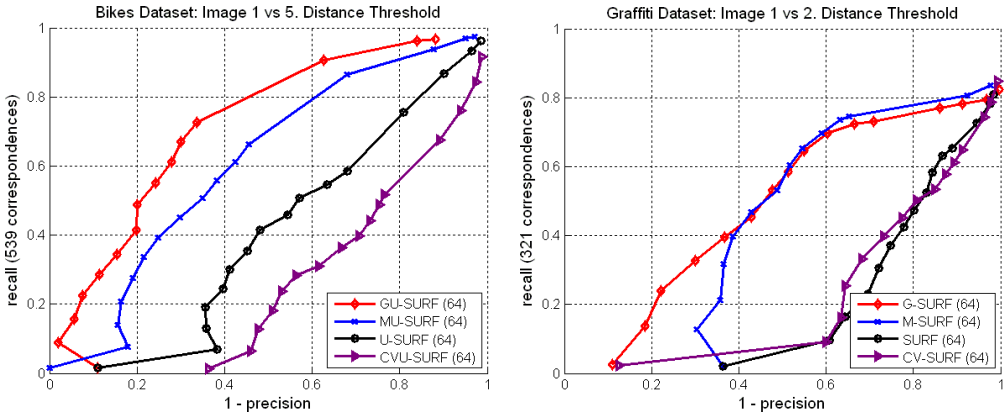
Figure 3.9: Image matching experiments: Recall versus 1-precision graphs, Similarity threshold based matching. (a) Van Gogh 2 vs 10 (b) Van Gogh 2 vs 18. Best viewed in color.

[4]http://lear.inrialpes.fr/people/mikolajczyk/Database/rotation.html

3.5.2 Comparison to OpenCV

In this section, we also compare our G-SURF descriptors with the latest OpenCV[5] implementation of the SURF descriptor. According to [Calonder et al., 2010], OpenCV's SURF implementation has become a de facto standard for fast-to-compute descriptors. However as we will show in our results, the descriptor performance is poor and much lower compared to the default OpenSURF's M-SURF descriptor. This low performance is because the SURF implementation in OpenCV uses also the single Gaussian weighting scheme as proposed in the original SURF paper [Bay et al., 2008].

Figure 3.10 depicts *recall versus 1-precision* graphs for two image pairs from the Bikes and Graffiti datasets. In this experiment, we compare G-SURF (64) with respect to M-SURF (64), SURF (64) and CV-SURF (64) both in the upright and standard forms of the descriptors. We denote by CV-SURF, the OpenCV implementation of the SURF descriptor using the single weighting scheme as described in Section 3.3. According to the results, we can see that the OpenCV implementation gives poor results, comparable to SURF (64) OpenSURF's implementation, since both algorithms use the mentioned single Gaussian weighting scheme. We can appreciate a huge difference in recall with respect to G-SURF (64) and M-SURF (64).



(a) (b)

Figure 3.10: Image matching experiments: Recall versus 1-precision graphs, Similarity threshold based matching. (a) Bikes 1 vs 5 (b) Graffiti 1 vs 2. Best viewed in color.

3.5.3 Application to 3D Structure from Motion

In this section, we evaluate the performance of G-SURF based descriptors in large-scale 3D SfM applications. In particular, we use the learning local image descriptors dataset from [Brown et al., 2011]. In the mentioned work, Brown et al. proposed a framework for learning dense local image descriptors from training data using 3D correspondences from large-scale SfM datasets. For generating ground truth image correspondences between real interest points, the authors used multi-view stereo matching techniques [Goesele et al., 2006, 2007] that allow to obtain very accurate correspondences between 3D points.

The available dataset consists on several scale and orientation normalized 64×64 image patches centered around detected Harris corners or Difference of Gaussian (DoG) [Lowe, 2004] features. Those patches were extracted from real 3D points of large-scale SfM

[5]Available from http://sourceforge.net/projects/opencvlibrary/

scenarios. In our evaluation, we used 40,000 patch pairs centered on detected Harris corners from which the 50% are match pairs and the rest 50% are considered non-match pairs. We attach the set of matches/non-matches image patches used for the evaluation as a supplementary material of the thesis. In the evaluation framework of Brown et al., two patches are considered to be a match if the detected interest points are within 5 pixels in position, 0.25 octaves in scale and $\pi/8$ radians in angle. Figure 3.11 depicts some of the pre-defined match, non-match pairs from the Liberty dataset.

Figure 3.11: Some of the predefined match, non-match pairs from the Liberty dataset. Each row shows 3 pairs of image patches and the two image patches in each pair are shown in the same column. (a) Match pairs (b) Non-match pairs.

We performed an evaluation of the upright version of the descriptors U-SURF (64), MU-SURF (64), GU-SURF (64), MGU-SURF (64), NGU-SURF (64) and U-SIFT (128) for both the Liberty and Notre Dame datasets. We chose a scale of 2.5 pixels to make sure that no Haar wavelet responses were computed outside the bounds of the image patch. For all the image pairs in the evaluation set, we computed the distance between descriptors and by means of sweeping a threshold on the descriptor distance, we were able to generate ROC curves. Figure 3.12 depicts the ROC curves for the Liberty dataset, whereas Figure 3.13 depicts the ROC curves for the Notre Dame dataset.

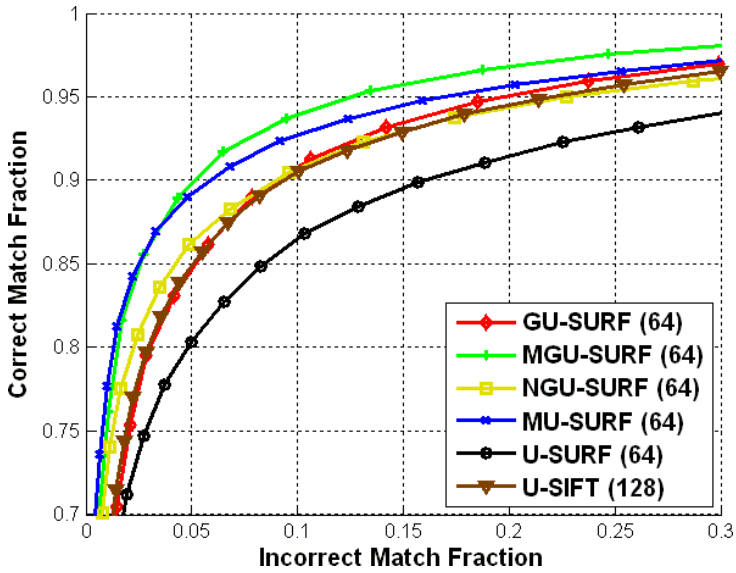


Figure 3.12: ROC curves for local image descriptors. Liberty dataset. Best viewed in color.

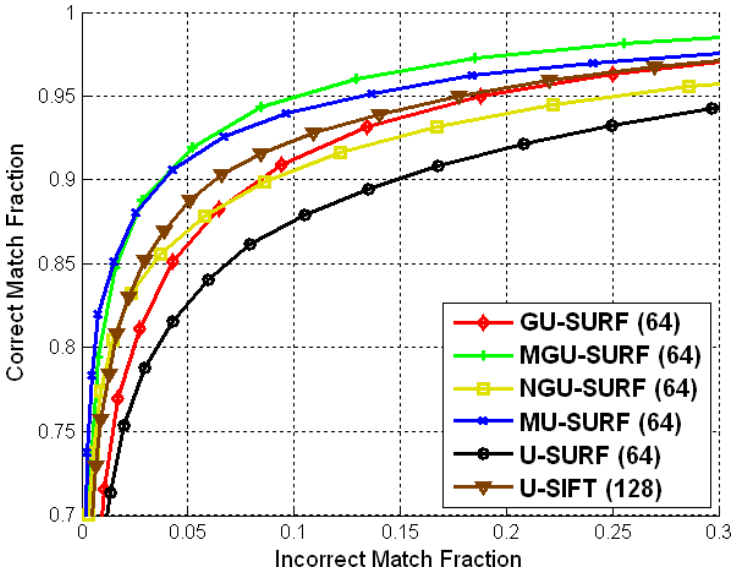


Figure 3.13: ROC curves for local image descriptors. Notre Dame dataset. Best viewed in color.

In addition, in Table 3.2 we also show results in terms of the 95% error rate which is the percent of incorrect matches obtained when the 95% of the true matches are found.

Descriptor	Liberty	Notre Dame
GU-SURF (64)	19.78	18.95
MGU-SURF (64)	**12.55**	**10.19**
NGU-SURF (64)	22.95	25.22
MU-SURF (64)	16.88	13.17
U-SURF (64)	36.49	34.18
U-SIFT (128)	21.92	17.75

Table 3.2: Local image descriptors results. 95% error rates, with the number of descriptor dimension in parenthesis.

According to the results, we can observe that the lowest incorrect match fraction rate for the 95% recognition rates was obtained by the MGU-SURF (64) descriptor. This descriptor uses the same square grid configuration, two Gaussian weighting steps and subregions overlap as proposed in [Agrawal et al., 2008] for the MU-SURF descriptor. In typical large-scale 3D SfM scenarios, there exist non-planar transformations and illumination changes resulting from viewing a truly 3D scene [Brown et al., 2011]. In addition, second-order derivatives are more sensitive to perspective or affine changes than first-order ones. Therefore, in those scenarios where the affine changes or changes on perspective are significant, the two-steps Gaussian weighting and subregions overlap seem to have a good effect on the descriptor performance. This is the reason why in this evaluation we obtained better results for MGU-SURF (64) and MU-SURF (64) against GU-SURF (64) and NGU-SURF (64), that do not use any kind of subregion overlap or Gaussian weighting steps. U-SIFT (128) also obtained good results, always better than NGU-SURF (64) and very similar results compared to GU-SURF (64), slightly better for the Notre Dame dataset. U-SIFT (128) also uses bilinear interpolation between the bins of the descriptor histogram [Lowe, 2004]. When comparing, gauge-derivatives based descriptors and first-

order local derivatives ones, without any subregion overlap nor any Gaussian weighting step, we can observe that GU-SURF (64) obtained much better results than NGU-SURF (64). As expected, the worst results were obtained for the U-SURF (64) descriptor, since in this descriptor configuration the single Gaussian weighting step smoothes in a very high degree the descriptor information, yielding in lower recognition rates.

Besides, in the OpenGSURF library, the user can choose between the SIFT-style clipping normalization or unit vector normalization of the descriptor. This normalization can a have a big impact on the matching performance of the descriptors, as demonstrated in [Hua et al., 2007; Brown et al., 2011], where one can obtain lower error rates considering the SIFT-style clipping normalization. However, in order to avoid the influence of this normalization style in our results, we just show results using the standard unit vector normalization, except for the SIFT descriptor, in which we use its default SIFT-style clipping normalization.

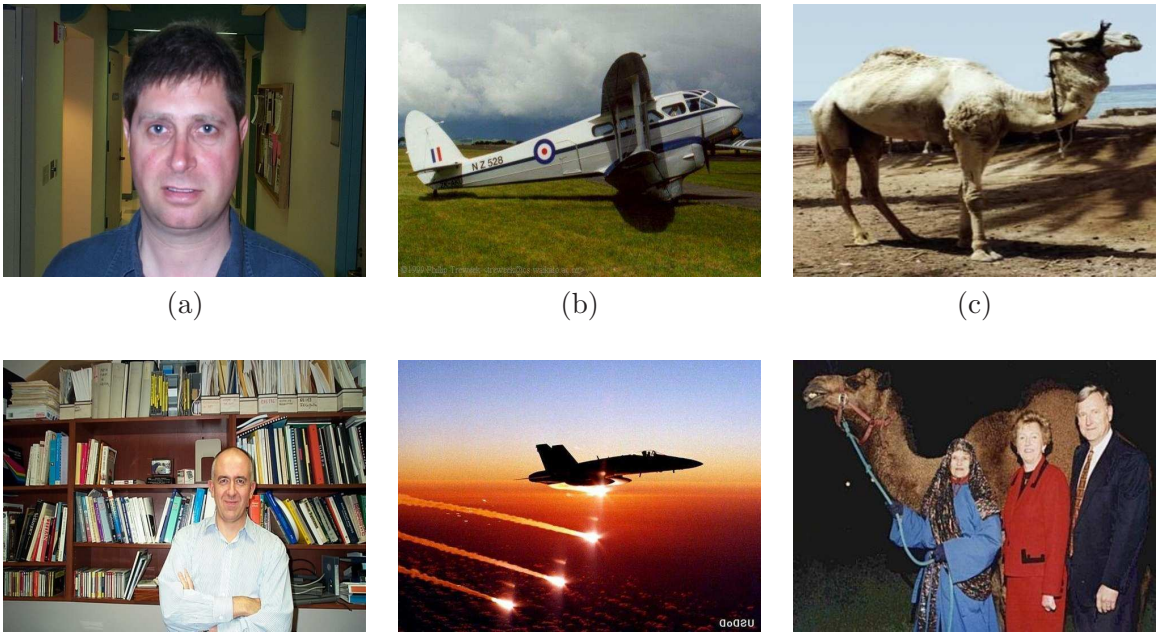
3.5.4 Application to Visual Categorization Problems

In this experiment, we show that G-SURF based descriptors can be used efficiently in typical visual image categorization or object recognition problems. Bay et al. have shown in previous works [Bay et al., 2006a,b, 2008] that SURF-based descriptors can be used efficiently in these kind of applications. Nowadays, SURF or SIFT invariant descriptors are of common use in typical visual categorization or object recognition schemes [Csurka et al., 2004]. In a similar way to [Fergus et al., 2003], we performed our tests considering the Caltech faces, airplanes and camels dataset [6]. Firstly, we resized all the images to a 640×480 resolution and selected the 25% of all the images (randomly distributed among the three categories) for training. The rest of the images was used for test evaluation.

Even though this is a simple visual categorization problem, we want to evaluate if G-SURF based descriptors can exhibit higher recognition rates than traditional first-order spatial derivatives based approaches due to the extra invariance offered by using gauge derivatives. Figure 3.14 depicts three image pairs of the different categories that we used in our evaluation. In particular, we can expect a higher confusion between the faces and camels categories. This is because in some images of the camels dataset we can observe some human faces as shown for example in Figure 3.14(f), and also that camel and human faces share some degree of similarity.

In order to perform an evaluation of the different local descriptors, we used our own implementation of the visual bag of keypoints method described in [Csurka et al., 2004]. This implementation has been successfully tested before in an occupant monitoring system based on visual categorization [Yebes et al., 2011]. Basically, we used the standard Fast-Hessian detector to detect features of interest at different scale levels, and then we computed different local descriptors. In this experiment, we only show a comparison between 64 dimensional descriptors in its upright form (U-SURF, MU-SURF, GU-SURF, NGU-SURF). Once the descriptors are extracted, the visual vocabulary is constructed by means of the standard *k-means* clustering scheme [Bishop, 2007]. This clustering algorithm proceeds by iterated assignments of keypoints descriptors to their closest cluster centers and recomputation of the cluster centers. The selection of the number of clusters and the initialization of the centers are of great importance in the performance of the algorithm. Finally, the visual categorization is done by using a simple Näive Bayes classifier [Lewis, 1998]. In order to reduce the influence of the clustering method on the final

[6] http://www.vision.caltech.edu/html-files/archive.html



(a) (b) (c)

(d) (e) (f)

Figure 3.14: Three pairs of images from the Caltech dataset. (a,d) Faces (b,e) Airplanes (c,f) Camels. Notice the possible confusion between the faces and camels categories.

results, we decided to use a small number of clusters $k = 20$ and performed a random initialization of the cluster centers. To avoid cluster initialization problems, the clusters were randomly initialized ten times in each of the experiments, reporting categorization results just for the cluster initialization that obtained minimum compactness measure.

Tables 3.3, 3.4, 3.5, 3.6 show information about the performance of each of the different descriptors in the test evaluation. Similar to [Csurka et al., 2004], we used three performance measures to evaluate the performance on visual categorization: the confusion matrix, the overall error rate and the mean ranks. For more information about the meaning of these performance measures, we recommend the reader to check the experiments section in [Csurka et al., 2004].

True Classes	**Faces**	**Airplanes**	**Camels**
Faces	82.6531	0.8714	19.0000
Airplanes	1.3605	91.5033	12.0000
Camels	15.9864	7.6252	69.0000
Mean Ranks	1.1973	1.1154	1.3100
Overall Error Rate		0.1352	

Table 3.3: Confusion matrix, mean ranks and overall error rate for U-SURF (64).

With respect to the confusion matrix, we can observe that GU-SURF (64) descriptor obtained higher recognition rates for the faces (85.3741%) and camels (72.0000%) categories. However, the MU-SURF (64) descriptor obtained a higher recognition rate for the airplanes (93.68%) dataset. In the same way, GU-SURF (64) obtained the lowest mean ranks for the faces (1.1564) and camels (1.2800) datasets and MU-SURF (64) obtained the lowest one for the airplanes dataset (1.0824). Regarding the overall error rate, GU-SURF

True Classes	Faces	Airplanes	Camels
Faces	79.2517	0.3267	25.5000
Airplanes	0.6802	93.6819	7.0000
Camels	20.0680	5.9912	67.5000
Mean Ranks	1.2142	**1.0824**	1.3250
Overall Error Rate		0.1303	

Table 3.4: Confusion matrix, mean ranks and overall error rate for MU-SURF (64).

True Classes	Faces	Airplanes	Camels
Faces	**85.3741**	0.2178	22.5000
Airplanes	0.3401	91.8301	5.5000
Camels	14.2857	7.9520	**72.0000**
Mean Ranks	**1.1564**	1.1132	**1.2800**
Overall Error Rate		**0.1232**	

Table 3.5: Confusion matrix, mean ranks and overall error rate for GU-SURF (64).

True Classes	Faces	Airplanes	Camels
Faces	80.6122	0.3267	20.0000
Airplanes	1.36054	93.3551	10.0000
Camels	18.0272	6.31808	70.0000
Mean Ranks	1.2074	1.0882	1.3
Overall Error Rate		0.1260	

Table 3.6: Confusion matrix, mean ranks and overall error rate for NGU-SURF (64).

(64) was the descriptor that achieved the lowest error (0.1232). There is a reduction in the overall error rate of the 8.88% with respect to U-SURF (64), 5.45% with respect to MU-SURF (64) and 2.22% with respect to NGU-SURF (64). Even though the experimental evaluation was a simple visual categorization problem, we can conclude that G-SURF based descriptors can be used efficiently in these visual recognition schemes. In addition, G-SURF descriptors can also obtain lower error rates and higher recognition rates than traditional approaches that are based only on first-order local derivatives.

3.5.5 Implementation Details and Timing Evaluation

In this section, we describe some implementation details of G-SURF descriptors and perform a timing evaluation. One of the criticisms about using second-order derivatives in the context of local descriptors, is the higher computational cost that sometimes is not accompanied by a better performance. In this section, we show that by means of using gauge derivatives we can obtain much better performance than first-order based methods with comparable computational cost. Table 3.7 shows timing results for descriptor computation and also the number of the most important operations in the process of building the upright SURF based descriptors. All timing results were obtained on an Intel i7 2.8GHz computer.

Descriptor	**U-SURF**	**MU-SURF**	**MGU-SURF**	**GU-SURF**	**GU-SURF**	**GU-SURF**
Dimension	**64**	**64**	**64**	**36**	**64**	**144**
# 1st-Order Wavelets	800	2592	2592	648	800	1152
# 2nd-Order Wavelets	0	0	3888	972	1200	1728
# Gaussian Weights	800	2608	2608	0	0	0
Square area	20×20	24×24	24×24	18×18	20×20	24×24
# Integral Image Areas	1600	5184	15552	3888	4800	6912
Time (ms)	0.03	0.16	0.30	0.06	0.07	0.10

Table 3.7: Descriptor Building Process: Number of operations, square area and average computation time per descriptor keypoint.

In Table 3.7, the number of integral image areas means the number of areas that we have to obtain in order to compute the descriptor. Based on OpenSURF's implementation details [Evans, 2009], one can estimate first-order Haar wavelets L_x, L_y with just the difference of two areas of the integral image for each of the first-order wavelets. For each of the second-order Haar wavelets L_{xx}, L_{yy} it is necessary to compute two areas of the integral image and sum these areas in a proper way. Finally, the most consuming Haar wavelet is L_{xy}, since it requires the computation of 4 areas of the integral image. For example, for the U-SURF (64) case, the total number of areas of the integral image that we need to compute is: $(4 \times 4) \cdot (5 \times 5) \cdot (2 + 2) = 1600$. Due to the extra-padding of $2s$, the MU-SURF (64) case yields: $(4 \times 4) \cdot (9 \times 9) \cdot (2 + 2) = 5184$. On the other hand, the GU-SURF (64) case yields: $(4 \times 4) \cdot (5 \times 5) \cdot (2 + 2 + 2 + 2 + 4) = 4800$. However, the core observation is that for the GU-SURF (64) descriptor one can obtain substantial speed-up for those points in the rectangular grid where the gradient is equal to zero. For those cases we do not need to compute the second-order wavelets, since gauge coordinates are not defined for these points. This corresponds to regions of the images of equal value,

and therefore these regions are non-Morse.

Using the same settings as described in Table 3.1, we can show the fraction of non-Morse points among all the points where Haar wavelets were evaluated. For example, for the following images the ratio is: Leuven Image 1 (17.96%), Bikes Image 1 (17.73%) and Iguazu Image 1 (32.43%). Another computational advantage of the G-SURF descriptor is that it is not necessary to interpolate the Haar wavelet responses with respect to a dominant orientation, since gauge derivatives are rotation invariant.

As explained above, the number of operations for U-SURF (64) is the smallest, yielding a small computation time per descriptor, but the performance is the worst compared to the other SURF-based cases. NGU-SURF (64) descriptor has similar computation times than the U-SURF descriptor, with the advantage that no Gaussian weighting operations are necessary and exhibiting much better performance. The modified version of the descriptors introduces more computations in the descriptor building process, since the square area is $24s \times 24s$. This yields higher computation times per descriptor. In particular, for the MGU-SURF (64) descriptor, the number of integral image areas is the highest (15552), and also the associated computation time per descriptor (0.30 ms). However, this descriptor only offers small advantages in performance against GU-SURF (36), GU-SURF (64) and GU-SURF (144) when we have sequences with strong changes in viewpoints and non-planar rotations (e.g. Wall, Graffiti, Liberty and Notre Dame datasets). In addition, GU-SURF (36), GU-SURF (64) and GU-SURF (144) are faster to compute than MU-SURF (64) and also exhibit much better performance. For the U-SIFT (128) descriptor, we obtained an average computation time per keypoint of 0.42 ms. Besides, for any SIFT-based descriptor one needs to compute the Gaussian scale space since the gradients are precomputed for all levels of the pyramid [Lowe, 2004]. Pre-computing the scale space is a highly consuming task in contrast to the fast integral image computation. We obtained a computation time of 186 ms for the SIFT scale space generation, whereas for the SURF integral image we obtained 2.62 ms. For the CVU-SURF case, we obtained an average computation time per keypoint of 0.05 ms.

According to these results, it is clear that image matching using the G-SURF descriptors can be accomplished in real-time, with high matching performance. For example, we think that GU-SURF (36) and GU-SURF (64) are of special interest to be used efficiently in real-time SfM and SLAM applications due to excellent matching performance and computational efficiency.

3.6 Conclusions and Future Work

We have presented a new family of multiscale local descriptors, a novel high performance SURF-inspired set of descriptors based on gauge coordinates which are easy to implement but are theoretically and intuitively highly appealing. Image matching quality is considerably improved relative to standard SURF and other state of the art techniques, especially for those scenarios where the image transformation is small in terms of change in viewpoint or the image transformation is related to blur, rotation, changes in lighting, JPEG compression or random Gaussian noise. Our upright descriptors GU-SURF (64) and GU-SURF (36) are highly suited to SfM and SLAM applications due to excellent matching performance and computational efficiency. Furthermore, the rotation invariant form of the descriptors is not necessary in applications where the camera only rotates around its vertical axis, which is the typical case of visual odometry [Nistér et al., 2004; Kaess et al., 2009] or visual SLAM [Davison et al., 2007] applications. We also showed

successful results of our familiy of descriptors in large-scale 3D SfM applications and visual categorization problems.

Another important conclusion that we showed in this chapter, is that descriptors based on gauge-derivatives can exhibit much higher performance than first-order local derivatives based descriptors. This is possible, due to the extra invariance offered by gauge-derivatives and also our G-SURF descriptors have comparable computational cost with respect to other approaches.

As future work we are interested in testing the usefulness of G-SURF descriptors for more challenging object recognition tasks (e.g. The PASCAL Visual Object Classes Challenge). In addition, we also plan to incorporate our descriptors into real-time SfM applications and evaluate them in loop closure detection problems such as in [Angeli et al., 2008]. Future work will aim at optimising the code for additional speed up and also we will exploit the use of gauge coordinates in the detection of features in non-linear scale spaces. Moreover, we would like to introduce our gauge-based descriptors on a DAISY-like framework [Tola et al., 2010] for performance evaluation on different computer vision applications.

According to the obtained results and other successful approaches such as *geometric blur*, we hope that in the next future we can break with the standard scale-space paradigm in feature detection and description algorithms. In the standard scale-space paradigm the true location of a boundary at a coarse scale is not directly available in the coarse scale image. The reason for this is simply because Gaussian blurring does not respect the natural boundaries of objects. We believe that introducing new invariant features that fully exploit non-linear diffusion scale spaces (both in detection and local description of features) can represent step forward improvements on traditional image matching and object recognition applications.

Chapter 4

Visibility Learning in Large-Scale Urban Environment

Large-scale 3D applications, such as robot localization and Structure from Motion (SfM), have recently become a more and more popular topic in robotics and computer vision. With the introduction of handheld devices equipped with cameras, accelerometers and GPS sensors (e.g., mobile phones), extending these 3D applications to such devices is very much desired. However, most of the existing approaches are designed for offline processing due to their high computational cost.

One of the most computationally expensive steps in vision-based localization is data association, in which matching candidates between a large map of 3D points and 2D features are retrieved and then usually validated by geometric constraints using RANdom SAmple Consensus (RANSAC) [Fischler and Bolles, 1981]. For the environments with highly repetitive textures, such as cities, the traditional methods mainly depend on the appearance information, which results in a very large number of matching candidates due to the ambiguities introduced by visually similar features [Schindler et al., 2008].

Visibility prediction is a commonly used technique to greatly reduce the ambiguities and speed up the data association by making an accurate and robust prediction of the most likely visible 3D points for a given camera pose [Wuest et al., 2007; Alcantarilla et al., 2010b]. More specifically, in the problem of visibility prediction, we want to determine whether a certain 3D point in the known 3D environment can be perceived by a given query camera. In this chapter, we propose a novel way to predict the visibility of 3D points efficiently and robustly.

There are two main types of information that are used for aiding data association. First, we have geo-spatial information, which is widely used in tracking based localization approaches, such as the works by Klein and Murray [2007] and Davison et al. [2007]. These methods can be quite efficient in a limited-size indoor space but do not generalize well to large outdoor scenes. Moreover, they need small baselines between the consecutive images hence are not proper for the wide-baseline images taken by most of the handheld devices.

Second, the appearance of the 3D structure is another important source of information, which is typically exploited by using feature descriptors such as SIFT [Lowe, 2004]. However, for very large databases, the computation time to match the features in the current image to the 3D points in the database can be prohibitively expensive, and even the most advanced algorithms [Nistér and Stewénius, 2006; Schindler et al., 2007] do not run in real-time. One important reason is that those methods tend to ignore the previous visibility information, i.e. the visible 3D points and their corresponding cameras poses.

This information can directly lead to a very fast prediction based on the weak priors of the current pose and the appearance of the corresponding images without involving expensive descriptors computations. Figure 4.1 depicts a graphical example of the problem of data association for large-scale vision-based applications.

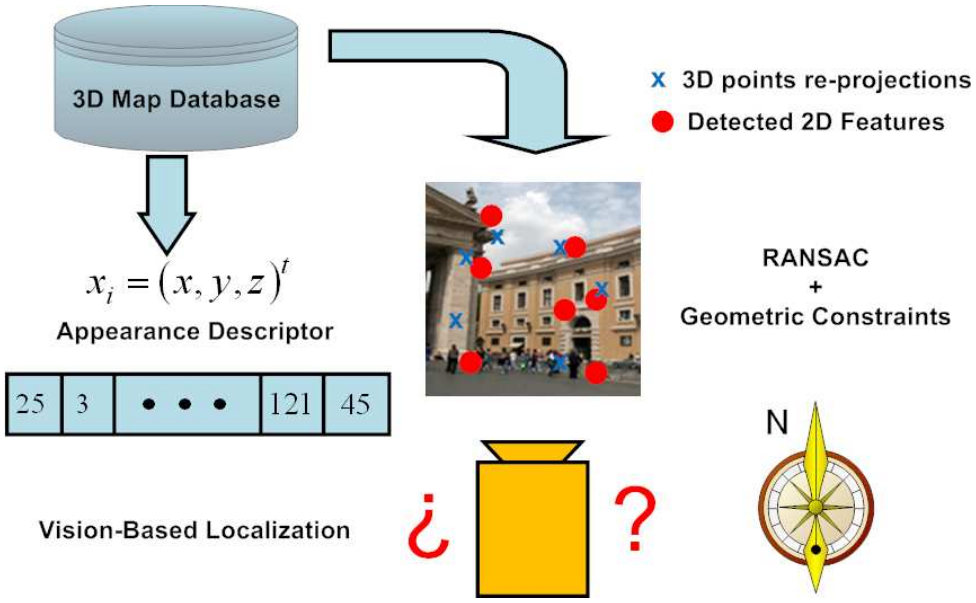


Figure 4.1: The problem of data association in large-scale vision-based localization applications.

In this chapter, we introduce a memory-based learning framework to predict, for each 3D point, its visibility with respect to a query camera pose. Our approach memorizes camera poses for each 3D point in the dataset and uses a non-parametric model to efficiently capture the visibility of landmarks in cluttered large-scale urban environments. Figure 4.2 [1] shows an example featuring a large city-scale 3D reconstruction, comprising of recovered 3D points and camera poses. A new camera with noisy pose prior is queried, and our algorithm predicts the visibility of the 3D points for that camera by fusing the information from the nearby cameras, exploiting all the geometric information available from the 3D environment. In this way, we can considerably improve the data association between the large map of 3D points and the features in the current image, yielding higher quality matches than conventional approaches.

In the remainder of the chapter, we describe the related work in Section 4.1 and introduce our probabilistic modeling of visibilities in Section 4.2. Then, we describe our metric learning framework in Section 4.3 and how to obtain a very fast visibility prediction is discussed in Section 4.4. In Section 4.5, we show the experimental results of our algorithm in a large-scale 3D reconstruction of the St. Peter's Basilica in Rome. Finally, main conclusions and future work are described in Section 4.6.

4.1 Related Work

Zhu et al. [2008] showed how to build an optimal 3D landmark database and how to use this database for real-time global localization. Through an intelligent subsampling of the landmark database based on geometry constraints, the size of the database was reduced

[1] The authors would like to thank Microsoft Photosynth for the St. Peter's Basilica dataset

Figure 4.2: Given a large city-scale 3D reconstruction, we predict the visible 3D points for a query camera view by fusing both geometric and appearance information from multiple neighbor cameras. Best viewed in color.

without sacrificing the accuracy in localization. In this approach landmarks are characterized by their appearance using Histogram of Oriented Gradients (HOG) [Dalal and Triggs, 2005]. The selection of putative matches for pose estimation relies vastly on appearance descriptors without exploiting the available geometry information from the reconstruction. Data association between current features in the image and 3D points in the database is done by means of a vocabulary tree, which is built by hierarchical *K-means-clustering* [Nistér and Stewénius, 2006]. In order to speed-up their hierarchical database search strategy, they performed two different pruning stages of the large database by means of geo-spatial constraints (3D camera pose location and its uncertainty) and via a vocabulary tree.

In [Wuest et al., 2007], the authors showed an augmented reality application in which they modeled for each feature the probability of the locations from where every feature can be tracked successfully. This probability is modeled by means of a finite set of Gaussian mixtures. The extension of this method for larger environments is difficult and computationally expensive. Moreover, since they only take into account the camera translation for their visibility prediction, their method only works with small scenarios where the degrees of possible camera orientations are very limited. As we will show later in our experimental results (see Section 4.5), the viewing direction or camera orientation has a much stronger impact than camera translation when predicting whether two cameras share common features or not. This is also one of the most important drawbacks of the mentioned work by Zhu et al. [2008], since their geo-spatial pruning only takes into account camera translation and its corresponding uncertainty, ignoring viewing directions.

Alcantarilla et al. [2010b] proposed a non-parametric framework for learning the visibility of reconstructed 3D points and used this visibility prediction for robust and fast vision-based localization under small indoor scenarios and baselines. Their algorithm learns a kernel function that measures the similarity between two camera poses, combining Euclidean distance and normalized dot product between camera translations and viewing directions respectively. They learn the kernel parameters by fitting a sigmoid function from the training data using nonlinear regression techniques [Bishop, 2007], ig-

noring the correlations between the different cues that are used in the metric. This makes the convergence of the nonlinear regression highly dependent on the initial values of the unknown parameters, and good guesses of the final value of the unknown kernel parameters are necessary for convergence and feature scaling problems [Atkeson et al., 1997].

In this chapter, we propose a new metric learning algorithm combining the Gaussian kernel and the Mahalanobis distance [Mahalanobis, 1936] and show results over large-scale urban environment 3D reconstructions. Our algorithm does not suffer from initialization problems and learns its own scaling, being more suitable for the addition of new cues to the proposed metric. We have further investigated the addition of new cues such as local histograms, which have been succesfully applied to location recognition problems recently [Torralba et al., 2003; Ni et al., 2008] .

4.2 Probabilistic Visibility Model

In the visibility prediction problem, we are interested in the posterior distribution of the visibility v_j for a certain 3D point x_j given the query camera pose θ, denoted as $P(v_j|\theta)$. For this purpose, we propose to use a form of lazy and memory-based learning technique known as *Locally Weighted Learning* [Atkeson et al., 1997]. This technique is a simple memory-based classification algorithm and can be implemented very efficiently. The idea is very simple: given the training data that consists of a set of reconstructed camera poses $\Theta = \{\theta_1 \dots \theta_N\}$, the 3D point cloud $X = \{x_1 \dots x_M\}$ and a query camera pose θ, we form a locally weighted average at the query point and take that as an estimate for $P(v_j|\theta)$ as follows:

$$P(v_j|\theta) \approx \frac{\sum_{i=1}^{N} k(\theta, \theta_i) \cdot v_j(\theta_i)}{\sum_{i=1}^{N} k(\theta, \theta_i)} \tag{4.1}$$

Now, we will explain the meaning of the functions $k(\cdot)$ and $v_j(\cdot)$ that are involved in the locally weighted average in Equation 4.1:

- The function $k(\theta, \theta_i)$ is a weighting function or kernel function that is used to calculate a weight, which emphasizes those camera poses that are similar to the query camera pose θ and deemphasizes very different camera poses. This makes sense, since if the camera at pose θ_i has already seen the point x_j, we may expect that the closer the query camera is to θ_i, the more likely it is to see point x_j.
- The function $v_j(\theta_i)$ just assigns a real value equal to 1 for those cases where a certain 3D point x_j is visible by a camera pose θ_i and 0 otherwise. Specifically, this function is a boolean random variable defined as follows:

$$v_j(\theta_i) = \begin{cases} 1 & \text{if } x_j \text{ is visible from camera } \theta_i \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

According to Equation 4.2, the final approximation of the locally weighted average for the visibility posterior can be derived as:

$$P(v_j = 1|\theta) \approx \frac{\sum_{i=1}^{N} k(\theta, \theta_i^{v_j=1})}{\sum_{i=1}^{N} k(\theta, \theta_i)} \tag{4.3}$$

where $\theta_i^{v_j=1}$ are the camera poses from the training data in which the 3D point x_j is visible.

4.3 Learning Kernel Functions

In this section, we show how to learn a proper distance metric or kernel function between two camera poses. A good distance metric (i.e. the kernel function k) is crucial for the overall performance of the proposed approach. Similar to the framework proposed by Weinberger and Tesauro [2007], our approach combines the Mahalanobis distance and the Gaussian kernel, and it applies to any distance-based kernel function with differentiable dependencies on parameters specifying the distance function. The Gaussian kernel is generally defined as follows:

$$G_{ij} = \frac{1}{\sigma\sqrt{2\pi}} \cdot \exp\left(-\frac{d(\vec{\theta}_i, \vec{\theta}_j)}{\sigma^2}\right) \tag{4.4}$$

where $d(\vec{\theta}_i, \vec{\theta}_j)$ is the squared distance between the vectors $\vec{\theta}_i$ and $\vec{\theta}_j$. These vectors encode different information related to the camera pose and its associated image view. We will explain in Section 4.3.1 and 4.3.2 how to define these input vectors. For simplification, we can drop the constant factor before the exponent in Equation 4.4 and absorb σ^2 in $d(\cdot)$, fixing $\sigma = 1$. A Mahalanobis distance is a generalization of the Euclidean metric that takes into account the correlations between variables and hence, it is scale-invariant. The Mahalanobis distance between two vectors $\vec{\theta}_i$ and $\vec{\theta}_j$ is defined as:

$$d(\vec{\theta}_i, \vec{\theta}_j) = (\vec{\theta}_i - \vec{\theta}_j)^T \mathbf{M} (\vec{\theta}_i - \vec{\theta}_j) \tag{4.5}$$

where $\mathbf{M}$ can be any symmetric positive semidefinite real matrix, i.e. a matrix whose eigenvalues are all non-negative. If $\mathbf{M}$ is equal to the identity matrix, the Mahalanobis distance reduces to the Euclidean distance. Unfortunately, learning the matrix $\mathbf{M}$ directly requires enforcing a positive semidefinite constraint in the optimization problem, which is highly non-linear and is also expensive to solve. One way to get rid of such an expensive constraint is to decompose $\mathbf{M}$ as:

$$\mathbf{M} = \mathbf{A}^T \mathbf{A} \tag{4.6}$$

where $\mathbf{A}$ is a full rank matrix. By substituting Equation 4.6 into 4.5, we can express the Mahalanobis distance as the Euclidean distance after the mapping $\vec{\theta} \rightarrow \mathbf{A}\vec{\theta}$:

$$d(\vec{\theta}_i, \vec{\theta}_j) = \left\| \mathbf{A}(\vec{\theta}_i - \vec{\theta}_j) \right\|_2 = \left\| \mathbf{A}\vec{\theta}_{ij} \right\|_2 \tag{4.7}$$

We learn an appropriate kernel function between two camera poses by minimizing the loss function L defined as follows:

$$L = \sum_i \sum_{j \geq i} (y_{ij} - k_{ij})^2 \tag{4.8}$$

where y_{ij} is the target value or the similarity score between two camera poses, and k_{ij} is the estimate of the target value obtained by combining the Gaussian kernel and Mahalanobis distance. By means of Equation 4.6 we can rewrite the loss function L in Equation 4.8 in terms of the unconstrained matrix $\mathbf{A}$ (instead of $\mathbf{M}$). By minimizing Equation 4.8 we

obtain a metric in an entirely non-parametric way. Finally, our kernel function measures the similarity between two camera poses as:

$$k_{ij} \equiv k(\vec{\theta_i}, \vec{\theta_j}) = \exp\left(-\left\|\mathbf{A}(\vec{\theta_i} - \vec{\theta_j})\right\|_2\right) \tag{4.9}$$

Note that Equation 4.9 indeed describes the similarity between the two camera poses: when the output is equal to 1, the two cameras are identical, and conversely when the output is equal to 0, it means that they are extremely different with no visible 3D points shared between each other. In addition, Equation 4.9 shows how to compute the target estimates k_{ij} involved in the minimization problem described in Equation 4.8.

We define the target values y_{ij} as the mean of the ratios between the intersection of the common 3D points with respect to the number of 3D points visible to each of the two cameras:

$$y_{ij} = \frac{1}{2} \cdot \left| \frac{|X_i \cap X_j|}{|X_i|} + \frac{|X_j \cap X_i|}{|X_j|} \right| \tag{4.10}$$

Indeed, the above equation gives an estimate of the overlap between two views. Figure 4.3 depicts the similarity matrix for all pairs of camera poses in the St. Peter's Basilica dataset we used in our experiments.

Our algorithm learns its own scaling (encapsulated by matrix $\mathbf{A}$) and is therefore invariant to the scaling of the input vectors. This invariance to scaling is very important when dealing with complex functions such as visibility that involves cues with very different scales, such as the translation and the orientation. In our experiments we set the initial value of the matrix $\mathbf{A}$ to the identity matrix, and the algorithm converges in few iterations, satisfying the full-rank matrix $\mathbf{A}$ constraint. The Levenberg-Marquardt (LM) algorithm [Marquardt, 1963] has been used for all non-linear optimization in the metric learning procedure. Next, we will explain how to define the input vector $\vec{\theta_i}$ for each camera pose.

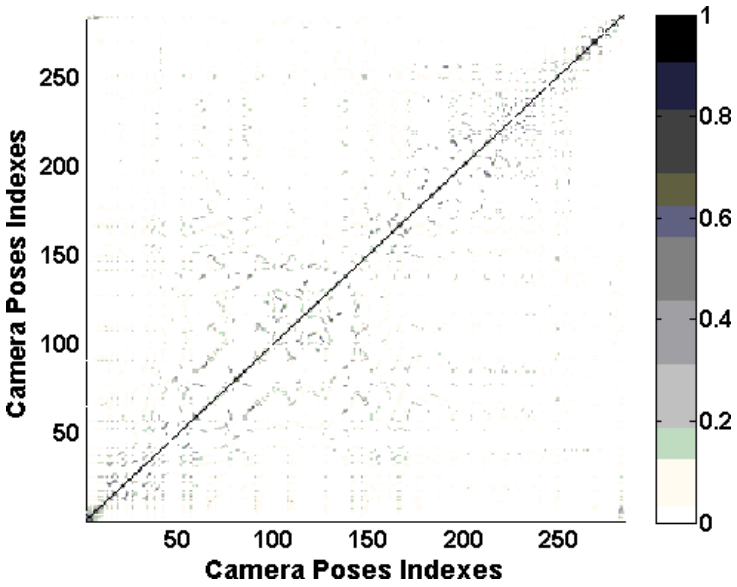


Figure 4.3: Similarity score matrix for the St. Peter's Basilica dataset: Note that most of the camera poses have common visible 3D points with few camera poses. This means, that only the training data nearby a query pose θ_i are informative to correctly predict the visibility of a 3D point.

4.3.1 Euclidean Distance and Viewing Direction Change

For each camera pose we define an input vector $\vec{\theta}_i$ that encodes geometrical information about the pose with respect to the global coordinate frame of the 3D reconstruction. Each camera pose is parametrized by means of a vector $\vec{\theta}_i = \{T_i, R_i\}$ (3D vector for the translation and 4D unit quaternion for the rotation). In particular for our metric learning, we use two cues (difference in camera translation and viewing direction), $\vec{\theta}_{ij} = [d_R \; d_T]^T$, where d_T is the Euclidean distance between the translation of two cameras, and d_R defines the normalized inner product between the viewing directions of the two cameras (θ_i, θ_j). Figure 4.4 depicts an example of the euclidean distance and viewing directions that we use in our kernel formulation.

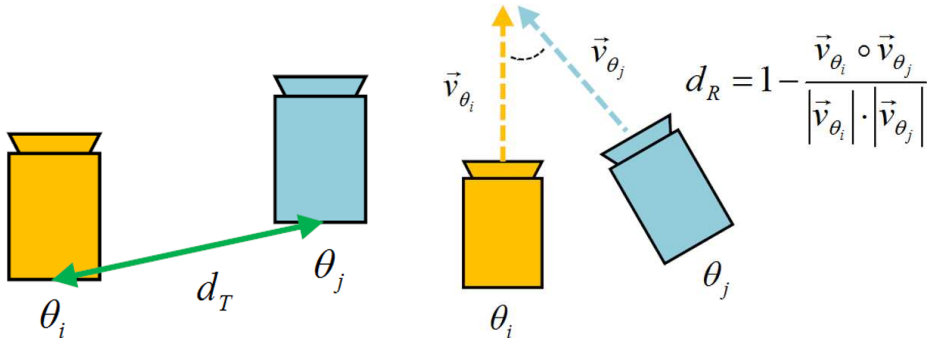


Figure 4.4: Euclidean distance and viewing direction cues.

4.3.2 Image Appearance Cue

Another important cue for a distance metric between two camera poses is their corresponding image appearance. The simplest way to utilize the image is by computing the sum of pixel-wise distances. However, the performance of this approach tends to drop considerably when dealing with wide-baseline images with high spatial variance. In our approach, we model the appearance by local histograms, which are more invariant to small camera motions and have been successfully applied to location recognition problems [Torralba et al., 2003; Ni et al., 2008] recently.

More specifically, the RGB features are accumulated over spatially localized 3×2 image grids. Within each grid, the RGB space is quantized into 50 bins. Hence, each image is represented as a 300-dimensional vector. Adding the image appearance cue, we have the following difference vector for any two cameras: $\vec{\theta}_{ij} = [d_R \; d_T \; d_H]^T$, where d_H is the Euclidian distance between two local histograms. Figure 4.5 depicts an example about how to incorporate the appearance cue in our kernel formulation.

4.4 Speeding Up by Pre-pruning

In this section, we introduce a pre-pruning technique to decrease the complexity of computing the visibility for all the map elements, from the order of the map size $O(M)$ to $O(K)$ where K denotes the number of nearest neighbors of a given camera pose. A naive way of computing the visibility is to iterate over every one and apply Equation 4.3. The complexity of such an approach increases linearly with respect to the size of the map,

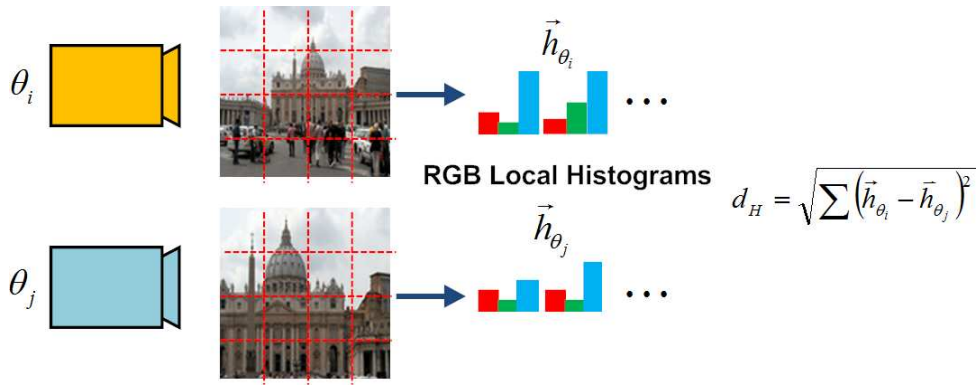


Figure 4.5: Local RGB histograms cue. Best viewed in color.

M. However, the core observation is that the results of the visibility prediction using Equation 4.3 will be mostly zero, since most of the map elements in a large map would not be observed at all by the K Nearest Neighbors (KNNs) of the current query pose θ, effectively making the numerator in Equation 4.3 zero.

As a consequence, once we find the KNNs of the current query pose, we only need to predict the visibility for the subset of map elements seen at least once by these KNNs. Then, we can set the visibility to zero for the rest of the map elements without computing them at all. Finally, the locally weighted K nearest neighbor approximation for the visibility posterior is:

$$P(v_j = 1|\theta) \approx \frac{\sum_{i=1}^{K} k(\theta, \theta_i^{v_j=1})}{\sum_{i=1}^{K} k(\theta, \theta_i)} \tag{4.11}$$

where only the nearest K samples of the query pose $\Theta^K = \{\theta_1 \dots \theta_k\}$ are considered.

4.5 Experimental Results

We evaluate our algorithm using a real 3D database built from 285 photographs of St. Peter's Basilica in the Vatican City, as depicted in Figure 4.2. The SfM process is done as proposed in Snavely's paper [Snavely et al., 2006] and will not be elaborated in this chapter. The dataset comprises of 142, 283 3D points, 466, 222 image measurements, and 285 cameras. This difficult dataset is popular in the computer vision community and has been previously used to evaluate 3D reconstruction approaches such as [Ni et al., 2007].

4.5.1 KNNs Classification

We evaluate our Mahalanobis metric in terms of KNNs classification. For this purpose, we compare the predicted KNNs of our metric to the ground truth neighbors of all the poses from the training dataset. The ground-truth KNNs are obtained by means of the proposed similarity score between two camera poses as shown in Equation 4.10. For a query camera pose from the training set, we compute the similarity score for all the connected poses that share at least one visible feature in common. Then, we obtain the KNNs by sorting the camera poses in a decreasing order of the similarity scores. In this experiment, we used leave-one-out cross-validation, skipping the query camera pose from the training data.

First, we investigate the effectiveness of the cues used in our proposed distance metric. For KNNs classification, we compared the following metrics: Mahalanobis metric with two cues (translation, viewing direction), Mahalanobis metric with three cues (translation, viewing direction, local histograms) and the Euclidean distance of camera translation, viewing directions and image histograms separately. Figure 4.6(a) depicts the average KNNs classification error for all the camera poses of the dataset that see at least one 3D point. A predicted nearest neighbor will be classified as a false positive if that neighbor is not included in the ground truth neighbors set of a given query pose. Figure 4.6(b) depicts a more restrictive ratio, the average KNNs classification recall. This ratio is computed by considering only the first K ground truth nearest neighbors, i.e. even if a predicted neighbor shares some common 3D points with a query pose, we will classify that neighbor as a wrong one if it is not included in the first K ground truth nearest neighbors. For obtaining these graphs, we varied the number of neighbors K to consider in the prediction and compute the ratios according to this K.

We can observe in Figure 4.6(a) that the lowest error ratio is obtained with the proposed Mahalanobis metric with three cues (translation, viewing direction, histograms). The Mahalanobis metric with two cues also exhibits small error ratios similar to the ones obtained with three cues, especially when K is above 10.

The error slightly increases as long as we consider more NN in the prediction and a near constant error is obtained for a large number of neighbors. The reason for this, is that as long as we consider more NN, we are adding neighbors whose similarity score with respect to a query pose is low. For these neighbors, the differences in camera translations and local histograms can be very large, but viewing directions must be very similar. KNNs classification for high values of K is more challenging, and error ratios will slightly increase, since for these cases viewing directions play a more important role than camera translation and local histograms. Hence, for high values of K viewing directions, error ratios are similar to the ones obtained with the Mahalanobis metric with three and two cues. In the experiment in Figure 4.6(b) conclusions are similar. However, since we compute averaged recall ratios considering only the first K ground truth nearest neighbors, here we can observe a gain in adding the appearance cue into the Mahalanobis metric, and also a higher performance of the Mahalanobis metric with three and two cues with respect to viewing directions compared to Figure 4.6(a).

Viewing directions are a much more robust evidence than camera translation distances, due to the fact that people tend to shoot the same interesting object from multiple views which have large baselines between each other. In this case, even if the cameras can move wildly, the viewing direction remains approximately the same. Image histograms are also very useful cues, especially as viewing directions become less reliable as the cameras get close to the interested object. Imagine the scenario in which people pass by a certain facade of St Peter's basilica while taking pictures. Both translations and rotations of the cameras may change dramatically, yet the appearance of the images will be very similar, and so will the histograms of the images.

Figure 4.7 depicts two examples of KNNs classification, showing the different NN rankings that were obtained by means of the proposed Mahalanobis metric with three cues and the ground truth information. That is, a neighbor which is ranked as 1 should be a neighbor that shares the highest number of visible 3D points with respect to the query pose among all the connected poses. Figure 4.7(a) shows classification results considering only $K = 4$ neighbors of one of the images of the St. Peter's Basilica that was taken from St. Angel's Bridge at a distance of approximately 1 km from the basilica.

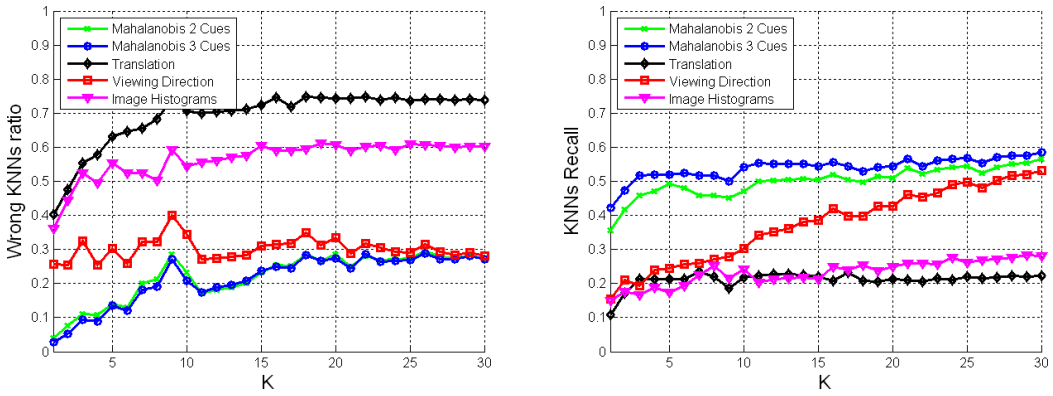


Figure 4.6: Evaluation of Metric Learning for KNNs classification. (a) Depicts the average KNNs classification error for all the camera poses of the dataset that see at least one feature whereas (b) shows the average correct KNNs classification recall only considering the K ground truth nearest neighbors. Best viewed in color.

Figure 4.7(b) depicts a more challenging classification scenario, since the photo was taken in the middle of St. Peter's Square, where appearance details are very similar between slightly different view points due to the similar sculptures surrounding the square. We can observe, that even if rankings are not exactly the same, even in a very difficult scenario such as Figure 4.7(b), all the first predicted neighbors share visible 3D points with respect to the query pose.

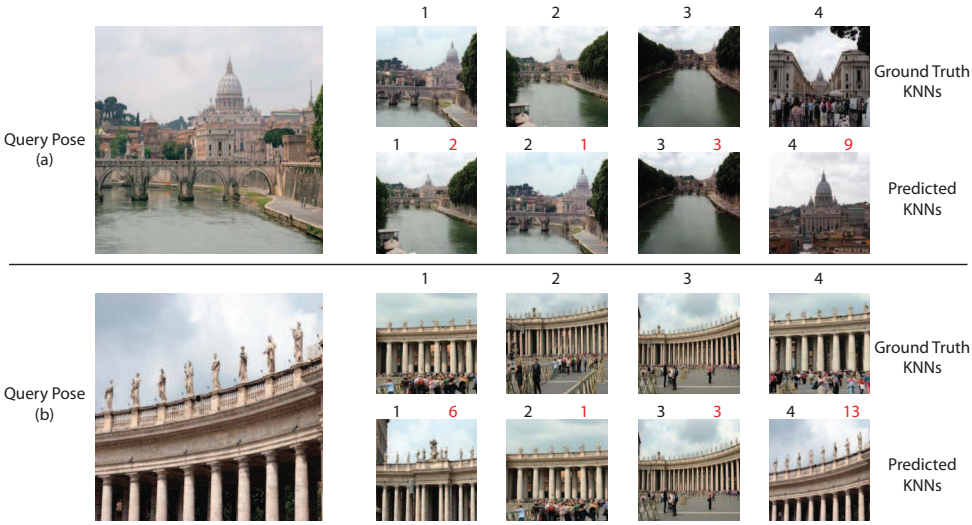


Figure 4.7: KNNs ranking: (a) St. Angel's Bridge (b) St. Peter's Square. We show on the left the KNNs ranking results obtained from ground truth, and at the right, the ranking results obtained with our Mahalanobis metric (three cues). We show in red the ranking results obtained with the proposed metric, whereas for ground truth, we display results in bold. For this experiment, we only considered the first 4 nearest neighbors. Best viewed in color.

4.5.2 Sensitivity to Noise and Number of Nearest Neighbors

Now we evaluate the performance of our visibility prediction with respect to different levels of noise, and study the influence of the number of nearest neighbors K in the resulting prediction. We evaluate the performance of our approach by means of *recall* versus *1-precision* graphs Mikolajczyk and Schmid [2005]. For visibility prediction, we define *recall* and *1-precision* as:

$$recall = \frac{\#predicted\ visible}{\#real\ visible}$$
$$1 - precision = \frac{\#false\ predicted\ visible}{\#all\ predicted\ visible} \tag{4.12}$$

where we know the number of *real visible* 3D points for each of the poses from the dataset. To simulate the noisy signals of a GPS sensor in the city, we added noise with normal distribution to the ground-truth camera pose and then predict the visible 3D points for this noisy pose. We consider random Gaussian noise (mean μ, standard deviation σ) for all the camera translation and viewing direction components and then normalize the noisy viewing direction vector to unit length. In addition, we also added noise to the image histograms (the scale of image histograms is normalized between 0 and 1). Table 4.1 shows the two Gaussian noise settings that we used in our experiments, distinguishing two different levels of noise.

Cue	**Noise Level** $N1$	**Noise Level** $N2$
Translation	$\mu = 15m, \sigma = 30m$	$\mu = 5m, \sigma = 10m$
Orientation	$\mu = 0.1\,, \sigma = 0.2$	$\mu = 0.05, \sigma = 0.1$
Histograms	$\mu = 0.0, \sigma = 0.3$	$\mu = 0.05, \sigma = 0.1$

Table 4.1: Random Gaussian noise settings in our experiments.

According to Equation 4.11 we decide if a feature is visible if its probability of being visible is higher than a fixed threshold. By varying the value of this threshold, we can obtain *recall* versus *1-precision* graphs. In this experiment we include all the camera poses as possible KNNs of the noisy camera pose.

Figure 4.9(a) depicts averaged *recall* versus *1-precision* graphs for all camera poses in the dataset. We added random Gaussian noise ($N1$ and $N2$ as defined in Table 4.1) and considered K in the visibility prediction, comparing Mahalanobis metrics with 2 and 3 cues (denoted as M2 and M3 respectively). The average number of *real visible* 3D points per camera pose in this dataset is 1666.

An important conclusion from Figure 4.9(a) is that a higher *recall* is obtained when considering a small number of nearest neighbors in the prediction. The reason for this is that as long as we increase the number of neighbors, we are also adding more 3D points in the prediction of Equation 4.11. Some of these 3D points may not be truly visible from the query noisy pose, but they may be visible from a local neighborhood of the query noisy pose. Typically, for large baseline 3D reconstructions, one camera view shares most of its visible 3D points with few camera poses, i.e. only the training data nearby a query pose is informative enough to correctly predict the visibility of a feature, as shown in Figure 4.3. In addition, an approximate 20% gain in *recall* can be obtained when the appearance cue is used. This cue becomes less important, however, when the number of neighbors increases. Furthermore, considering that we can have noisy position measurements from a GPS or another hand-held device, the image appearance is a very

useful cue, since in general, it is the less affected by noise. Also, the reduction in *recall* when increasing the number of neighbors is more significant for the Mahalanobis 3 cues case. This is because some neighbors may look similar in terms of appearance but images can be taken from different viewpoints, as for example happens inside St. Peter's Square, where image appearance is similar and repetitive, but viewpoints are different.

4.5.3 Comparison with Heuristic Visibility Prediction

We compare our approach to a *length and angle visibility heuristic* that has been widely used in the literature [Davison et al., 2007]. This heuristic is very easy to compute and provides good results in non-cluttered and small environments. Visibility is calculated considering the difference between the viewpoint from which the 3D point was initially seen and a new viewpoint. This difference in viewpoint has to be below some length and angle ratio to predict the 3D point as visible. Usually a point is expected to be visible if the length ratio $|h_i|/|h_{orig}|$ is close enough to 1 (in practice between 5/7 and 7/5 and the angle difference $\beta = \cos^{-1}((h_i \cdot h_{orig})/(|h_i||h_{orig}|))$ is close to 0 (less than 45° in magnitude). However, the main drawbacks of this criterion is that it can not deal with occlusions since it assumes a *transparent* world and that it has to predict its visibility individually for every point, which can be computationally expensive for very large 3D reconstructions. Figure 4.8 depicts one graphical example of this heuristic visibility prediction.

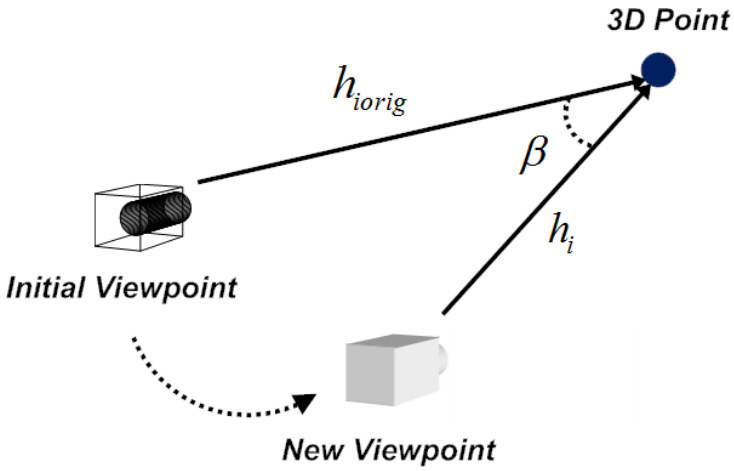


Figure 4.8: Heuristic length and angle visibiliy prediction

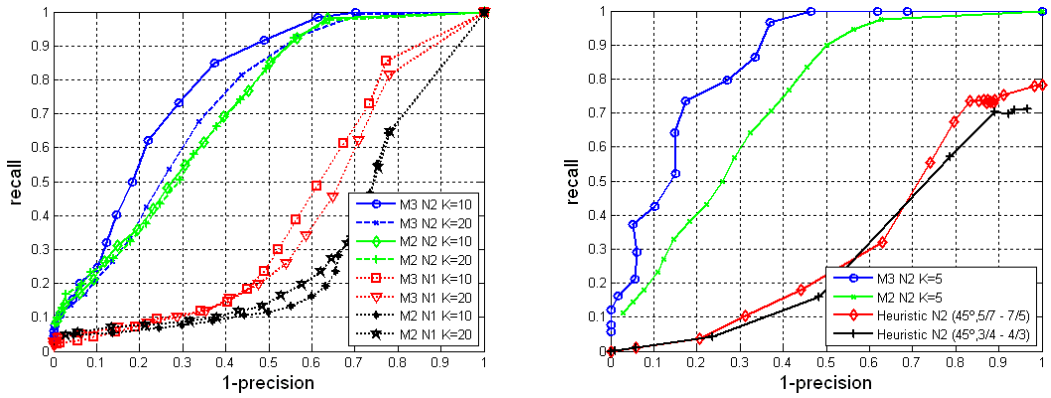
Usually for very large city-scale reconstructions, the number of camera poses is more than a hundred times smaller than the number of 3D points. For example in the reconstructions presented in [Agarwal et al., 2009], in the Dubrovnik reconstruction we have $4,585$ poses and $2,662,981$ 3D points and in the St. Mark's Square we have $13,699$ poses and $4,515,157$ 3D points, which means that we can obtain a substantial speed up if we predict visibility by obtaining the KNNs of a query camera pose, instead of predicting the visibility for each point individually.

We compare our approach with the heuristic visibility prediction described above, but instead of checking the visibility of each 3D point individually, which will incur in a high computational demand and low recall, we introduce some modifications. We first bound the regions for possible camera neighbors, using geo-spatial constraints (3D camera pose location) placing a sphere centered on the query pose. Then we predict the visibility according to the mentioned heuristic, just for those 3D points that are seen by the camera

pose neighbors that lie inside the sphere. Finally, by varying the radius of the sphere, we can generate *recall* versus *1-precision* graphs.

As can be seen in Figure 4.9(b), we obtain results with our visibility prediction which are superior to other common geo-spatial constraints based on translation and angle heuristics. Moreover, we can obtain a very high recall for very high precision values, i.e. we get rid of most of non-visible 3D points yielding a better matching and a faster localization. We obtained the graphs shown in Figure 4.9(b), considering noise level $N2$, and a number of neighbors equal to $K = 5$ for our visibility prediction. For the heuristic prediction, we used the default values: $5/7 < |h_i|/|h_{orig}| < 7/5$ and $\beta < 45°$. We also performed another experiment considering a more restrictive length ratio keeping the same angle ratio: $3/4 < |h_i|/|h_{orig}| < 4/3$. In addition, we can also observe that there is a gain in performance comparing the graphs with $K = 5$ and the graphs in Figure 4.9(a), where K was set to 10 and 20.

A timing evaluation revealed that our MATLAB implementation runs faster than 30 Hz (or 33.3 ms per frame). Just predicting the visible 3D points for a query camera pose takes 12.20 ms for $K = 5$. Visibility heuristic in combination with geo-spatial pruning for a radius of 10 m takes 139.49 ms, whereas predicting the visibility of all the 3D points individually with the heuristic takes 4.44 s. All timing results were obtained on a Core 2 Duo 2.2GHz computer. As shown in 4.9(b), we can see that our algorithm obtained a more robust and faster visibility prediction, yielding better results than conventional approaches.



(a) (b)

Figure 4.9: Recall versus 1-precision graphs: (a) Evaluations under different noise settings and the number of neighbors K (b) Comparisons with respect to the heuristic visibility. The number of neighbors K is set to 5. Best viewed in color.

4.6 Conclusions and Future Work

In this chapter we proposed a novel method for predicting the visibility of 3D points in large-scale urban environments. In our approach, every map point models its visibility with respect to the camera poses via non-parametric distributions. By means of the combination of the Mahalanobis distance and the Gaussian kernel, we showed how to learn a similarity metric between two camera poses in an entirely non-parametric way that is invariant to the scale of the input vectors. We have discussed the importance of the cues used in our metric and shown the benefits of adding local image histograms

instead of using only camera translation and viewing direction.

For very large 3D reconstructions, the number of map elements is more than a hundred times higher than the number of camera poses. Based on this observation, we think that for fast and robust vision-based localization over large 3D reconstructions, our algorithm can dramatically reduce the computation time and improve robustness, contrary to other common approaches that check the visibility of each of the 3D points from the dataset individually or pruning-out information based in geo-spatial constraints.

As future work, we are interested in applying our idea to incremental SfM reconstruction, by incrementally learning the Mahalanobis distance metric between camera poses, and then showing the results of our method in vision-based localization experiments under very large city-scale environments. Moreover, the addition of new cues to the proposed metric will be studied.

Chapter 5

Visual SLAM and Vision-Based Localization for Humanoid Robots

In this chapter, we consider the problem of real-time localization for humanoid robots using a single camera as the only sensor. In order to obtain fully autonomous robots an accurate localization of the robot in the world is much more than desirable. Furthermore, if we can obtain an accurate localization in real-time, we can use the remaining computational resources to perform other important humanoid robotic tasks such as planning [Perrin et al., 2010], 3D object modeling [Foissote et al., 2010] or visual perception [Bohg et al., 2009].

Indeed, many humanoid robotics applications will be benefited from an accurate and fast localization of the robot. For a robust localization, we can choose between different alternatives: One option is to estimate simultaneously the localization of the robot and the map of the environment, yielding the well-known Simultaneous Localization and Mapping (SLAM) problem in the robotics community [Durrant-White and Bailey, 2006]. However another possible option is to dedicate more computational resources in the reconstruction of a persistent map, and then use this map for long-term localization or navigation purposes. In this way, we can take advantage of the prior map of the robot's environment learning different parameters ranging from *visibility prediction* [Alcantarilla et al., 2011], 3D object reconstruction [Stasse et al., 2007] to scene understanding [Li et al., 2009].

However, accurate vision based localization for humanoid robots is still a challenging problem due to several aspects such as: noisy odometry, innacurate 3D data, complex motions and motion blur originated due to fast robot motion and to the large jerk caused by the landing impact of the feet. In addition, humanoids usually cannot be assumed to move on a plane to which their sensors are parallel due to their walking motion [Hornung et al., 2010]. Therefore, compared to wheeled robots, there is still open research for robust and accurate localization for humanoid robots.

In the particular case of humanoid robots, it is very important that the sensors are light-weight and small. Humanoids should be stable under all possible motions, and heavy sensors can compromise this stability. Furthermore, not all sensors are suitable for humanoid robots. For example not all laser scanners can be mounted on humanoid platforms, especially the heavy ones such as for example the SICK LMS-221. Only small laser range sensors (e.g. Hokuyo URG-04LX) are suitable for humanoid robotics applications [Kagami et al., 2005]. However, the main problem of these small laser range sensors is the limited distance range (up to 4 m for the Hokuyo URG-04LX). All these reasons make cameras an appealing sensor for humanoid robots: they are light-weight and cheaper than laser scanners and stereo cameras can also provide higher distance ranges (depending

on the the stereo rig baseline). Moreover, most of advanced commercial humanoids platforms are already equipped with vision systems. However, there have been only limited attempts at vision-based localization for humanoid robots.

In this chapter, we show that is possible to obtain a real-time robust localization of a humanoid robot, with an accuracy of the order of cm just using a single camera and a single CPU. Prior to localization we compute an accurate 3D map of the environment by means of the stereo visual SLAM algorithm described in Chapter 2. For building an accurate 3D map we use stereo vision mainly for two reasons: we can measure in a direct way the scale of each detected point and we can obtain dense depth information, which is a well-studied problem for stereo vision [Scharstein and Szeliski, 2002]. In this way we can solve the main drawback of monocular SLAM approaches, i.e. recovering the scale of a map due to observability problems in recovering 3D information from 2D projections. However, once we have obtained a 3D map of the environment, we can perform monocular vision-based localization using the 3D map as a prior. Hence, for localization experiments we can avoid the dense depth map generation process, which in certain occassions can be a high time consuming operation, and perform robust and efficient real-time localization just using a single camera.

To satisfy all these demands, we firstly build a 3D map of the environment using stereo visual SLAM techniques based on Bundle Adjustment (BA). BA is a non-linear least squares optimization based on Gauss-Newton methods, such as for example the well known Levenberg-Marquardt (LM) [Marquardt, 1963]. Inspired by recent works in non-linear SLAM, we propose to use a stereo visual SLAM algorithm combining local BA and global BA to obtain accurate 3D maps with respect to a global coordinate frame. Then, these maps can be used later for monocular vision based localization or navigation. In this way, 3D points and camera poses are refined simultaneously through the sequence by means of local BA, and when a loop closure is detected, the residual error in the reconstruction can be corrected by means of global BA adding the loop closure constraints.

Once we have a 3D map of the environment, we would like to use this map for different robotics applications such as localization, planning or navigation. Vision-based localization in a large map of 3D points is a challenging problem. One of the most computationally expensive steps in vision-based localization is the data association between a large map of 3D points and 2D features perceived by the camera. Then, matching candidates are usually validated by geometric constraints using a RANdom SAmple Consensus (RANSAC) framework [Bolles and Fischler, 1981]. Therefore, we need a smart strategy to sample the large database of 3D points and perform an efficient data association between the 3D map points and perceived 2D features by the camera. Given a prior map of 3D points and perceived 2D features in the image, our problem to solve is the estimation of the camera pose (with known intrinsic parameters) with respect to a world coordinate frame. Basically, this problem is known in the literature as the Perspective-n-Point (PnP) problem [Lu et al., 2000; Ansar and Danilidis, 2003; Schweighofer and Pinz, 2008].

For solving efficiently the PnP problem, we propose to use the *visibility prediction* algorithm described in [Alcantarilla et al., 2011]. Visibility prediction exploits all the geometric relationships between camera poses and 3D map points in the prior 3D reconstruction. Then, during vision-based localization experiments we can speed-up tremendously the data association and robot localization by predicting only the most highly visible 3D points given a prior on the camera pose. In this way, we can solve the PnP problem in an efficient and fast way, reducing considerably the number of outliers in the set of 3D-2D matching correspondences.

In [Alcantarilla et al., 2010b], the visibility prediction idea was successfully used for estimating the pose of a hand-held camera in cluttered office-like environments. Results were quite satisfactory taking into account that no appearance descriptor was considered in the matching process between the 3D map points reprojections and perceived 2D features. In this thesis, we use the visibility prediction algorithm in the context of humanoid robot localization, but adding more capabilities due to the use of appearance information. In our map, each 3D point is also described by a low-dimensional descriptor vector that encodes the appearance information of a local area centered on the point of interest. By means of appearance information, we can easily perform fast robot re-localization for those cases where the robot gets lost or is kidnapped.

The chapter is organized as follows: In Section 5.1 we review the different approaches regarding humanoid robots localization and their main limitations. The main characteristics of the HRP-2 humanoid robot are described in Section 5.2. In Section 5.3, we explain the main steps of our monocular vision-based localization algorithm using visibility prediction. In Section 5.4 we show extensive localization experiments with the HRP-2 robot. Finally, main conclusions and future work are described in Section 5.5.

5.1 Related Work

Most humanoid robotic platforms have vision systems. Cameras seem to be an appealing sensor for humanoid robotics applications: they are small, cheap and light-weight compared to other sensors such as laser scanners. However, there have been only limited attempts at vision-based localization, whereas more interesting results have been obtained using laser scanners as the main sensor. Now, we will briefly review the main works in laser-based localization and will put more emphasis on the main limitations of vision-based approaches.

5.1.1 Laser-Based Localization

Stachniss et al. [2008] proposed a method for learning accurate 2D maps for a humanoid robot equipped with a laser scanner. Then, these maps can be used efficiently in applications that require global localization such as the museum robot guide *Robotinho* presented in [Faber et al., 2009]. Figure 5.1 depicts one example of a learned grid map by means of the approach described in [Stachniss et al., 2008] acquired with the humanoid robot *Robotinho*. Recently, Hornung et al. [2010] proposed a 6-Degrees of Freedom (DoF) humanoid robot laser-based localization approach for complex environments. In the mentioned work, only localization is addressed and therefore a volumetric 3D map of the environment is created beforehand.

5.1.2 Vision-Based Localization

Ozawa et al. [2007] proposed to use stereo visual odometry to create local 3D maps for online footstep planning. They validate their algorithm, performing several experiments with biped robots walking through an obstacle-filled room, while avoiding obstacles. The main drawback of this approach is the drift created by the accumulation of errors that typically occur in visual odometry systems [Nistér et al., 2004; Kaess et al., 2009]. In addition, this approach lacks the ability to close loops and the local nature of the obtained 3D maps prevents the final maps from life-long mapping. Within the visual odometry context, Pretto et al. [2009] proposed a visual odometry framework robust to motion

Figure 5.1: A learned grid map using noisy and short-range laser data acquired with the humanoid *Robotinho*.

blur. Motion blur is one of the most severe problems in grabbed images by humanoid robots, specially for the smaller ones, making vision-based applications more challenging.

In [Michel et al., 2007], the authors proposed a real-time 3D tracking for humanoid robot locomotion and stair climbing. By tracking the model of a known object they were able to recover the robot's pose and to localize the robot with respect to the object. Real-time performance was achieved by means of Graphic Processing Units (GPUs). The main limitations of this approach are that it is extremely dependent on the 3D object to track and that the 3D model is relatively small. However, it can be useful for challenging humanoid robots scenarios such as stairs climbing.

Kwak et al. [2009] presented a 3D grid and particle based SLAM for humanoid robots using stereo vision. The depth data from the stereo images was obtained by capturing the depth information of the stereo images at static positions in the environment, measuring the distance between these positions manually. This tedious initialization and the computational burden introduced by the grid matching process, prevents the system from mapping more complex environments and from real-time performance, which is of special interest in humanoid robots applications. Davison et al. [2007] showed succesful monocu-

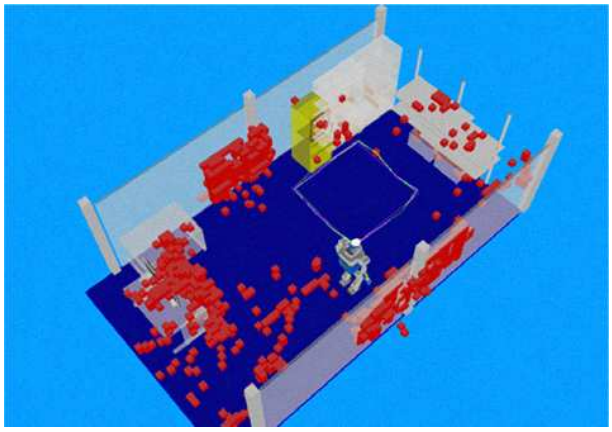


Figure 5.2: The resulting 3D grid map obtained by means of paticle based SLAM proposed in [Kwak et al., 2009].

lar SLAM results for small indoor environments using the HRP-2 robot. This approach, known as MonoSLAM, is a monocular Extended Kalman Filter (EKF) vision-based system, that allows to build a small map of sparse 3D points. This persistent map permits almost drift-free real-time localization over a small area. However, only accurate results

were obtained when the pattern generator, the robot odometry and inertial sensing are fused to aid the visual mapping into the EKF framework as it was shown in [Stasse et al., 2006]. The fusion of the information from different sensors can reduce considerably the uncertainty in the camera pose and the 3D map points involved in the EKF process, yielding better localization and mapping results. Although in most of occasions, odometry in humanoid robots can be estimated only very roughly [Hornung et al., 2010].

The main drawback of EKF-based approaches is the limited number of 3D points that can be tracked, apart from divergence from the true solution due to linearization errors. As it has been shown in several works [Dellaert and Kaess, 2006; Strasdat et al., 2010a] non-linear optimization techniques such as BA or Smoothing and Mapping (SAM) are superior in terms of accuracy to filtering based methods, and allow to track many hundreds of features between frames. BA is a very popular and well-known technique used in computer vision, and in particular for Structure from Motion (SfM) problems. A complete survey on BA methods can be found in [Triggs et al., 1999]. These kind of methods have been successfully employed in different problems such as augmented reality [Klein and Murray, 2007] or large-scale mapping for mobile robot platforms [Konolige and Agrawal, 2008; Mei et al., 2010].

One of the most successful monocular SLAM approaches is the Parallel Tracking and Mapping (PTAM) approach [Klein and Murray, 2007]. PTAM was originally developed for augmented reality purposes in small workspaces and combines the tracking of many hundred of features between consecutive frames for accurate camera pose estimates and non-linear map optimization. The map optimization uses a subset of all camera frames of special importance in the reconstruction (keyframes) to build a 3D map of the environment. Recently, [Blösch et al., 2010] showed a vision-based navigation approach for Micro Aerial Vehicles (MAVs) that uses PTAM for accurate pose estimates. The main limitations of PTAM are that it does not scale well with larger environments and that it is necessary to simulate a virtual stereo pair to initiliaze the algorithm. This initialization is carried out in order to estimate an approximate depth of the initial 3D points. Then, new 3D points will be triangulated according to previous reconstructed keyframes. This initialization procedure plays a very important role in the final quality of the 3D map and results can differ substantially from real ones if this stereo initialization is not accurate enough as for example it was shown in [Wendel et al., 2011]. Therefore, in order to avoid these problems we propose to use our own stereo visual SLAM algorithm to build an accurate 3D map of the scene and then perform efficient and fast monocular vision-based localization with visibility predicition. In Section 5.4.3 we compare our monocular vision-based localization algorithm to the PTAM approach under one sequence performed by the HRP-2 robot.

5.2 The HRP-2 Humanoid Robot

The HRP-2 humanoid platform is equipped with a high-performance forward-looking trinocular camera rig and a wide angle camera. The wide-angle camera is normally used for grasping or interaction tasks, providing the capability to make accurate 3D measurements of objects located very close to the camera. In this work, we only consider the two cameras that are attached to the ears of the robot. These two cameras have a baseline of approximately 14.4 cm and an horizontal field of view of 90° for each of the cameras.

For the stereo visual SLAM algorithm, we use both left and right cameras, but considering the left camera as the reference one in the 3D reconstruction process. Then, during

monocular vision-based localization experiments we just consider only the left camera for the pose estimation problem. This is possible, since we use a prior 3D map of the scene and therefore we can perform vision-based localization with a single camera. Figure 5.3 depicts an image of the HRP-2 stereo rig settings. The height of HRP-2 is 155 cm in standing up position and the total weight is about 58 kg. More detailed specifications of this humanoid platform can be found in the work by Kaneko et al. [2004].

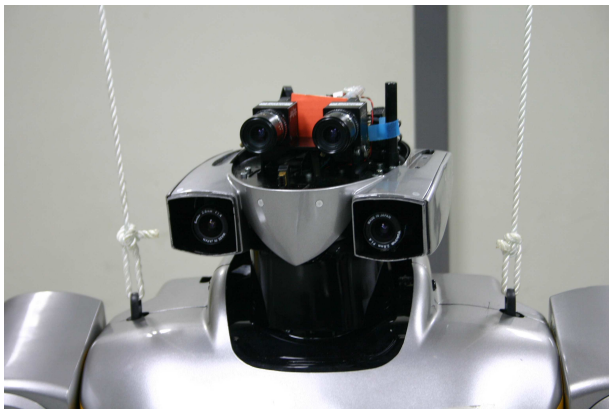


Figure 5.3: HRP-2 stereo rig settings. In this work we consider the two cameras attached to the ears that have a baseline of approximately 14.4 cm.

5.3 Monocular Vision-Based Localization

Once we have obtained a 3D map of the environment (by using the stereo visual SLAM algorithm described in Chapter 2), we are interested in using that map for common humanoid robot tasks such as navigation or planning, while providing at the same time an accurate localization of the robot with respect to a global coordinate frame. For this purpose, obtaining a real-time and robust vision-based localization is mandatory. Given a prior map of 3D points and perceived 2D features in the image, our problem to solve is the estimation of the camera pose with respect to the world coordinate frame. Basically, the problem we have to solve now is known as the Perspective-n-Point (PnP) problem.

The PnP problem, i.e. estimating the pose of a calibrated camera based on 2D measurements and known 3D scene, is a thoroughly studied problem in computer vision [Lu et al., 2000; Ansar and Danilidis, 2003]. In general, even with a perfect set of known 3D-2D correspondences, this is a challenging problem. Although there exist some globally optimal solutions [Schweighofer and Pinz, 2008] that employ Second Order Cone Programs (SOCP), the main drawback of the current globally optimal solutions to the PnP problem is the computational burden of these methods. This makes difficult to integrate these algorithms for real-time applications such as the ones we are interested with humanoid robots.

Our main contribution for solving the PnP problem efficiently, is using the output of the *visibility prediction* algorithm (given a prior on the camera pose) to predict only the most highly visible 3D points, reducing considerably the number of outliers in the set of correspondences. In this way, we can make the data association between 3D map points and 2D features easier, thus speeding up the pose estimation problem. Figure 5.4 depicts an overall overview of our vision-based localization approach with visibility prediction. To clarify, the overall vision-based localization algorithm works through the following steps:

1. While the robot is moving, the camera acquires a new image from which a set of image features $Z_t = \{z_{t,1} \dots z_{t,n}\}$ are detected by a feature detector of choice. Then, a feature descriptor is computed for each of the detected features. Notice, that even any kind of feature detector and descriptor may be used, it is necessary that both detector and descriptor are the same and have the same settings as in the map computation process described in Chapter 2.

2. Then, by using the visibility predicition algorithm, a promising subset of highly visible 3D map points is chosen and re-projected onto the image plane based on the estimated previous camera pose θ_{t-1} and known camera parameters.

3. Afterwards, a set of putative matches C_t are formed where the i-th putative match $C_{t,i}$ is a pair $\{z_{t,k}, x_j\}$ which comprises of a detected feature z_k and a map element x_j. A putative match is created when the Euclidean distance between the appearance descriptors of a detected feature and a re-projected map element is lower than a certain threshold.

4. Finally, we solve the pose estimation problem minimizing the following cost error function, given the set of putative matches C_t:

$$\arg\min_{R,\mathbf{t}} \sum_{i=1}^{m} \|z_i - K\left(R \cdot x_i + \mathbf{t}\right)\|_2 \tag{5.1}$$

where $z_i = (u_L, v_L)$ is the 2D image location of a feature in the left camera, x_i represents the coordinates of a 3D point in the global coordinate frame, K is the left camera calibration matrix, and R and t are respectively the rotation and the translation of the left camera with respect to the global coordinate frame. The PnP problem is formulated as a non-linear least squares procedure using the LM algorithm implementation described in [Lourakis, 2004]. The set of putative matches may contain outliers, therefore RANSAC is used in order to obtain a robust model free of outliers.

5.3.1 Initialization and Re-Localization

During the initialization, the robot can be located in any particular area of the map. Therefore, we need to find a prior camera pose to initialize the vision-based localization algorithm. For this purpose, we compute the appearance descriptors of the detected 2D features in the new image and match this set of descriptors against the set of descriptors from the list of stored keyframes from the prior 3D reconstruction. In the matching process between the two frames, we perform a RANSAC procedure forcing epipolar geometry constraints. We recover the camera pose from the stored keyframe that obtains the highest inliers ratio score. If this inliers ratio is lower than a certain threshold, we do not initialize the localization algorithm until the robot moves into a known area yielding a high inliers ratio. At this point, we are confident about the camera pose prior and initialize the localization process with the camera pose parameters of the stored keyframe with the highest score.

Eventually, it may happen that the robot gets lost due to bad localization estimates or that the new camera pose is rejected due to a small number of inliers in the PnP problem. In those cases, we perform a fast re-localization by checking the set of appearance descriptors of the robot's new image against only the stored set of descriptors of the keyframes that are located in a certain distance area of confidence around the last accepted camera pose estimate.

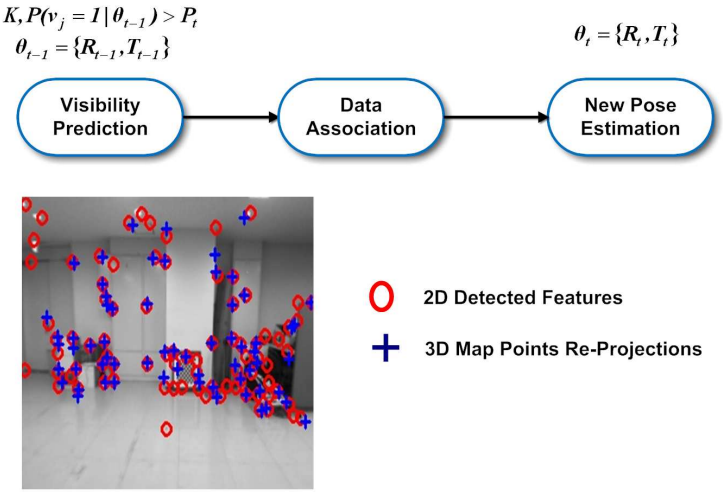


Figure 5.4: The input for the visibility prediction algorithm is the latest camera pose θ_{t-1}, the number of KNNs (K) and a probability threshold P_t. Only the highly visible 3D map points are re-projected onto the image plane of the left camera, and a set of putative matches between 2D detected features and map elements is formed. Then, the PnP problem is solved yielding the localization of the robot with respect to a world coordinate frame θ_t at time t. Best viewed in color.

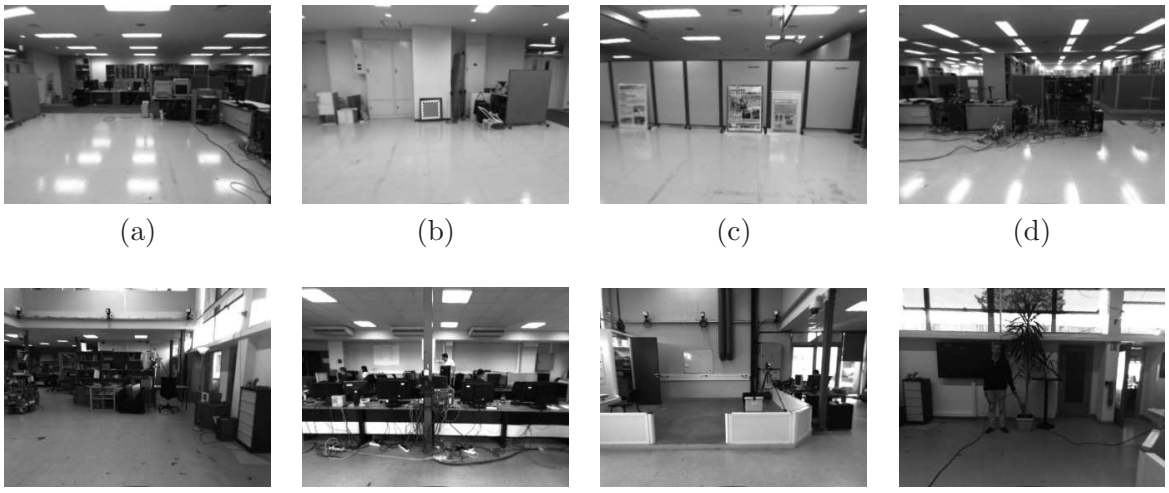
5.4 Results and Discussion

In this section, we show several localization experiments conducted on the HRP-2 humanoid robot. We created two different datasets of common humanoid robotics laboratory environments. The first dataset is called *Tsukuba*, and it was done at the Joint Robotics Laboratory, CNRS-AIST, Tsukuba, Japan. This dataset comprises of different sequences for the evaluation of the monocular vision-based localization algorithm under the assumption that a prior 3D map is known. In particular, in this dataset we have different robot trajectories (square, straight) and challenging situations for the localization such as robot kidnapping, people moving in front of the robot and changes in lighting conditions. For this dataset, we performed experiments with an image resolution of 320×240 and a frame rate of 15 frames per second. The main motivation of using that image resolution is that in this dataset we focused more on achieving real-time localization results while at the same time obtaining robust pose estimates.

The second dataset called *Toulouse* was done at the Gepetto Robotics and Artificial Intelligence Laboratory, LAAS/CNRS, Toulouse, France. For this dataset, we performed experiments with an image resolution of 640×480 and a frame rate of 15 frames per second. By using a higher resolution, computation times will be higher than for the Tsukuba dataset, however we can expect some improvements in localization accuracy and quality of the 3D reconstruction. In addition, in this dataset we chose that resolution to perform a fair comparison against PTAM. Originally, PTAM stores a three level scale-space pyramid representation of each frame, being the level zero an image resolution of 640×480, and the coarsest level 80×60 pixels. In this dataset we have different robot trajectories (square, straight, circular) and also difficult scenarios such as people moving in front of the robot and some changes in the environment.

Figure 5.5 depicts some of the extracted keyframes for two different sequences from the Tsukuba and Toulouse datasets respectively. It can be observed that in some areas of

the two different environments, there is a lack of texture due to the presence of walls (Figure 5.5(c)), fluorescent and natural lighting (Figure 5.5(d,h)) and foliage (Figure 5.5(h)).



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

(e) (f) (g) (h)

Figure 5.5: Some keyframes of the reconstructed environments. The environments are typical from humanoid robotics laboratories. (a)-(d) Four extracted keyframes from one reconstruction of the Tsukuba dataset. (e)-(h) Four extracted keyframes from a sequence of the Toulouse dataset.

In order to evaluate the accuracy of our vision-based localization algorithms, we compare our localization results against ground truth measurements for some of the sequences of the Toulouse dataset. We obtained ground truth information by using a Vicon motion capture system [1]. The Vicon motion capture system is a state-of-the-art infrared marker-tracking system that offers millimeter resolution of 3D spatial displacements.

We used the pattern generator described in [Stasse et al., 2008] to perform a set of pre-computed sequences of interest. Due to noisy odometry, there exists a discrepancy between the desired trajectory and the real one. This is the reason why in the sequences the robot is not able to fully close the loop in some of the planned trajectories.

Firstly, we show the accuracy of our stereo visual SLAM algorithm in Section 5.4.1. We stand out the accuracy of our approach by comparing our trajectory estimates with respect to the ground truth obtained by the motion capture system. Then, we show the monocular vision-based localization results with visibility prediction in Section 5.4.2 and Section 5.4.3, both for the Tsukuba and Toulouse datasets respectively. Finally in Section 5.4.4 we show a timing evaluation for the two different datasets.

5.4.1 Stereo Visual SLAM Accuracy

For a robust localization of the robot, we compute an accurate 3D map of the environment by means of the stereo visual SLAM algorithm described in Chapter 2. In addition, since the visibility prediction algorithm described in Chapter 4 depends on the number of camera poses that are present in the prior 3D reconstruction, this reconstruction should comprise of enough camera viewpoints and map 3D points to perform an efficient long-term localization.

[1] For more information, please check the following url: http://www.vicon.com/

Figure 5.6 depicts a comparison of the obtained trajectory for a circular 3 m diameter sequence by our visual stereo SLAM algorithm and the ground truth collected by means of a Vicon motion capture system. We can observe that the estimated trajectory is very approximate to the motion capture data. It can also be observed that in some parts of the sequence the motion capture system missed to compute reliable pose estimates, mainly because the retro-reflective marker attached to the robot's waist was partially occluded.

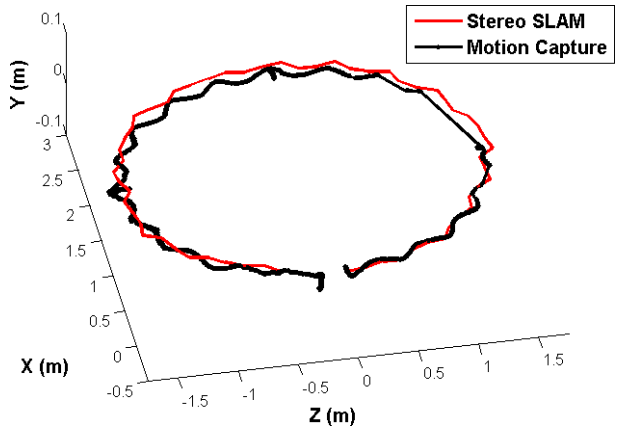


Figure 5.6: Camera trajectories for the circle 3 m diameter sequence from the Toulouse dataset. The estimated trajectory of our stereo visual SLAM algorithm is depicted in red, and the ground truth trajectory obtained by the motion capture system is depicted in black. Best viewed in color.

Due to the mentioned discrepancy between the desired trajectory and the real one performed by the robot, the robot was not able to close the loop in this sequence. Therefore, there is a drift between the initial and end position of the robot in the sequence. Figure 5.7 depicts the final 3D map and keyframes obtained with our stereo visual SLAM system. One can clearly appreciate a circular trajectory of 3 m diameter.

Table 5.1 shows the information about the latest robot's pose in the sequence both for the stereo visual SLAM and motion capture system. According to those results we can observe that the absolute error at the end of the sequence was about 2 cm in the Y axis and 10.80 cm and 18.80 cm for the X and Z axes respectively. The error increases at the end of the sequence, mainly because in the last part of the sequence the robot was facing a challenging low-textured environment. Figure 5.21(b) depicts one keyframe extracted from this area.

Camera Pose Element	**Final Position Stereo SLAM**	**Final Position MOCAP**	**Error** $\lvert\epsilon\rvert$ (m)
X (m)	-0.1322	-0.0242	0.1080
Y (m)	0.0018	0.0297	0.0279
Z (m)	-0.5142	-0.3262	0.1880

Table 5.1: Comparison of stereo Visual SLAM and motion capture (MOCAP) camera trajectories for a circular 3 m diameter sequence from the Toulouse dataset.

Figure 5.8 depicts another comparison of our stereo visual SLAM against motion capture data. In this case, the robot performed a 3 m straight line sequence. For this sequence we had always good visibility conditions between the retro-reflective marker attached to

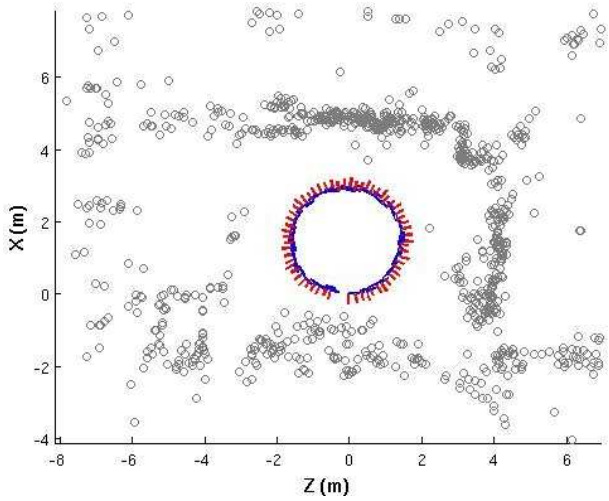


Figure 5.7: Stereo Visual SLAM results: Final 3D map and set of reconstructed keyframes for the circle 3 m diameter sequence from the Toulouse dataset.

the robot and the motion capture camera. We can observe again that both trajectories are very similar. Table 5.2 shows the information of the latest robot's pose in the sequence both for the stereo visual SLAM and motion capture system. This time, we can observe that the trajectory estimates for our vision-based method are pretty accurate about the order of few cm. The estimated trajectory length of our method for this sequence is 3.0934 m and for the motion capture system the estimated length is 3.0833 m.

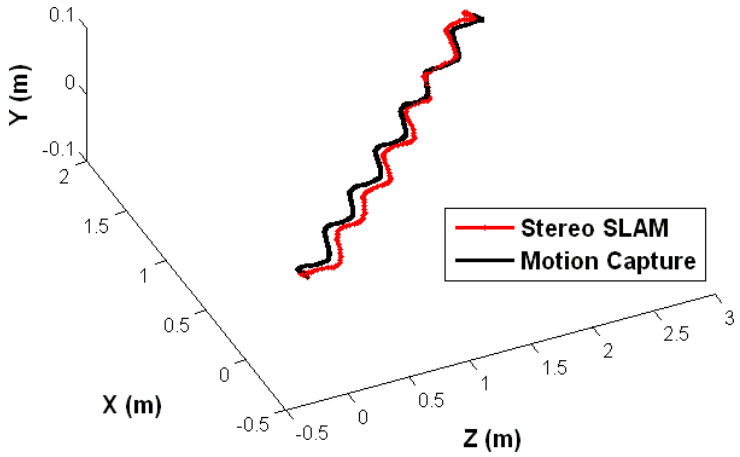


Figure 5.8: Camera trajectories for the straight 3 m sequence from the Toulouse dataset. The estimated trajectory of our stereo visual SLAM algorithm is depicted in red, and the ground truth trajectory obtained by the motion capture system is depicted in black. Best viewed in color.

5.4.2 Localization Results: Tsukuba Dataset

In this section we evaluate the accuracy and robustness of our monocular vision-based localization algorithm with visibility prediction under different robot trajectories and scenarios. In the Tsukuba dataset, the experiments were performed considering an image resolution of 320×240 and a frame rate of 15 frames per second. For the visibility predic-

Camera Pose Element	Final Position Stereo SLAM	Final Position MOCAP	Error $\|\epsilon\|$ (m)
X (m)	-1.8357	-1.7780	0.0577
Y (m)	0.0000	-0.0033	0.0033
Z (m)	2.5260	2.5190	0.0070

Table 5.2: Comparison of stereo Visual SLAM and motion capture (MOCAP) camera trajectories for a straight 3 m length sequence from the Toulouse dataset.

tion algorithm we consider the following input parameters of the algorithm: $K = 10$ and $P_t > 0.20$. We chose a threshold value of 2 pixels in the RANSAC process, for determining when a putative match is predicted as an inlier or outlier in the PnP problem.

Square 2 m Size Sequence

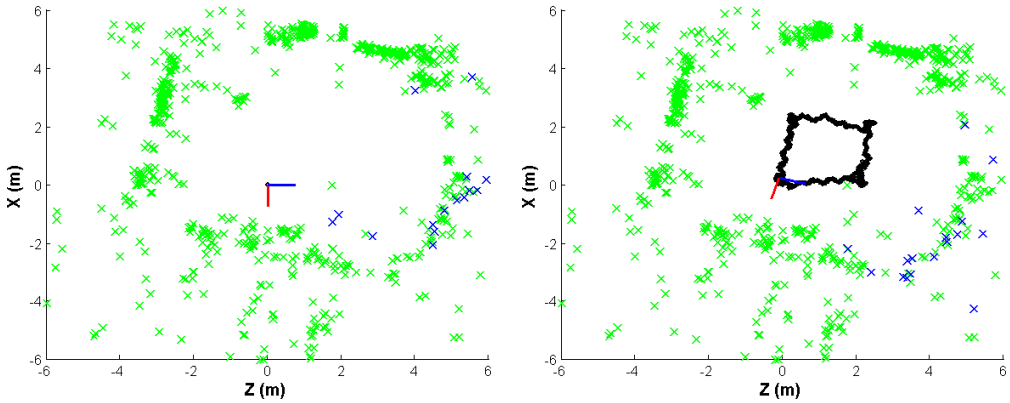
In this sequence, the robot performed a 2 m size square in a typical humanoid robotics laboratory. This sequence was designed for capturing different camera viewpoints both in translation and orientation. Firstly, we built a 3D map of the environment by using the stereo visual SLAM algorithm described in Chapter 2 in a similar sequence and performed visibility learning. The resulting 3D map comprises of 935 points and 75 keyframes.

At the start of the sequence, we placed the robot at the origin of the map, and then by using the pattern generator, the robot performed a square of 2 m size. We measured manually the final position of the robot, and this position was $(X = 0.14, Y = 0.00, Z = -0.02)$ in meters. Due to the existing drift between the planned trajectory and the real one, the robot was not able to close the loop itself. Then, we validate our vision-based localization algorithm with visibility prediction under a similar square sequence.

Figure 5.9 depicts the initial and final position of the robot, and the performed trajectory. Table 5.3 shows the obtained localization results using visibility prediction for this square sequence. According to the results we can see that the localization accuracy is very good, about the order of cm. The differences with respect to the real trajectory for the final position are very small 9 cm, in the X coordinate and about 7 cm in the Z coordinate. While the robot was walking the pattern generator fixed the Y coordinate always to the same value. Therefore, in the PnP problem we add this constraint to speed up the process, although our algorithm can deal with 6DoF.

Camera Pose Element	Start Position	Final Position
X (m)	0.0000	0.2320
Y (m)	0.0000	0.0000
Z (m)	0.0000	-0.0092
q_0	1.0000	0.9905
q_x	0.0000	0.0034
q_y	0.0000	0.1375
q_z	0.0000	0.0050

Table 5.3: Square 2 m size monocular vision-based localization results (Tsukuba dataset).



(a) (b)

Figure 5.9: Square 2 m size localization results Tsukuba dataset. (a) and (b) depict the initial and final position of the robot and the performed trajectory in the sequence. In these two images the robot trajectory is depicted in black, the visible 3D points are depicted in blue and the rest of 3D points in green. Best viewed in color.

Straight Line 3 m Length Sequence

Now, in this experiment we validate our vision-based localization algorithm under new camera viewpoints that were not captured during the map computation process. The visibility prediction algorithm depends on the number and locations of the keyframes in the prior 3D reconstruction. Therefore, the PnP problem should be more difficult to solve in those areas where we have a small density of keyframes. Data association is also more challenging as well, due to the fact that the appearance of new perceived 2D features may not be captured properly by the stored descriptors of the map elements. For this purpose, we planned a sequence in which the robot started in a known position in the 3D map and moved in a straight line of 3 m length. Since in the prior 3D map we have only keyframes in a square 2 m×2 m area, in this experiment we have 1 m length without keyframes. In this new area we should expect from the visibility prediction algorithm lower visibility probabilities for the predicted 3D points than in a well-mapped area where we can have a higher number of keyframes. Figure 5.10 depicts the initial and final position of the robot in the sequence, and their associated image views with detected 2D features and 3D map reprojections.

Camera Pose Element	Start Position	Final Position
X (m)	0.1191	0.5644
Y (m)	0.0000	0.0000
Z (m)	0.0045	3.1633
q_0	1.0000	0.9994
q_x	0.0000	0.0196
q_y	0.0000	-0.0293
q_z	0.0000	0.0038

Table 5.4: Straight line 3 m length monocular vision-based localization results (Tsukuba dataset).

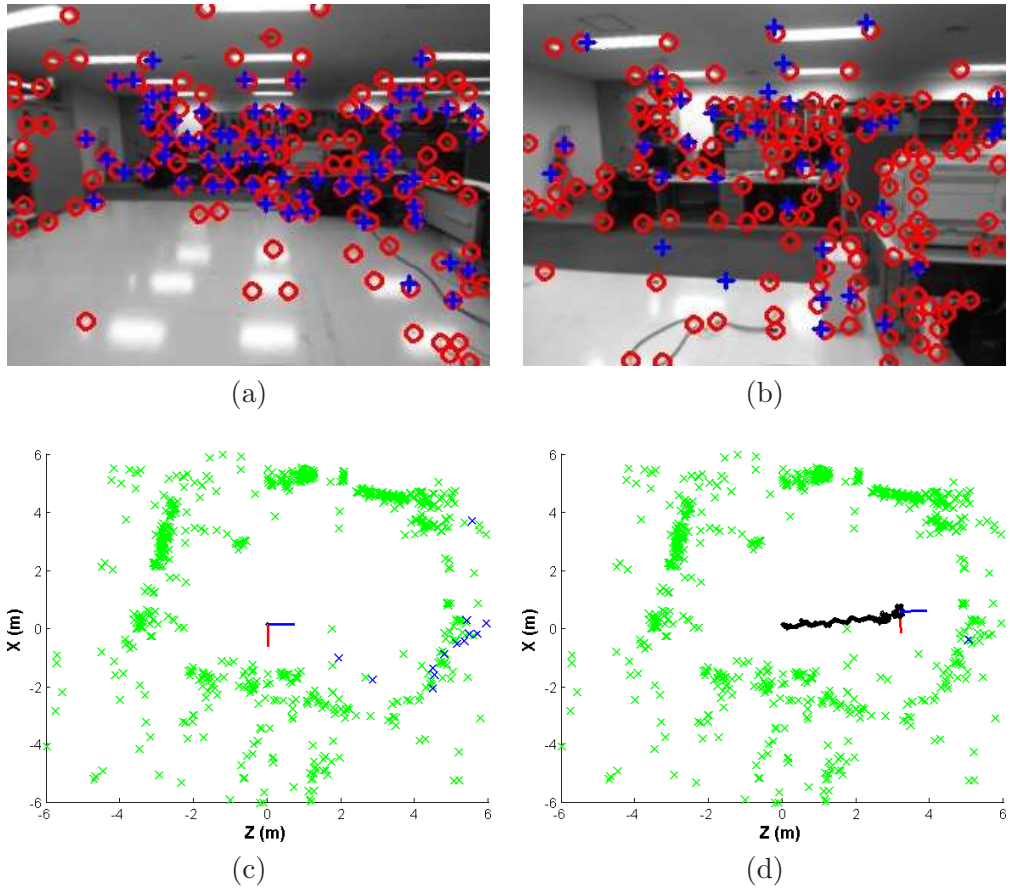


Figure 5.10: Straight line 3 m localization results Tsukuba dataset. (a) and (b) depict the initial and final associated image views of the sequence. The red circles are the detected 2D image features, whereas the blue crosses represent the reprojection of predicted visible 3D points. On the other hand, (c) and (d) depict the initial and final position of the robot and the performed trajectory in the sequence. Best viewed in color.

In this sequence, we measured manually the final position of the robot which was 3.0 m in the Z direction and 0.23 m in the X direction. Compared to the obtained localization results we can observe that we have a higher absolute error in the X axis of 33 cm than in the Z axis, which is about 16 cm for this sequence. These errors are reasonable acceptable, since this area was not captured properly in the map and therefore the PnP problem and data association were more difficult to solve.

Figure 5.11 depicts information about the inliers ratio and number of RANSAC iterations for the square and straight sequences. As expected, we can observe in Figure 5.11(c) and Figure 5.11(d) how the inliers ratio decreases and how the number of RANSAC iterations increases for the straight sequence from the frame 450 approximately. This is because at that point of the sequence the robot started to move into a new area, and therefore both the PnP problem and data association were more difficult to solve. In contrast, the mean inliers ratio for the square sequence 0.9558 is higher than for the straigth sequence one 0.8744. Also, the number of RANSAC iterations is smaller for the square sequence case 2.3808 than for the straight one 7.8144.

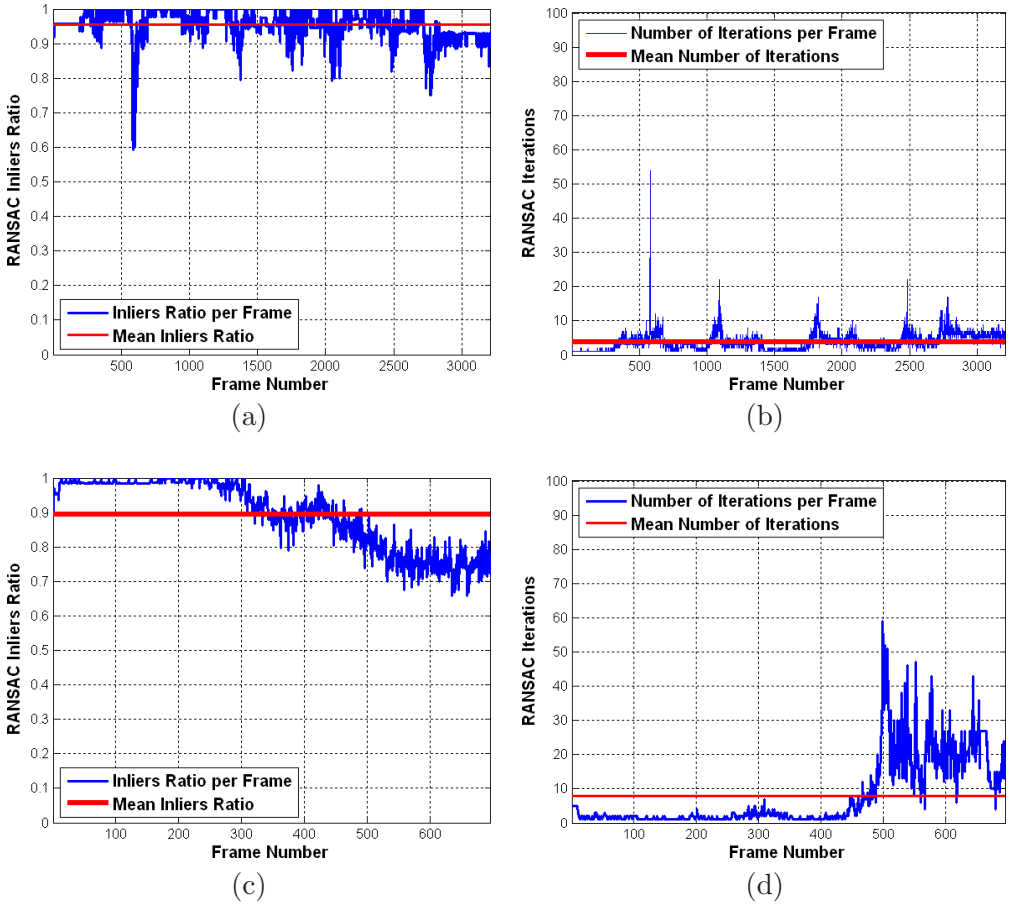


Figure 5.11: Comparison of localization results of the square 2 m size sequence versus straight 3 m sequence, using a prior 3D map and visibility prediction (Tsukuba dataset). (a) Inliers ratio % and (b) Number of RANSAC iterations for the square 2 m size sequence. (c) Inliers ratio % and (d) Number of RANSAC iterations for the straight 3 m sequence.

People Moving in front of the Robot

In typical humanoid robotics laboratories is common that while the robot is performing different tasks in the environment, people may pass close to the robot, occlude some areas of the map or even perform human-robot interaction [Dominey et al., 2007]. In all the mentioned situations it is important that the robot is always localized correctly in the environment.

In this experiment we placed the robot at the origin of the map and planned a straight sequence of 1 m length while some people were walking in front of the robot, without occluding completely the camera field of view. Even though moving people or objects can occlude some areas of the image and the 3D map, we are still able to obtain reliable pose estimates. Outliers are rejected either for apperance information in the data associtions step or by means of RANSAC. Roughly speaking, as long as we have two 2D-3D good correspondences we can estimate the robot's pose. Figure 5.12(a,b) depicts two frames of the sequence where we can appreciate two persons performing common tasks such as going to the printer or picking up the chessboard pattern. At the same time the students were walking in the environment, the robot was moving 1 m straight from its initial position. Figure 5.12(c) depicts the initial position of the robot in the experiment, whereas Figure 5.12(d) depicts the final position.

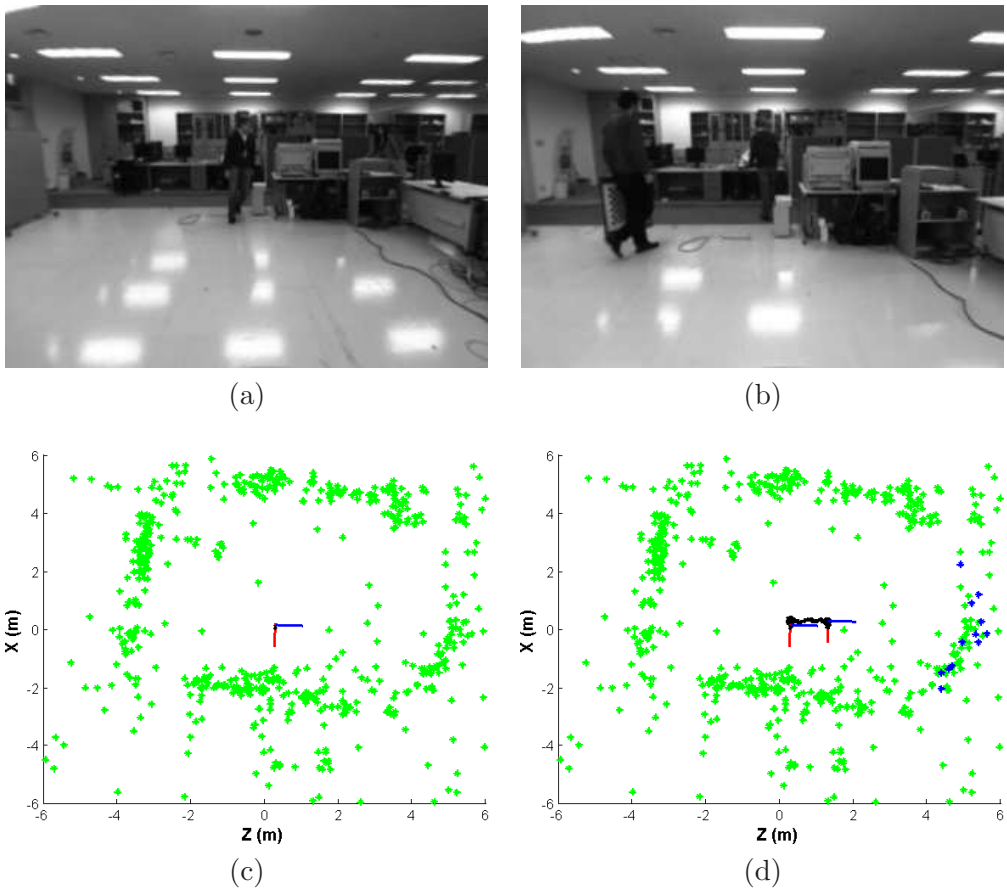


Figure 5.12: People moving in front of the robot. The robot performed a 1 m straight line sequence while at the same time some students were walking in the environment, occluding some 3D map points. Best viewed in color.

Robot Kidnapping

In this experiment, the robot was in a known location and then the robot was suddenly kidnapped, obstructing completely the camera field of view. Although, in the previous sequences (square, straight) the robot did not get lost, it may happen that eventually the robot gets lost if it moves into a new area or the robot is kidnapped as happens in this experiment. In this occasion, for kidnapping recovering, we used the re-localization procedure described in Section 5.3.1. This re-localization procedure takes an average of 25.27 ms per frame. When the robot was kidnapped we moved the robot 1.40 m to the left, and let the system to re-localize itself.

Figure 5.13 (a) and (b) depict the moment of kidnapping and after kidnapping. We can observe in (a) that even a large area of the image is occluded we are still able to obtain some good 2D-3D correspondences, and therefore localization estimates. Figure 5.13 (c) and (d) depict the location of the robot when the kidnapping was going to start and after kidnapping respectively.

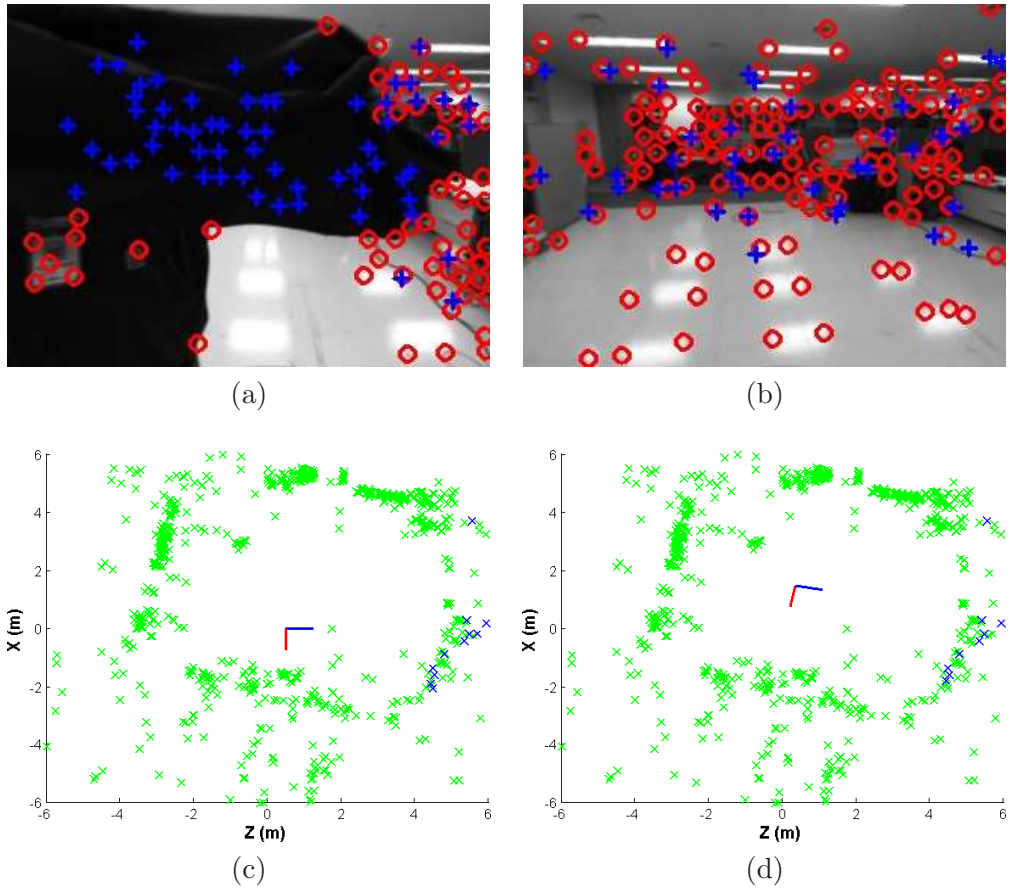


Figure 5.13: Robot kidnapping experiment. (a) The moment when the robot was kidnapped. Notice that even a large area of the image is occluded, we are still able to find good 2D-3D correspondences (b) After kidnapping, the robot is re-localized (c) Robot location in the moment of the kidnapping (d) Robot location after kidnapping. Best viewed in color.

Localization Robustness against changes in Lighting Conditions

In this experiment we want to evaluate the robustness of our vision-based localization approach and the quality of the reconstructed 3D map against changes in lighting conditions. Even though, most of humanoid robots operate under indoors controlled lighting conditions it may happen that under special circumstances lighting conditions can change drastically. Invariance to changes in lighting is even much more important for outdoor scenarios where robots have to explore the same area during different hours of a day. Therefore, it is important that even if the lighting conditions change, the localization of the robot in the environment must be robust and accurate.

For evaluating the quality of our vision-based localization framework against changes in lighting, the robot performed a square 2 m size trajectory with low-intensity lighting conditions, using a prior 3D map that was obtained in normal lighting conditions. Local image descriptors exhibit some invariance against changes in lighting. Invariance to contrast can be achieved by turning the descriptor into a unit vector. For example in [Mikolajczyk and Schmid, 2005], local image descriptors are evaluated under different image transformations including illumination changes. However, not only the descriptor invariance is important, it is also necessary that the feature detector exhibits high repeatability against these changes. If the feature is not detected, it is not possible to match a 3D map element with the corresponding 2D feature, making the data association more challenging.

Figures 5.14 depicts two frames of the environment with normal lighting conditions, where the prior 3D reconstruction was done. Figures 5.14(c,d) depict two frames of approximately the same places of the same environment but under low-intensity lighting conditions. It can be observed the difference in contrast between the two images of the same place under different lighting conditions. Figure 5.15(a) depicts the square 2 m size performed by the robot under low-intensity lighting conditions. Figure 5.15(b) shows the inliers ratio score per frame for the experiment. At the beggining of the sequence the inliers ratio score was small. This was because during the initial frames of the sequence the system was trying to obtain a stable pose initialization. Once the initialization process converged, the inliers ratio score increased and the localization was stable.

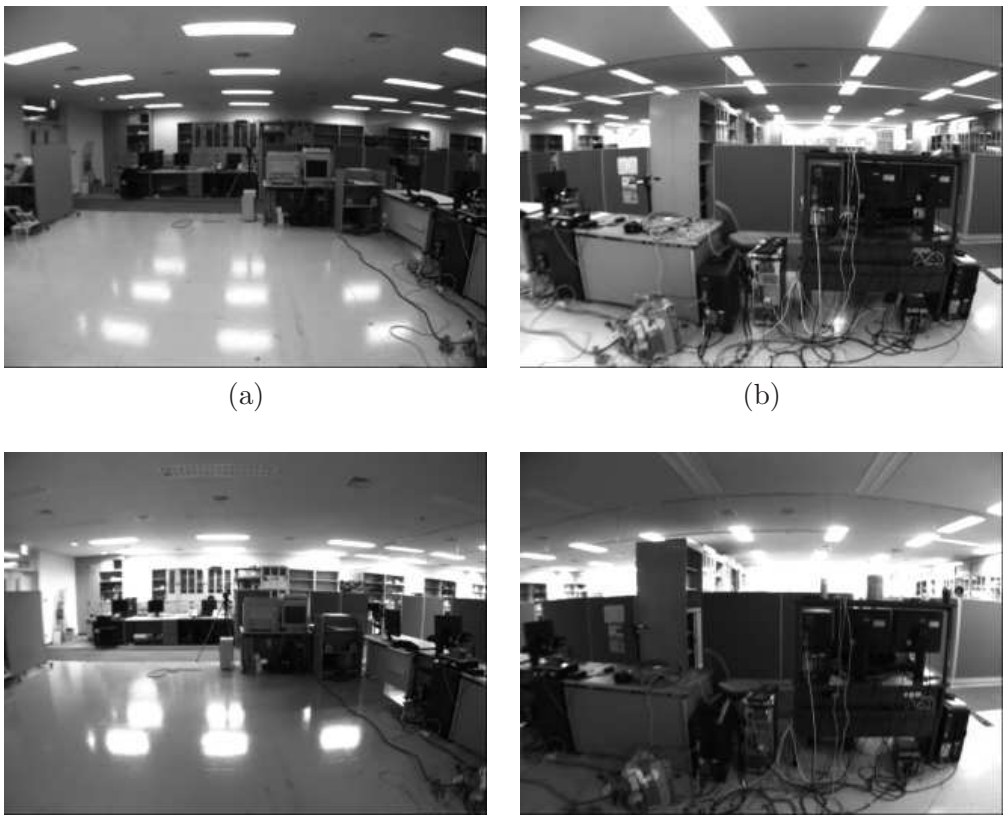
5.4.3 Localization Results: Toulouse Dataset

For this dataset, we performed experiments considering an image resolution of 640×480 and a frame rate of 15 frames per second. We also compare our monocular vision-based localization results with respect to the PTAM approach.

Firstly, we obtained a prior 3D reconstruction of the environment from a square 3 m size sequence that was done by the robot. From this prior reconstruction, visibility was learned and this visibility prediction was used for testing the algorithm under different scenarios. The resulting 3D map comprises of 1768 points and 97 keyframes. In general, the set of experiments from this dataset are more challenging than the ones from the Tsukuba dataset. This is mainly because to moving people and some challenging low-textured areas.

Square 3 m Size Sequence

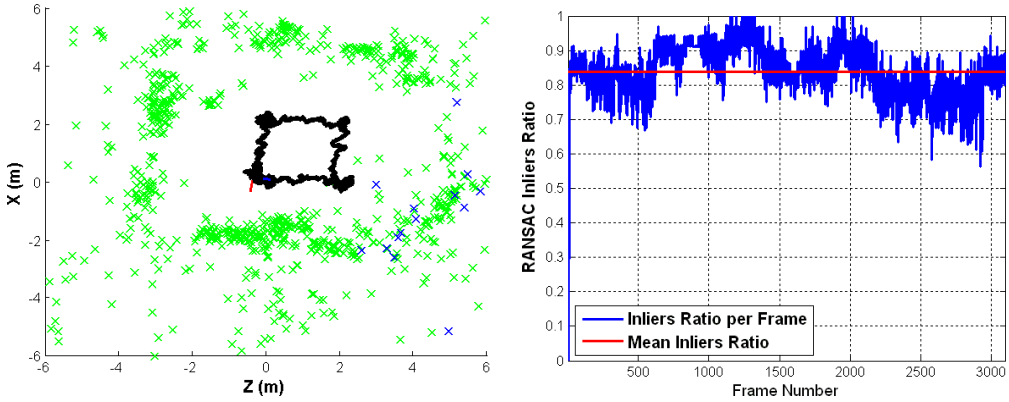
In this experiment we evaluate our localization framework in a square sequence but including dynamic objects such as people. These dynamic objects were not captured in the map computation sequence, and therefore in the evaluation sequence, these objects can



(a) (b)

(c) (d)

Figure 5.14: (a-b) Two frames of the sequence under normal lighting conditions where the prior 3D map was obtained. (c-d) Two captured frames at approximately the same positions as (a-b) but considering low-intensity lighting conditions. Notice the difference in contrast between the image pairs (a-c) and (b-d) respectively.



(a) (b)

Figure 5.15: Evaluation of localization robustness against changes in lighting conditions. (a) The square 2 m size trajectory performed by the robot (b) Inliers ratio % per frame.

occlude some visible 3D points. We consider again the same input parameters for the visibility prediction algorithm as in the Tsukuba experiments, i.e. $K = 10$ and $P_t > 0.20$. However, in order to cope with the new image resolution we chose a threshold value of 4 pixels in the RANSAC process.

Figures 5.16(a,b) depict two frames from the sequence where some people are walking in the environment occluding some visible 3D points from the prior map. Figure 5.16(a) depicts one particular area of the sequence in which vision-based localization is challenging. This is due to the fact that in this area there is a lack of highly textured features. Most of the features are detected in vegetation or around the windows. Due to this lack of texture, the resulting 3D reconstruction in this area can contain higher errors than in other more textured areas of the sequences, since the disparity maps obtained during the map computation are much more sparser and noisier than in other areas. Furthermore, the person walking occludes some predicted visible 3D points. Then, when the robot moves to another more textured area (Figure 5.16(b)), the localization algorithm is able to find correct pose estimates even in the presence of people occluding some predicted visible 3D points.

(a) (b)

Figure 5.16: (a) In this area localization is more difficult, mainly due to the lack of textured features. (b) Even though there are some persons walking in the environment occluding some visible 3D points, the algorithm is able to find correct pose estimates without problems. The red circles are the detected 2D features, the blue crosses represent the reprojection of predicted visible 3D points. The set of inliers putatives after solving the PNP problem are represented by cyan rectangles, whereas the outliers are represented as yellow rectangles. Best viewed in color.

Figure 5.17 depicts the two associated disparity maps for the frames shown in Figure 5.16. The disparity maps were obtained by means of the method proposed in [Konolige, 1997]. As mentioned before, it can be observed how the disparity map is sparser and noisier for the low-textured area (Figure 5.17(a)) than for the textured one (Figure 5.17(b)).

Figure 5.18(a) depicts the performed square 3 m size trajectory done by the robot. In order to stand out the accuracy of the localization per area, the trajectory is depicted in a typical *cool* color space. In this case, the value in the color space is the inliers ratio per frame in the PnP problem. This inliers ratio can be interpreted as an indicator of how good is the localization or how easy to solve is the PnP problem. Other quantities could have been used as for example the covariance result from the PnP problem. We can observe that the inliers ratio tend to decrease when the robot was facing the area depicted by Figure 5.16(a). After this area, the localization is more robust and the inliers

(a) (b)

Figure 5.17: Disparity maps of two frames. Disparity is coded by using a *hot* color space representation. In this representation, close 3D points to the camera are depicted in yellow, whereas far points are depicted in red. Best viewed in color.

ratio increases. In average the inliers ratio per frame was 0.76 for this sequence.

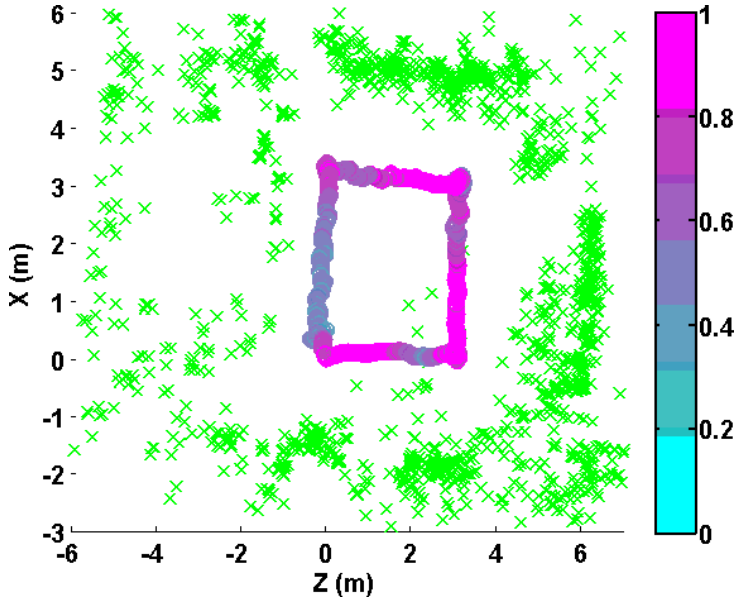


Figure 5.18: Square 3 m Size Localization Results. The trajectory is coded considering a *cool* color space by means of the inliers ratio per frame in the PnP problem. Best viewed in color.

Circle 3 m Diameter Sequence

Now, we evaluate localization considering very different viewpoints from the ones that were captured in the map sequence. In particular, the robot performed a circular 3 m diameter sequence, including very different viewpoints that were not captured in the prior 3D reconstruction. In addition, this experiment was done in a different day than the prior 3D reconstruction. Therefore, there are some changes in the environment, such as for example boxes or a tripod placed in different positions from the original map sequence. Introducing changes in the environment, implies a more difficult localization since our map and localization assume rigid SfM. For example, Figure 5.19 depicts one example of these changes in the environment. We consider the following input parameters for

the visibility prediction algorithm: $K = 10$ and $P_t > 0.05$. The probability threshold is reduced in this case, since in this scenario we have very different camera viewpoints than the ones captured in the map computation sequence and therefore the weights given by the learned kernel function will be much lower.

(a) (b)

Figure 5.19: (a) An image from the map computation sequence (b) An image from approximately the same place as image as (a) but for the circle sequence. Since this sequence was captured in a different day than the map one, there are some changes in the environment, e.g. tripod, chair and white box.

Figure 5.20(a) depicts the performed circular 3 m diameter sequence done by the robot. The trajectory is depicted again considering a *cool* color space coded by means of the inliers ratio per frame in the PnP problem. Again we can observe that the lowest inliers ratios were obtained when the robot was facing the low-textured area depicted by Figure 5.16(a). In average the inliers ratio per frame was 0.49 for this sequence. Although the inliers ratio in this scenario is smaller compared to the square sequence, we need to take into account that viewpoints are very different compared to the map sequence and that some changes in the environment were introduced. Despite of these facts, we are able to obtain a robust localization in real-time, as we will show in the timing evaluation section.

Comparison to PTAM

In this section we compare our localization results with respect to PTAM under the same circular sequence from the previous experiment. At the beginning of the sequence PTAM was able to estimate a correct camera trajectory, but then when the robot performed pure rotation steps the pose estimation error increased considerably and PTAM had problems adding new 3D points to the map. Figure 5.21(a) depicts one frame where PTAM tracking was successful. However, when the robot moved to a low-textured area (Figure 5.21(b)) PTAM tracking got lost. There are several aspects why PTAM tracking got lost at that point, such as: low-textured area, motion blur, camera blinding and a pure rotation step. From this keyframe the PTAM trajectory error started to increase considerably.

Figure 5.22 depicts a comparison of the estimated robot trajectory considering PTAM, the motion capture system and monocular vision-based localization results with a prior 3D map. For the monocular vision-based localizaton results with visibility prediction, we consider two different prior maps: one obtained from a square sequence as described in Section 5.4.3 and another one obtained from the circular 3 m diameter sequence. We can observe in the Figure that PTAM obtained good trajectory estimates at the beginning of

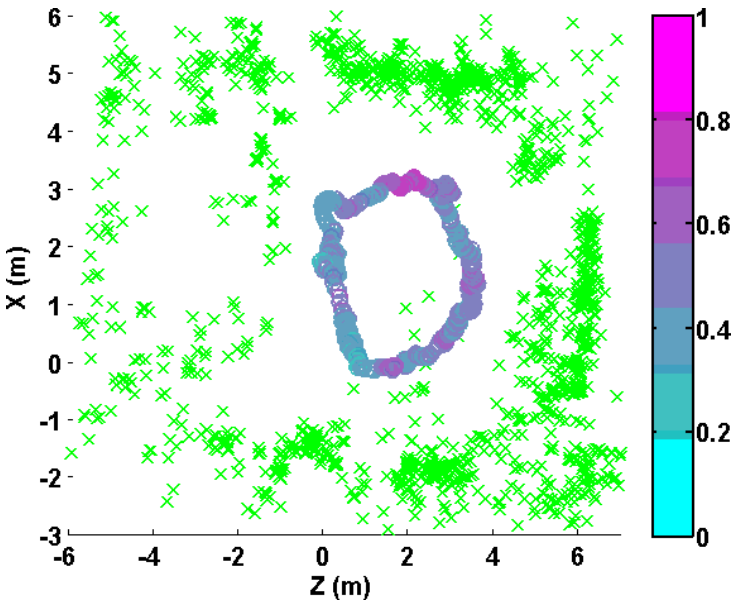
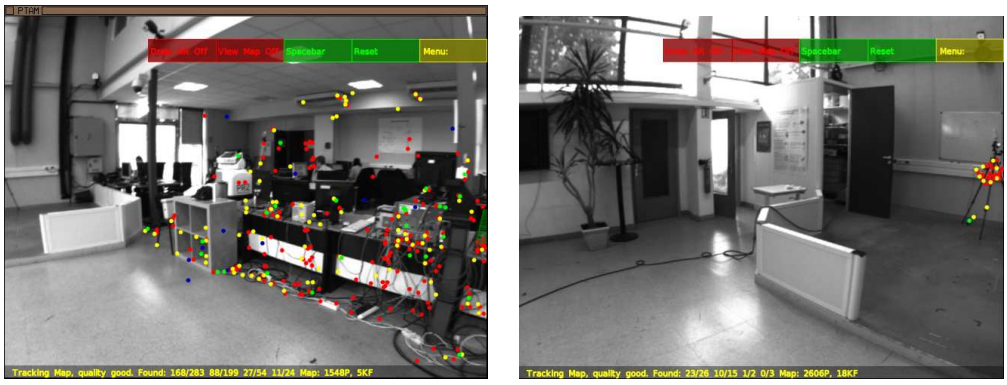


Figure 5.20: Circle 3 m Diameter Localization Results. The trajectory is coded considering a *cool* color space by means of the inliers ratio per frame in the PnP problem. Best viewed in color.



(a) (b)

Figure 5.21: PTAM Tracking Results: (a) One frame of the circular sequence where PTAM tracking was successful (b) One frame where PTAM tracking had severe problems and new 3D map points can not be added to the map.

the sequence, but as soon as the robot was doing pure rotation steps the error increased considerably. Also the PTAM error compared to the motion capture system in the vertical axis was about 0.81 m at the moment when the tracking was completely lost. It can also be observed that the monocular localization results with a prior 3D map obtained a very similar trajectory compared to the motion capture system.

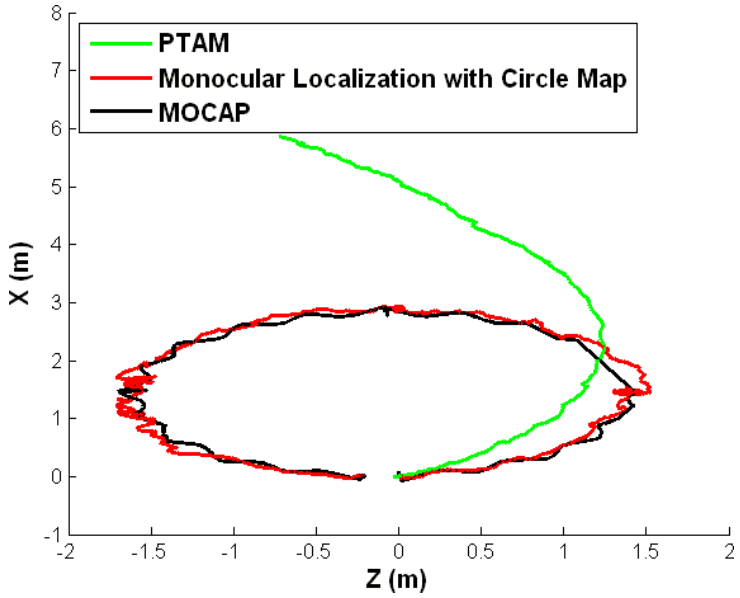


Figure 5.22: Comparison to PTAM Localization Results. Best viewed in color.

5.4.4 Timing Evaluation

In this Section, we show a timing evaluation of our vision-based localization algorithm for both the Tsukuba and Toulouse datasets. All timing results in this Section were obtained on a Core i7 2.87GHz desktop computer using a single CPU.

Figure 5.23 depicts timing results for the localization experiments of the square and straight sequence from the Tsukuba dataset. We can observe that in average the mean computation time for the square sequence, 5.35 ms, was slightly smaller than for the straight one, 6.49 ms. For a faster localization, we only detect 2D features at the finest scale-space level. In the environment we carried out our experiments, we observed that with one single-scale level we have enough amount of features to perform robust localization.

Table 5.5 shows mean computation times for the analyzed experiments, but describing timing evaluation for the main steps involved in the localization algorithm. In general, most time consuming steps per frame are feature detection, descriptors computation and pose estimation. Initialization only takes place during the first frame or an initial transitory time of the sequence until the robot detects that it is in a known area with high confidence.

Figure 5.24 depicts timing results for the localization experiments of the square and circular sequence from the Toulouse dataset. For the square sequence we obtained a mean computation time per frame of 20.31 ms. For the circular sequence the computation time is higher (30.36 ms). This is mainly because the PnP problem is more difficult to solve, due to the fact that viewpoints are very different from the ones captured in the map sequence and the changes in the environment also make the PnP problem more challenging.

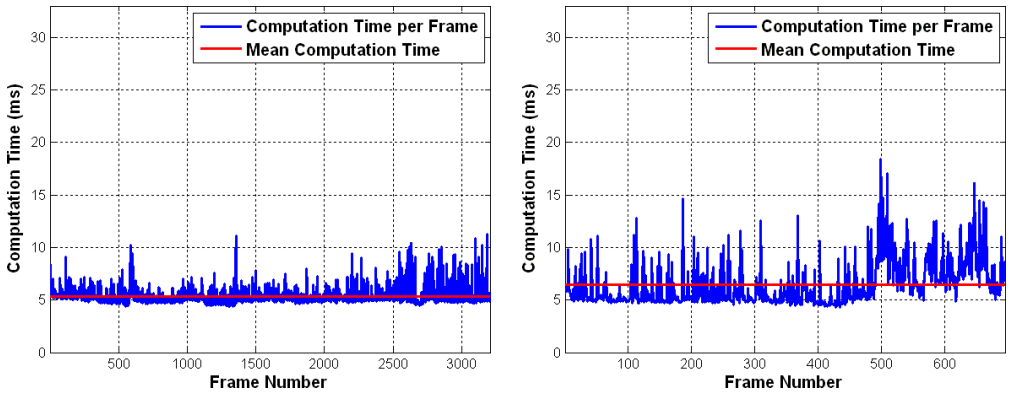


Figure 5.23: Monocular Vision-Based Localization Timing Evaluation Tsukuba dataset: (a) Computation times per frame for the 2 m square sequence (b) Computation times per frame for the 3 m straight sequence.

Localization Step	**Square Time (ms)**	**Straight Time (ms)**
Initialization	1636.86	1641.99
Undistortion and Rectification	0.76	0.87
Feature Detector	2.43	2.62
Feature Descriptor (16)	0.94	1.04
Reprojection 3D Points	0.12	0.14
Data Association	0.10	0.09
Pose Estimation	1.00	1.72
Total per Frame	**5.35**	**6.49**

Table 5.5: Monocular vision-based localization mean computation times per frame (Tsukuba dataset). For this dataset the image resolution was 320×240.

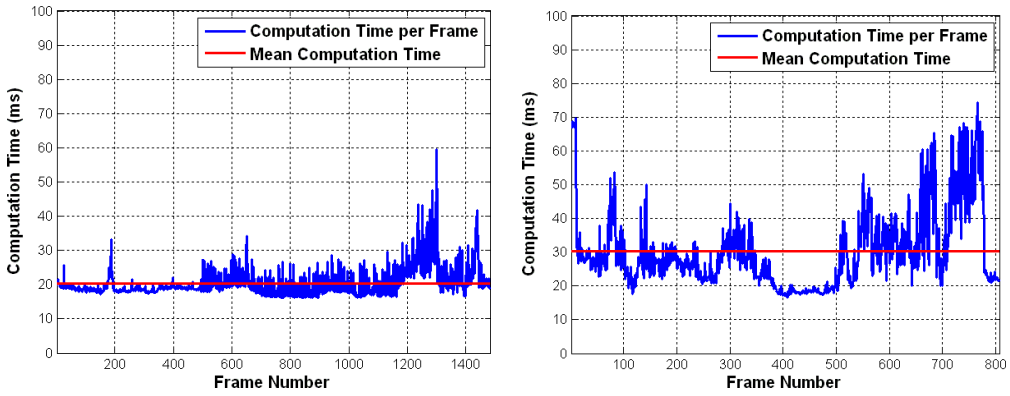


Figure 5.24: Monocular vision-based localization timing evaluation Toulouse dataset: (a) Computation times for the 3 m size square sequence (b) Computation times for the 3 m diameter circular sequence.

Table 5.6 shows mean computation times per frame for both sequences of the Toulouse dataset. Since in the Toulouse dataset we are using a 640×480 image resolution, the feature detection and description steps are more time consuming than for the Tsukuba dataset. In the same way, since the image resolution is higher, the detected number of 2D features is also higher and therefore the PnP problem has a higher number of putative correspondences. In those areas where we have enough textured features, the PnP problem is solved very fast in real-time. However, in some particular areas where it may be difficult to find good 2D-3D correspondences the PnP problem can take more time to be solved efficiently (e.g. low-textured areas of the circular sequence).

Localization Step	**Square Time (ms)**	**Circle Time (ms)**
Initialization	2540.93	2723.15
Undistortion and Rectification	3.36	2.95
Feature Detector	10.28	9.81
Feature Descriptor (16)	2.79	2.15
Reprojection 3D Points	0.29	0.28
Data Association	0.55	0.51
Pose Estimation	3.02	14.64
Total per Frame	**20.31**	**30.36**

Table 5.6: Monocular vision-based localization mean computation times per frame (Toulouse dataset). For this dataset the image resolution was 640 × 480.

In general, with a small image resolution 320×240 we can obtain accurate localization

results in few ms. With a higher resolution such as 640×480 the localization results can be very accurate, although the computation time will also increase considerably. For all the analyzed experiments, mean computation times per frame are below real-time demands (30 Hz). If certain applications have some time restrictions, one can always fix a smaller threshold for the number of iterations of the RANSAC step. Usually if the set of putative matches is good, only few iterations are necessary to solve the PnP problem efficiently. Figure 5.25 depicts the number of RANSAC iterations for the square and circular sequence from the Toulouse dataset. In those experiments we fixed a maximum threshold of 400 iterations in the RANSAC process.

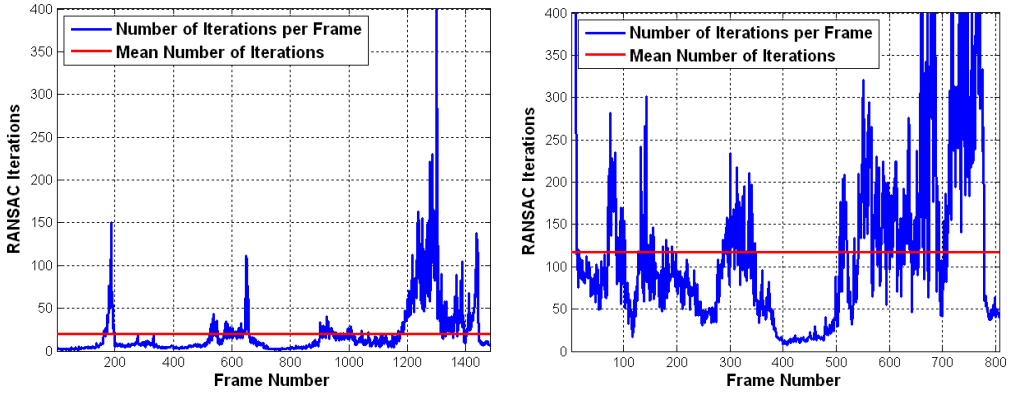


Figure 5.25: Monocular vision-based localization timing evaluation Toulouse dataset: (a) Number of RANSAC iterations per frame for the 3 m size square sequence (b) Number of RANSAC iterations per frame for the 3 m diameter circular sequence.

5.5 Conclusions and Future Work

In this chapter, we have presented a vision-based localization algorithm that works in real-time (even faster than 30 Hz) and provides localization accuracy about the order of cm. We first build a 3D map of the environment by using stereo visual SLAM techniques, and perform visibility learning over the prior 3D reconstruction. Then, for fast vision-based localization we use visibility prediction techniques for solving the PnP problem and obtaining the location of the robot with respect to a global coordinate frame. We measured the accuracy of our localization algorithm by comparing the estimated trajectory of the robot with respect to ground truth data obtained by a highly accurate motion capture system. We also compared our algorithm with respect to other well-known state of the art SfM algorithms such as PTAM, showing the benefits of our approach.

In addition, we are interested in improving the capabilities of our vision-based localization algorithm towards the goal of life-long localization and mapping. For example, if the robot moves into a new area, which was not mapped in the prior 3D reconstruction, the robot should detect automatically this new area and start a 3D reconstruction of this new environment by means of the stereo visual SLAM method described in Chapter 2. Then, this new reconstruction will be merged with the prior one, and finally the resulting 3D reconstruction will be optimized by means of BA.

We are also interested in combining visibility prediction with the Bayesian Surprise and landmark detection framework [Ranganathan and Dellaert, 2009]. In this way we

can model in a probabilistic way when the robot discovers a new *surprising* area and then adding this area into the whole reconstruction. Indeed, we also think that Bayesian surprise can be also useful for detecting a new object in the prior 3D reconstruction, and once the robot detects this new object, the robot can start a 3D reconstruction of the object using the localization information as a prior.

In this work, we have mainly put our focus in real-time vision-based localization. However, we think that the accuracy in localization can be increased if we fuse the information from our vision-based localization with the odometry information of the robot. Also the image resolution and length of the descriptors can be increased, but the price to pay is higher computational demands, that may prevent the algorithm from real-time performance. For example, an affine invariant detector such as ASIFT [Morel and Yu, 2009] can be used in order to make the algorithm more robust against new camera viewpoints. However, the main limitation is the higher computational demands of these kind of invariant feature detectors.

In the next future, we are interested in using our approach in related vision-based humanoid robotics problems such as control [Blösch et al., 2010], autonomous 3D object modeling [Foissote et al., 2010] or footstep planning [Perrin et al., 2010]. We think that our real-time vision based localization can improve considerably some previous humanoid robotics applications where vision-based localization was not exploited in all its capabilities.

Chapter 6

Visual SLAM and Vision-Based Localization for the Visually Impaired

Autonomous navigation is of extreme importance for those who suffer from visually impairment problems. Without a good autonomy, visually impaired people depend on other factors or other persons to perform typical daily activities. According to [Loomis et al., 2001], the most fundamental needs of visually impaired users include access to information (mainly in written format), accessibility to the environment and independence of movement. Within this context, a system that can provide robust and fast localization of a visually impaired user in urban city-like environments or indoor ones is much more than desirable. A good localization of the user in the environment will lead to a more robust and safer navigation and therefore, the mobility of visually impaired people can be increased considerably by the aids that these technological advances can offer them.

Nowadays, most of the commercial solutions for visually impaired localization and navigation assistance are based on Global Navigation Satellite Systems (GNSS). Among these systems, the Global Positioning System (GPS) is probably the most extended one. However, these solutions are not suitable enough for the visually impaired community mainly for two reasons: the low accuracy in urban-environments (errors about the order of several meters) and signal loss due to multi-path effect or line-of-sight restrictions. GPS does not work if an insufficient number of satellites are directly visible. Therefore, GPS cannot be used in indoor environments. In addition, in urban environments GPS signals are usually weak, since are blocked by buildings or even foliage [Feiner et al., 1997].

One of the main challenges for visually impaired navigation assistance systems is to obtain an accurate information about the location and spatial orientation of the user in a large-scale environment. Among the new modalities of localization techniques, computer vision-based approaches offer substantial advantages with respect to GPS-based systems and constitute a promising alternative to address the problem. By means of multi-view geometry and Structure from Motion (SfM) methods, one can obtain extremely accurate 3D models of cities [Agarwal et al., 2009] or urban environments [Pollefeys et al., 2008]. Then, these large 3D models can be used later for vision-based localization and navigation or related applications such as augmented reality. Furthermore, is also possible to build an incremental map of the environment by means of visual SLAM techniques [Saéz et al., 2005; Pradeep et al., 2010] providing at the same time the location and spatial orientation of the user within the environment. In addition, compared to other sensory modalities

computer vision also provides a very rich and valuable perception information of the environment such as for example obstacle detection [Saéz and Escolano, 2008] or 3D scene understanding [Geiger et al., 2011a].

In this chapter, we will describe visual SLAM and vision-based localization with a prior 3D map techniques that can be used for aiding visually impaired people during navigation. A vision-based localization system would be able to provide an accurate pose aligned with the head orientation, which can enable a system to provide the visually impaired with information about their current position and orientation and/or guide them to their destination through diverse sensing modalities [Walker and Lindsay, 2006].

Most of visual SLAM systems in the literature [Konolige and Agrawal, 2008; Mei et al., 2010] need to assume static features in the environment and that a dominant part of the scene changes only with the camera motion. As a result, these approaches are prone to failure in crowded scenes with many independently moving objects. Even though some of the outliers can be detected by geometric constraints validation, one needs to take special care about not introducing any outlier in the 3D reconstruction process or otherwise the estimated map and camera trajectory can diverge considerably from the real solution. In this chapter, we will show how stereo visual SLAM algorithms for crowded and dynamic environments, with many independent moving objects, can be improved by means of the detection of moving objects, thanks to a dense scene flow [Vedula et al., 1999] representation of the environment.

We will show monocular vision-based localization with a prior 3D map results in office-like environments. In addition, we will show some experimental results of the visual SLAM algorithm with moving objects detection in extremelly crowded and dynamic environments, such as inside the Atocha railway station (Madrid, Spain) and in the city center of Alcalá de Henares (Madrid, Spain). The rest of the chapter is organized as follows: In Section 6.1 we will review the main approaches related to localization of visually impaired users considering different sensory modalities, dedicating special importance to vision-based approaches. The main characteristics of the monocular vision-based localization system for the visually impaired for indoor office-like environments is detailed in Section 6.2. The localization results for office-like environments are discussed in Section 6.2.1. Then, in Section 6.3 we will explain how visual SLAM estimates can be improved in highly dynamic environments by means of a dense scene flow representation of the environment. The stereo visual SLAM results in crowded and dynamic environments are described in Section 6.3.1. Finally, main conclusions and future work are described in Section 6.4.

6.1 Related Work

In the literature, there have been different approaches related to localization and navigation assistance for visually impaired people that employ different sensory modalities. Each of the different sensory modalities provides certain advantages and drawbacks. Besides, some approaches propose the fusion of different sensory modalities [Hesch and Roumeliotis, 2010]. Most common employed sensors are: GPS, acoustic, radio frequency (RF), laser, vision or the fusion of several of them. Now, we will review the main approaches considering the different sensory modalities.

6.1.1 GPS-Based Systems

There exist several approaches for aiding visually impaired users during navigation by means of GPS information. Most of them share the problems mentioned before: low accuracy in urban-environments, signal loss, multi-path effect or line-of-sight restrictions due to the presence of buildings or even foliage. In [Petrie et al., 1996] the authors proposed one of the first GPS-based navigation systems for visually impaired users in which a speech synthesizer was used to describe city routes. In a similar way, Feiner et al. [1997] presented a prototype that used GPS information in the context of 3D augmented reality applications for exploring urban environments.

The work of [Loomis et al., 2001] constitutes a very interesting survey about GPS-based navigation systems for the visually impaired. In the mentioned work they state that commercial GPS accuracy considering good satellite visibility conditions is limited to approximately 20 m. Even though this error can be enough for certain applications such as road vehicle navigation, this error is very high and can represent dangerous situations for visually impaired users when they are walking into unknown or unfamiliar environments. One can obtain better GPS accuracy by employing differential corrections. In Differential GPS (DGPS), correction signals from a GPS receiver at a known fixed location are transmitted to the mobile receiver in order to correct its position. However, differential correction requires a separate receiver and the service is not available in many locations.

Oh et al. [2004] showed a Bayesian particle filtering approach for the localization of visually impaired people using a GPS sensor, incorporating the knowledge of map-based priors into the localization framework. Semantic available information of the environment is used to bias the motion model of the posterior density location towards high probability areas. By means of adding information of the environment as a prior, localization can be done in occassions when the main sensor is innacurate or unreliable. Figure 6.1 depicts one example of a prior map of the environment which is divided into different probability areas.

6.1.2 Audio-Based Systems

The works of Walker and Lindsey [2005; 2006] study the application of spatialized non-speech beacons for navigation of visually impaired users in the environment. In addition, there is also an exhaustive analysis about how sound timbre, waypoint capture radius and practice affect to navigation peformance. The main disadvantages of placing a network of sound beacons within the environment are the cost of installing and mantaining the network and the limited coverage. The research described in the mentioned works belongs to the Georgia Tech System for Wearable Audio Navigation (SWAN)[1], a mobility tool for the visually impaired.

6.1.3 RF-Based Systems

Kulyukin et al. [2004] proposed a robotic system based in Radio Frequency IDentification (RFID) for aiding the navigation of visually impaired users under indoor environments. The main purpose of RFID technology is to transmit the location identifier of an object or a place within the environment. RFID tags are small devices that can be

[1]For more information please check the following url: http://sonify.psych.gatech.edu/research/SWAN/

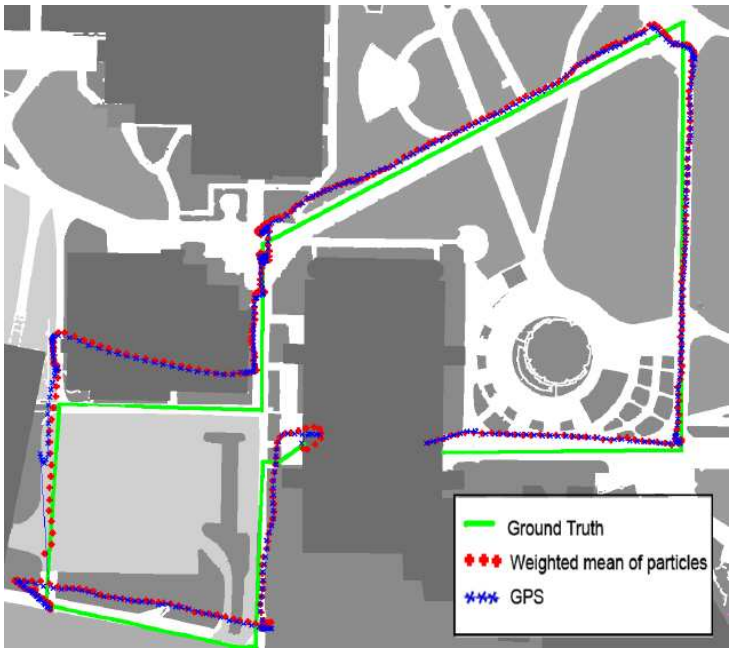


Figure 6.1: Map-based priors for localization: The areas with brighter colors correspond to high probability areas, whereas the darker colors correspond to low probability areas. The black zones denote the zero probability areas which may include the buildings and shrubs.

attached to any object in the environment or even worn on clothing. They have antennas that can receive or transmit information by means of RF. Passive RFID tags do not require any external power source or direct line of sight with respect to the RFID reader. They are activated by the spherical electromagnetic field generated by the RFID antenna within a radius of approximately 1.5 m. Similar to the audio-based works described in Section 6.1.2 the main drawback of RFID approaches is the design of a dense network of location identifiers and the associated cost of installing and maintaining the network.

6.1.4 Laser-Based Systems

In [Hesch and Roumeliotis, 2010], the authors proposed the fusion of different sensory modalities towards a portable indoor localization aid for the visually impaired. In particular, they fused the information from a 2D laser scanner, gyroscopes and a foot-mounted pedometer by means of an Extended Kalman Filter (EKF) framework. Figure 6.2 depicts the sensing package mounted near the handle of the mobility cane. In addition, a portion of the 2D laser scan plane is illustrated, along with the intersection between the scan plane and the floor.

They use a two-layer framework to estimate the location and orientation of the user in the environment. In the first layer, the 3D attitude of the mobility cane is estimated using the information from the 3-axis gyroscope and from the 2D laser scanner. Then, the heading direction of the person is computed by passing the cane's yaw estimate through a low-pass filter. In the second layer, the position of the person in the environment is updated using laser-scanner observations of corner features in the environment. In this work, corner features are typical wall intersections at hallway junctions. By matching the laser-scanner observations to known corners in a prior map, they are able to estimate the position of the user in the environment. This map of corner features needs to be

Figure 6.2: A portable indoor localization aid for the visually impaired based on the fusion of several sensory modalities (gyroscopes, pedometer and a laser scanner).

computed beforehand, either from the building blueprints or from other mapping techniques. Computer vision, seems to be here a much better alternative, since the maps can be updated while the user is localized in the environment by means of visual SLAM techniques [Davison et al., 2007; Mei et al., 2010]. In addition, the data association between the prior map corners and the laser observations can be critical in typical cluttered and crowded indoor environments with many independent moving objects.

6.1.5 Vision-Based Systems

In [Saéz et al., 2005], the authors presented one of the first stereo vision systems towards a 6-Degrees of Freedom (DoF) SLAM for the visually impaired. In their work, egomotion estimation is done by a point matching algorithm integrating 3D and 2D information. Mapping is done through a randomized global entropy minimization algorithm, considering othogonal indoor scenarios, with difficult extension to non-orthogonal and more complex environments. In [Saéz and Escolano, 2008], the mentioned visual SLAM system was used for predicting the next movement of visually impaired users and maintaining a local 3D map of the vicinity of the user. This 3D map information is used to evaluate and detect possible over-head obstacles during the navigation of visually impaired users. Figure 6.3 depicts an image of the wearable stereo device that was used in [Saéz et al., 2005], that comprises of a stereo camera and a laptop, both connected through a fireware cable.

Pradeep et al. [2010] described a head-mounted stereo vision system for the visually impaired. By means of stereo visual odometry and feature based metric-topological SLAM,

Figure 6.3: Wearable stereo device for 6-DoF SLAM for the visually impaired.

they create a 3D map representation around the vicinity of the user that can be used for obstacle detection and for traversability maps generation.

In [Liu et al., 2010], the authors described an indoor vision-based localization system based mainly on topological and appearance information. The first step of the algorithm is the registration of a reference sequence, in which orientation data and 2D keyframe positions are provided by means of an Inertial Measurement Unit (IMU). From this pre-recorded sequence, GIST [Oliva and Torralaba, 2001] and SURF [Bay et al., 2008] features are extracted and referenced to their respective keyframes. Then, during online localization experiments, GIST features are used to find similar keyframe images to the current view, and then in a second stage, SURF features are used to provide a set of matching correspondences between the features stored in the map and the detected features in the new image. The current camera position is estimated by means of Hidden Markov Model (HMM) [Russell and Norvig, 2003].

In [Treuillet et al., 2007], the authors proposed a vision-based localization algorithm for the visually impaired that is similar in spirit to our approach. The localization system is based on the work of Royer et al. [2005] and relies on two different steps: firstly, a *learning* stage in which a 3D map of the environment is computed in a batch mode by means of hierarchical Bundle Adjustment (BA), and secondly, a real-time localization approach for navigation assistance. During the localization step, the system provides information about the orientation and position of the user in the map. The data association between 3D map elements and 2D features is done by simply choosing the closest keyframe (in the sense of shortest Euclidian distance between the camera centers) with respect to the previous camera pose. Then, matching candidates are selected by means of cross correlation in an area of interest centered on the reprojection of the 3D map points onto the new image. The system was tested by visually impaired users considering an indoor

and an outdoor scenario without any loop closures. The length of the walking trips in the different scenarios was about 150 m for the outdoor trip and 70 m for the indoor trip. Their experimental results showed the effectiveness of the vision-based localization system to keep the user in a navigation corridor less than 1 m width along the intended path.

The main drawback of the localization framework described in [Royer et al., 2005; Treuillet et al., 2007] is the data association step. In the data association step, only the closest keyframe with respect to the previous camera pose is selected, and therefore, only those 3D points seen by the selected keyframe will be re-projected onto the image plane for estimating the new camera pose. This implies that the new camera viewpoints must be very similar to the stored keyframes in the prior reconstruction. With the visiblity prediction approach described in Chapter 4, we fuse the information from several keyframes. Then, for each 3D map point, we can infere a probability about how likely that 3D point will be visible by a given query pose, yielding a more robust and faster data association and localization.

6.2 Vision-Based Localization in Indoor Office-Like Environments

In this section, we will describe our overall vision-based localization framework for providing location and orientation information to visually impaired users in indoor cluttered office-like environments. Our approach is very similar to the monocular vision-based localization approach described in Section 5.3 for humanoid robots. However, the main difference is that for the experiments described in Section 6.2.1 we do not use any appearance descriptors in the matching process. In contrast, we perform the data association by means of the Euclidean norm between the position of a 2D detected feature and a projected map element. In this way we can analyze the benefits that are obtained by the *smart* use of the geometric information. In other words, any benefits to be demonstrated in these experiments are expected to be obtained from the improvements on the geometric information use, isolated from any advantages due to appearance-based matching methods.

A map is commonly defined as a set of high quality 3D points reconstructed from the images. Such a 3D map comprises of the location of each landmark and can be obtained through visual SLAM techniques e.g., [Konolige and Agrawal, 2008; Mei et al., 2010]. Even though stereo camera may be used for the 3D map reconstruction, we would focus on localization problem based on monocular vision and a prior 3D map of the environment. Given a prior map of 3D points and perceived 2D features in the image, our problem to solve is the estimation of the camera pose with respect to the world coordinate frame, i.e. we need to solve the Perspective-n-Point (PnP) problem. More in detail, the overall localization system works through the following steps:

1. While the camera is moving, the camera acquires a new image from which a set of image features $Z_t = \{z_{t,1} \dots z_{t,n}\}$ are detected by a feature detector of choice. In our experiments we used a multiscale version of the well-known Harris corner detector [Harris and Stephens, 1988].

2. Then, by using the visibility predicition algorithm, a promising subset of highly visible 3D map points is chosen and re-projected onto the image plane based on the estimated previous camera pose θ_{t-1} and known camera parameters.

3. Afterwards, a set of putative matches C_t are formed where the i-th putative match $C_{t,i}$ is a pair $\{z_{t,k}, x_j\}$ which comprises of a detected feature z_k and a map element x_j. A putative match is created when the error in Euclidean norm between the position of a 2D detected feature and a projected map element is very low.

4. Finally, we solve the pose estimation problem minimizing the following cost error function, given the set of putative matches C_t:

$$\arg\min_{R,\mathbf{t}} \sum_{i=1}^{m} \|z_i - K\left(R \cdot x_i + \mathbf{t}\right)\|_2 \tag{6.1}$$

where $z_i = (u_L, v_L)$ is the 2D image location of a feature in the left camera, x_i represents the coordinates of a 3D point in the global coordinate frame, K is the left camera calibration matrix, and R and t are respectively the rotation and the translation of the left camera with respect to the global coordinate frame. The PnP problem is formulated as a non-linear least squares procedure using the LM algorithm implementation described in [Lourakis, 2004]. The set of putative matches may contain outliers, therefore RANSAC is used in order to obtain a robust model free of outliers. The RANSAC-based framework described above is very popular and has been used successfully by many authors [Tariq and Dellaert, 2004; Yuen and MacDonald, 2005].

6.2.1 Localization Results in Indoor Office-Like Environments

In this section, we will show monocular vision-based localization experiments considering that a prior 3D map of the environment is available in indoor office-like environments. Our monocular vision-based localization algorithm uses the visibility prediction algorithm described in Chapter 4, to perform an efficient and fast data association. For these indoor office-like experiments we used a stereo camera of 15 cm baseline and an image resolution of 320×240 pixels. The acquisition frame rate was about 30 frames per second.

We designed our experiments in such a way that we can analyze the benefits that are obtained by the *smart* use of the geometric information. In other words, any benefits to be demonstrated in these experiments are expected to be obtained from the improvements on the geometric information use, isolated from any advantages due to appearance-based matching methods. A 3D map in a dense cluttered environment was computed using a stereo visual SLAM algorithm. The stereo visual SLAM algorithm is used to provide testbeds for localization as well as to provide training data for learning proper visibility kernel functions. During the map computation process, the different training poses from which each map element is seen, are memorized so as to be able to predict the visibility by means of the probabilistic visibility modeling during the localization stage. We show monocular vision-based localization results for the training and test datasets. Additionally, to stand out the contributions of our method, we compare our idea with two different methods:

- **Brute Force**: Under this assumption all the map features are re-projected onto the image plane for a given pose. Besides, only the features that are predicted to lie within the image plane, are considered to form the set of putatives to be used for pose estimation.

- **Length and angle heuristic**: Feature visibility is calculated considering the difference between the viewpoint from which the feature was initially seen and a new viewpoint. This difference in viewpoint has to be below some length and angle

ratio, and predicted to lie within the image, in order to predict the feature as visible. Usually the feature is expected to be visible if the length ratio $|h_i|/|h_{orig}|$ is close enough to 1 (in practice between 5/7 and 7/5 and the angle difference $\beta = \cos^{-1}((h_i \cdot h_{orig})/(|h_i||h_{orig}|))$ is close to 0 (less than 45° in magnitude).

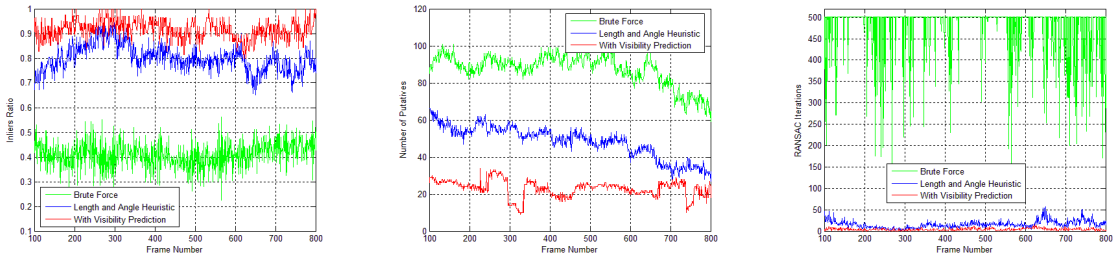
Considering the first of the above approaches, a *brute force* approach will yield in a high number of outliers and localization errors under very dense maps (thousands of features), whereas the second of the approaches can yield erroneous localization when the camera is facing occluded areas. The second approach has been widely used in the literature such as in [Davison and Murray, 2002; Davison et al., 2007], since it is very easy to compute and provides good results, since after the visibility prediction, matching is performed using 2D image templates. However, one of the drawbacks of this criteria is that can not deal with occlusions since it assumes a *transparent* world. This in fact, can be a problem for long term localization under cluttered environments with occlusions, such as the ones we are interested for the visual impaired (e.g. cities, underground stations, offices, etc.)

The training dataset is a large sequence of 5319 frames in which a sparse map with about 1316 3D points was obtained. The test dataset is a separate small sequence recorded in the same space and comprises of 2477 frames, including some camera views that were not fully captured in the training dataset. The data presents full 6-DoF motion where effort was put to imitate the common motion of a visually impaired person as much as possible. Figure 6.4 depicts the kind of environment where we have tested our localization experiments.

Figure 6.4: Some frames of the localization experiments in office-like environments.

In our experiments we use an adaptive threshold version of RANSAC to automatically determine the number of RANSAC iterations needed [Hartley and Zisserman, 2000]. The distance threshold that is used to create a putative match is set to 4 pixels. In addition we only consider visibility prediction of the 20 KNNs since we have found experimentally that this number of neighbors is enough for predicting visibility. Our experimental results shown in Figure 6.5 highlights the benefits of our method where our result is shown to provide less number of higher-quality putative matches, which eventually leads to faster and more accurate RANSAC computation. In detail, Figure 6.5 depicts the inliers ratio (a), the number of putatives per frame (b) and the number of RANSAC iterations (c) during some of the first frames of the test sequence. The highest inliers ratio is obtained using our visibility prediction approach and this ratio is normally above 80%. We set the number of RANSAC iterations to a maximum of 500 for computational purposes. As it can be seen in Figure 6.5, the number of iterations for the *brute force* case is very close to this bound, whereas for the other experiments the number of iterations is much lower, obtaining less than 20 iterations per frame for the visibility case.

In Table 6.1, the information about the mean inliers ratio, mean number of putatives



(a) (b) (c)

Figure 6.5: Test sequence results: (a) Inliers ratio, (b) number of putatives and (c) number of RANSAC iterations per frame. Best viewed in color.

and RANSAC iterations per frame is shown for the training and test datasets respectively where we can again observe the computational benefits of our method. Figure 6.6 and 6.7 show the localization results for the training and test sequence respectively, with respect to the ground truth data obtained by the visual SLAM algorithm. Map 3D points are characterized by a vector $x_i = \{X\ Y\ Z\}$ in a world coordinate frame (map 3D points are represented by orange dots in the next figures). Each camera pose is parametrized by means of a vector $\theta_i = \{X\ Y\ Z\ q_0\ q_x\ q_y\ q_z\}$ (translation and orientation given by a unit quaternion). We do not show any figure for the brute force case, since this approach fails to provide accurate localization for both of the datasets.

Training	**% Inliers**	**# Putatives**	**# Iterations**
Brute Force	0.6952	67.2181	461.7096
Heuristic	0.7392	36.7744	52.7885
Visibility Prediction	0.8335	26.3897	6.4221
Test	**% Inliers**	**# Putatives**	**# Iterations**
Brute Force	0.6420	74.2782	439.0642
Heuristic	0.6817	36.9931	84.3254
Visibility Prediction	0.7368	16.7587	17.6437

Table 6.1: Inliers ratio, number of putatives per frame for training and test sequences.

Finally, we show the overall localization accuracy of our monocular-vision system for the camera locations and rotations. In detail, Table 6.2 and 6.3 show the mean squared error with respect to the ground truth of the estimated localization for both translation and orientation, training and test sequences respectively. In the test sequence, the camera goes straight during approximately 1000 frames into a corridor where no occlusions are present, and after that it turns right into an area with severe occlusions. Brute force and heuristic approaches yield a wrong localization result since these two approaches are not able to estimate correctly the 180° rotation in the area with occlusions. On the contrary, with our approach, localization results are very similar to the ones obtained in the ground truth even in areas of the map with a dense level of occlusions. Besides, the number of RANSAC iterations per frame that are necessary are less than 20, showing that the method can work in real-time demands providing good localization estimates. It can be observed in Figure 6.7(b) a small gap in the camera position. This is due to the fact that in the test sequence there were some camera views that were not fully captured in the training dataset, decreasing slightly overall localization performance. With our approach

we have obtained the smallest errors with respect to the other two methods, both in translation and rotation components. Results are quite satisfactory, taking into account that appearance descriptors have not been used in the matching process.

Case	**Training** ϵ_x **(m)**	**Training** ϵ_y **(m)**	**Training** ϵ_z **(m)**	**Test** ϵ_x **(m)**	**Test** ϵ_y **(m)**	**Test** ϵ_z **(m)**
Brute Force	3.4857	0.0974	1.8305	1.1825	0.1230	1.3366
Heuristic	0.5642	0.0574	0.4142	1.1549	0.0954	0.5041
Visibility Prediction	0.0476	0.0243	0.0340	0.2781	0.0785	0.2736

Table 6.2: Localization errors in translation with respect to ground truth.

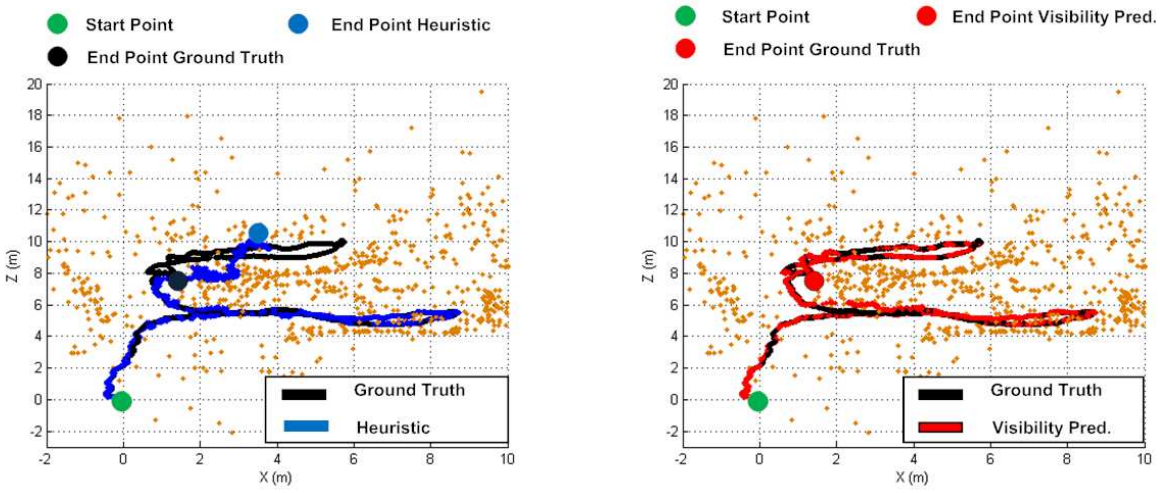
Training	ϵ_{q_0}	ϵ_{q_X}	ϵ_{q_Y}	ϵ_{q_Z}
Brute Force	0.2180	0.1030	0.3258	0.0497
Heuristic	0.2222	0.0415	0.1911	0.0264
Visibility Prediction	0.0068	0.0106	0.0100	0.0091
Test	ϵ_{q_0}	ϵ_{q_X}	ϵ_{q_Y}	ϵ_{q_Z}
Brute Force	0.2484	0.0488	0.2367	0.0346
Heuristic	0.1826	0.0452	0.1561	0.0304
Visibility Prediction	0.0516	0.0366	0.0476	0.0237

Table 6.3: Localization errors in rotation with respect to ground truth.

As it has been shown, our method depends on the quality of the input data. But how many training views are necessary to obtain accurate and fast localization results? Considering a huge number of training views can be an overwhelming computational burden for very large environments such as buildings, or even cities. We have done some experiments in which we reduce considerably the number of training views (sampling uniformly among the whole dataset of training views), and run our localization algorithm predicting features visibility. We have studied several localization runs with a different percentage of the total number of training views: 100%, 70%, 50%, 30% and 10%. In Table 6.4 ratios about the inliers ratio, number of putatives and RANSAC iterations per frame are shown for both the training and test dataset respectively.

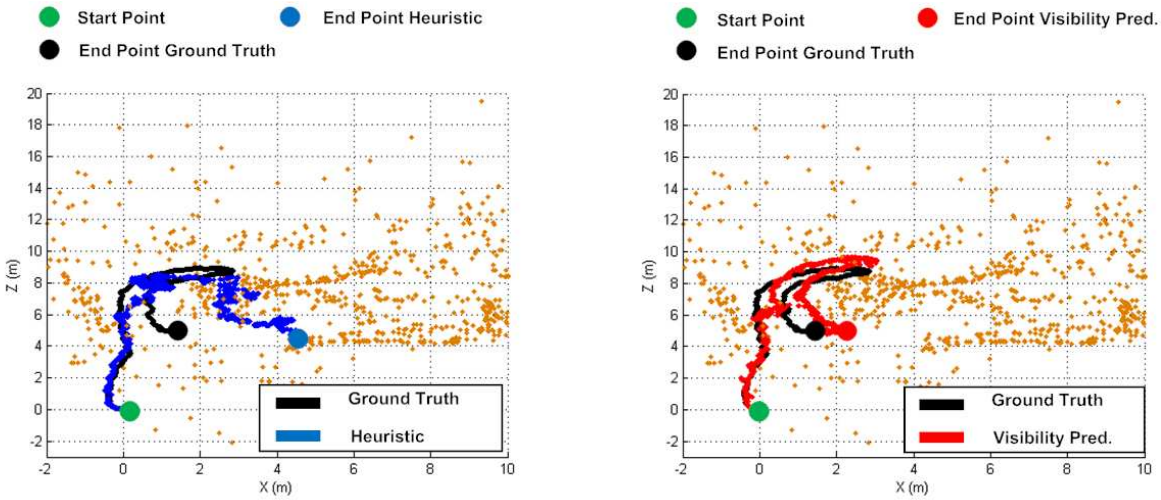
The inliers ratio is similar for all the experiments, being higher for the 100% of training views, since this is the case in which we have the highest level of detail of the 3D structure. As long as we reduce the sampling rate, keeping the same number of KNNs, the distance between the current camera pose and its nearest neighbors increases so it is easier to predict a feature to be visible when in fact it is not really visible from the current camera pose. This is the reason why the number of iterations that RANSAC needs to fit the best model increases as long as we reduce the sampling rate. In terms about the difference in localization results is very similar between all the cases, and even localization results with only a 10% of the training views are better than the brute force and heuristic approaches both for training and test sequence.

In our experiments the acquisition frame rate of the training sequence was 30 frames per second, the image resolution was 320×240 pixels and camera was carried in hand by a person at normal walking speeds ($3Km/h - 5Km/h$). According to the results it seems that our localization algorithm can provide good results with a considerably smaller



(a) (b)

Figure 6.6: Comparison of localization results for the training sequence: (a) Heuristic (b) With Visibility Prediction. Best viewed in color.



(a) (b)

Figure 6.7: Comparison of localization results for the test sequence: (a) Heuristic (b) With Visibility Prediction. Best viewed in color.

number of views than the whole dataset of training views. This is an important factor for large environments since we do not have to keep in memory the whole training dataset. It would be of benefit just selecting the views that give more information about the 3D reconstruction just in a similar spirit as it is explored in [Zhu et al., 2008].

Sequence	% Camera Poses	% Inliers	# Putatives	# Iterations	# Training Views
Training	100	0.8335	26.3897	6.4221	5319
Training	70	0.8331	30.3055	6.8429	3723
Training	50	0.8158	32.2688	8.0959	2661
Training	30	0.7803	36.2432	8.6119	1569
Training	10	0.7664	42.5013	11.2367	533
Test	100	0.7368	16.7587	17.6437	5319
Test	70	0.7194	20.0803	19.5891	3723
Test	50	0.7170	25.0831	22.5293	2661
Test	30	0.6983	26.3317	27.3240	1596
Test	10	0.6510	29.4727	30.3448	533

Table 6.4: Inliers ratio, number of putatives and RANSAC iterations per frame considering different number of training views.

6.3 Stereo Visual SLAM in Dynamic Environments

In this section, we will describe how by means of the dense scene flow information and derived residual motion likelihoods, we can detect possible moving objects in the image and avoid adding erroneous measurements into the SLAM process. For obtaining a valid dense scene flow representation, camera egomotion compensation is necessary. In our approach, we obtain the camera egomotion by means of visual odometry techniques [Nistér et al., 2004; Kaess et al., 2009]. By means of the estimated camera egomotion and the associated covariance, we can obtain a dense scene flow representation of the environment that describes the 3D motion vectors for every pixel in the current image.

Visual odometry is a key component in our visual SLAM algorithm, since we use the visual odometry information as a prior for the structure and motion in the reconstruction. Visual odometry assumes that a dominant part of the scene changes only due to camera egomotion. There can be some situations in crowded and dynamic environments where some visual odometry correspondences declared as inliers in the RANSAC step, will belong to moving objects, yielding wrong and inconsistent camera pose estimates. Those outliers can be detected if we have some prior information about the position of the moving objects in the image. Therefore, in order to use the information from the dense scene flow representation, we obtain a more robust visual odometry estimate by means of a two-step visual odometry approach:

1. First, we obtain visual odometry estimates between two consecutive images. With the resulting camera egomotion and associated uncertainty, we build a dense scene flow representation of the environment that describes the motion of 3D points. Even though the visual odometry estimate can be corrupted due to the presence of some outliers, this visual odometry estimate can be used as a motion prior for building an approximate dense scene flow representation.

2. Second, from the dense scene flow representation and derived residual motion likelihoods, we detect those possible visual odometry inliers that are located on moving

objects and discard those from the set of correspondences. Then, we re-estimate again visual odometry without the discarded set of correspondences. In this way, we can obtain more robust visual odometry estimates that will be used to create consistent priors on the 3D structure and camera motion in the SLAM process.

Fig. 6.8 depicts one comparison of visual odometry with and without moving objects detection by means of the residual motion likelihoods obtained from the dense scene flow. Even though RANSAC can detect most of the outliers or wrong correspondences, in extremelly challenging scenarios where moving objects can cover almost the whole image view there can be some remaining correspondences declared as inliers that belong to moving objects areas. By means of the dense scene flow representation, this areas can be identified and visual odometry can be re-estimated without the wrong set of correspondences, improving considerably the egomotion results.

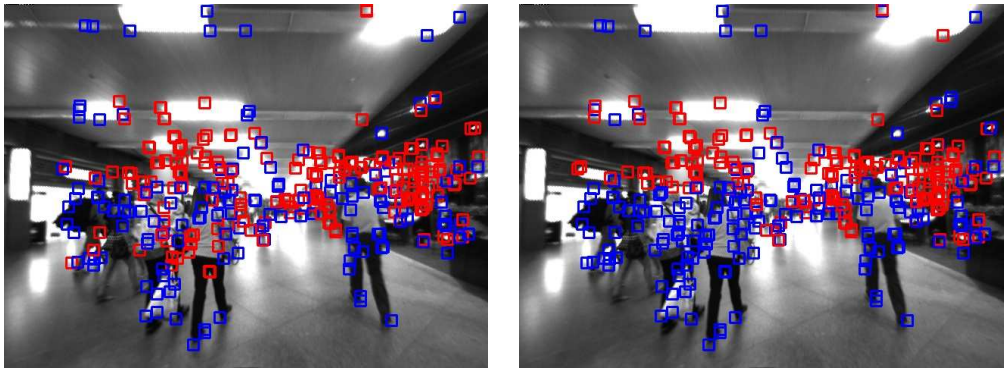
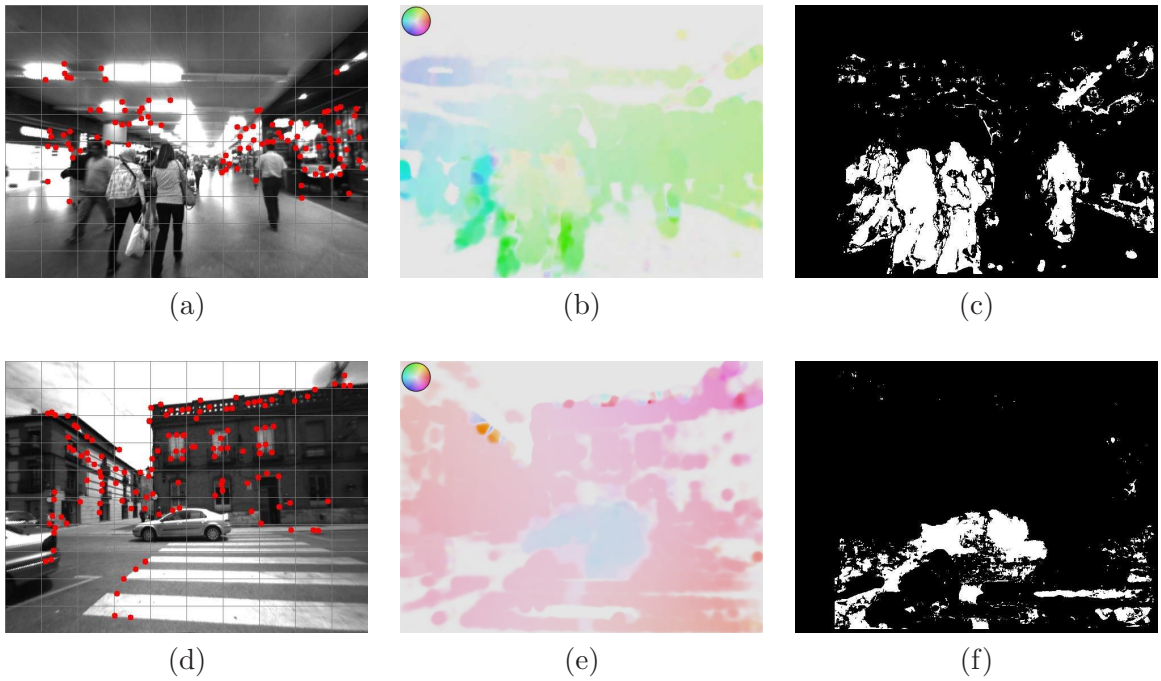


Figure 6.8: Visual odometry in the presence of moving objects. Inliers are depicted in red, whereas outliers are depicted in blue. (a) Without moving objects detection (b) With moving objects detection. Best viewed in color.

Once we have computed the residual motion likelihoods for every pixel in the current image, we can create a moving objects image mask. Those pixels that have a residual motion likelihood higher than a fixed threshold are discarded from the visual SLAM process, in order to avoid adding erroneous measurements. According to our experiments, we can only detect moving objects in a reliable way up to a distance of approximately 5 m. Detection of objects in a far range is more difficult due to the high errors introduced by the stereo reconstruction and wrong optical flow estimates due to the small size of the objects. Figure 6.9 depicts two different examples of satisfactory detection of moving objects, one from an indoor dataset and the other one from an outdoor dataset. Notice how there are no features (depicted in red) in the image since features that are located on moving objects are discarded from the SLAM process.

One of the problems of the scene flow computation is that the algorithm can detect some pixels in the image as moving objects, where in fact those pixels belong to static points due to measurement noise or optical flow problems. One of the problems of most dense optical flow methods is that they are not able to handle properly real non-artificial scenarios in textureless regions yielding constant image flow estimates over these areas. These constant image flow estimates in textureless regions do not correspond to the real observed flow. However, in visual SLAM applications this is not a big problem, since in textureless regions there are no detected 2D features at all, and even if some features are detected, these features are difficult to be tracked successfully during a large number of frames. Figure 6.10 depicts one example in which some static points located in the floor



(a) (b) (c)

(d) (e) (f)

Figure 6.9: Detection of moving objects by residual motion likelihoods. First row, experiment inside Atocha railway station. (a) Original image with the tracked SLAM features in red (b) Dense optical flow image (c) Mask of moving objects. Second row, experiments in the city of Alcalá de Henares. (d) Original image with the tracked SLAM features in red (e) Dense optical flow image (f) Mask of moving objects. For the dense optical flow images, the color encodes the direction and the saturation encodes the magnitude of the flow. For the mask of moving objects images, a pixel of white color means that the pixel belongs to a moving object. Best viewed in color.

are detected as moving points. Notice also that in these textureless areas there are no detected 2D features.

(a) (b) (c)

Figure 6.10: Problems of dense scene flow estimation in textureless areas. (a) Image with the tracked features depicted in red (b) Dense optical flow (c) Mask of moving objects. Notice that in the textureless areas there are no features of interest. Best viewed in color.

6.3.1 Visual SLAM Results in Dynamic and Crowded Environments

A portable aid for the visually impaired should consist of a small, light-weight camera and a small laptop or processor. Our vision-based system aid for the visually impaired consists of a stereo camera connected through a fireware cable to a small laptop for recording and processing the images. Figure 6.11 depicts one image of our vision-based system aid for the visually impaired.

Figure 6.11: The stereo camera system is attached to chest of the visually impaired user by means of a non-invasive orthopedic vest. Then the camera is connected to a small laptop by means of a fireware cable.

With such a non-invasive wearable vision system, the visually impaired users could potentially walk through the environment receiving audio cues that will eventually guide

them to their destination. In this way, visual SLAM approaches can serve on-line metric maps and location information to visually impaired users during navigation.

We conducted large-scale visual SLAM experiments with visually impaired users in highly dynamic environments with many independent moving objects such as pedestrians or cars. We performed experiments inside the Atocha railway station (Madrid, Spain) and in a crowded area of the city center of Alcalá de Henares (Madrid, Spain). Visually impaired users and organizations are very interested in mobility and accessibility to the environment experiments in those kind of scenarios. In these experiments, we were mainly interested in evaluating the performance of visual SLAM approaches in these kind of crowded environments, to know if visual SLAM approaches can be used successfully in future navigation applications for visually impaired users. For this purpose, the visually impaired user received several indications before the start of the sequence about going from one starting point to an end point.

For the mentioned experiments, we have used the Bumblebee2 stereo camera sold by Point Grey Research [2]. This commercial stereo rig provides highly accurate camera calibration parameters and also stereo rectification and dense depth map generation on-chip. The camera baseline is 12 cm and the horizontal field of view is of 100°. The image resolution was 640×480 pixels and the acquisition frame rate was about 15 frames per second, considering B&W images. The dimension of the G-SURF descriptors used in these experiments was 64. Now, we will show the obtained visual SLAM results for the Atocha and Alcalá experiments.

Atocha Railway Station Experiments

We performed a sequence in which a visually impaired user entered into the Atocha railway station and had to go to the entrance of the underground station, wich is located inside the railway one. Then from the entrance of the underground station, the user had to come back to the same starting place of the route in order to close the loop and correct the accummulated drift in the trajectory. Figure 6.12 depicts approximately the reference path performed by the user inside the railway station. The total length of the route (round trip) was approximately 647 m. The sequence comprises of a total number of 7,109 stereo frames and the total length in time of the experiment was 11 minutes.

Figure 6.13 depicts some image samples from the experiment inside the Atocha railway station. One of the singularities of this railway station is that it has a tropical garden inside of an approximately area of 4000 m^2 and several shops in the surroundings. Due to the presence of the tropical garden, there is a transparent roof in that area so that the sunlight can enter inside the station yielding changes in lighting conditions between the different hours of the day. The sequence is challenging not only due to the presence of many independent moving objects, but also there are changes in lighting conditions and fast camera motions that make this sequence challenging for visual SLAM applications. In addition there are some areas in the sequence where features are located in textureless regions and that most of the features are located at far depth distances, as for example shown in Figure 6.13(c). Figure 6.13(a) depicts the start of the route, Figure 6.13(b) depicts one image sample inside the station and finally Figure 6.13(c) depicts the entrance to the underground station.

Figure 6.14(a) depicts a comparison of the inliers ratio with respect to the stereo visual odometry step for the Atocha sequence. As it can be observed when we incorporate the

[2]For more information, please check: http://www.ptgrey.com/products/stereo.asp

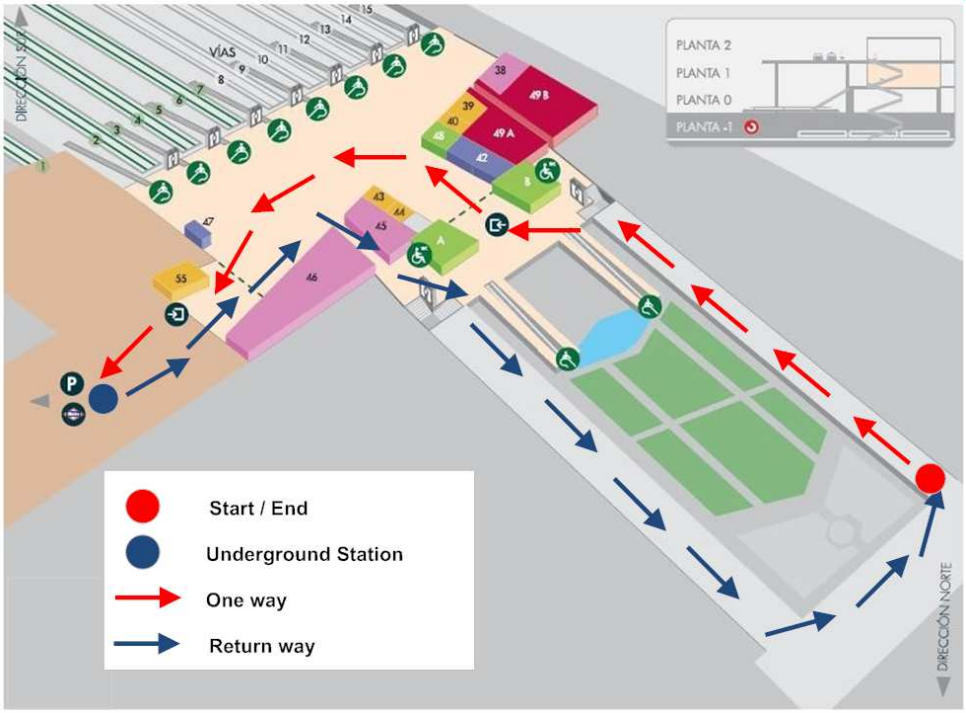
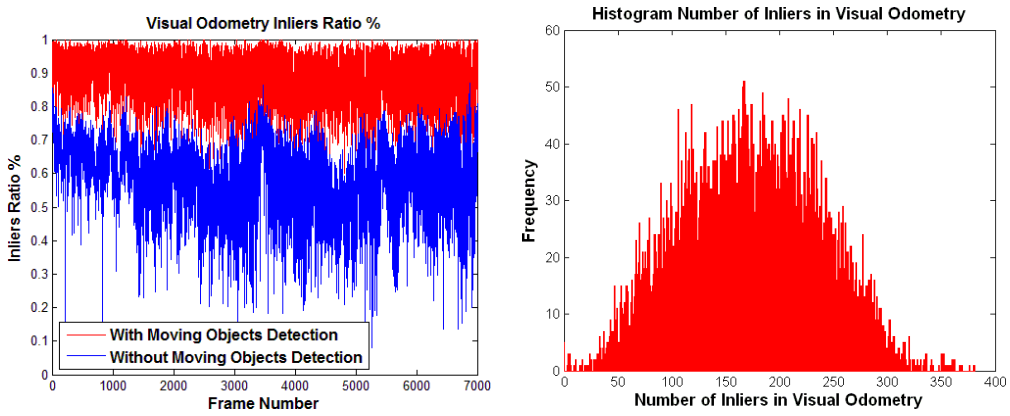


Figure 6.12: Localization experiments in Atocha railway station. Best viewed in color.

Figure 6.13: (a) Start of the route (b) One image sample inside the railway station (c) End of the route: Entrance to the underground station.

moving objects detection (MOD) module into the SLAM system, the final inliers ratio in the visual odometry estimation increases considerably. This is due to the fact that thanks to the dense scene flow representation and the derived motion likelihood values we are able to identify possible areas in the image that may belong to moving objects. With this information we can re-estimate again visual odometry without the wrong set of correspondences, yielding improved egomotion estimates. In contrast, Figure 6.14(b) depicts the histogram of the number of inliers in the visual odometry estimation. As it can be observed, there are several frames in which the number of inliers in the visual odometry estimation is below 100. Those situations correspond to images where almost the whole image view is covered by moving objects. Notice that as a default option we try to extract per frame 400 stereo features in order to find correspondences with the previous frame for the visual odometry estimation.

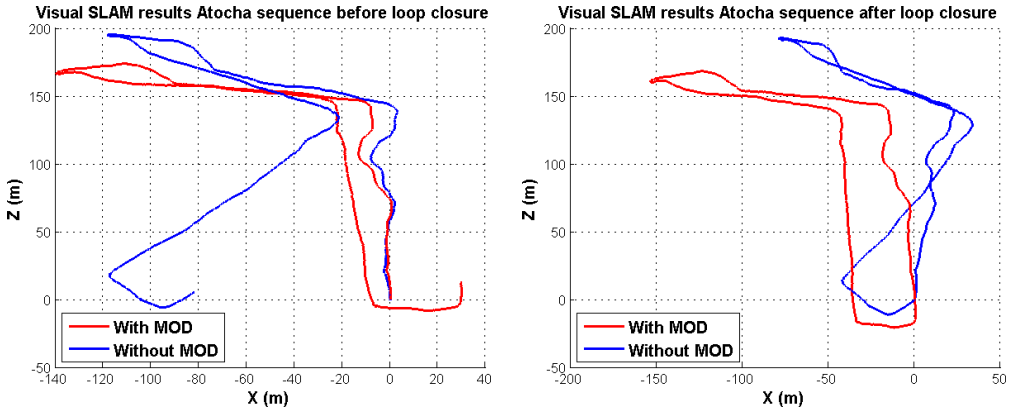


(a) (b)

Figure 6.14: Visual odometry results considering moving objects detection, Atocha sequence. (a) Comparison of the inliers ratio in visual odometry when using the moving objects detection module (b) Histogram of the number of inliers in visual odometry with moving objects detection by residual motion likelihoods. Best viewed in color.

Figure 6.15(a) depicts the trajectory performed by the visually impaired user in the Atocha sequence before the loop closure correction, considering the visual SLAM algorithm with (in red) and without (in blue) the moving objects detection module. In contrast, Figure 6.15 depicts the same comparison after the loop closure optimization by means of pose-graphs optimization techniques and a subsequent global BA optimization. It can be observed that the obtained camera trajectory considering the moving objects detection is more similar to the real camera trajectory and makes sense according to the real shape of the Atocha railway station. For example, in the corridors surrounding around the tropical garden of the station the user performed an almost straight trajectories. This is correctly estimated with the visual SLAM with moving objects detection. However, without the moving objects detection the estimated camera trajectory is completely inconsistent with the real-performed trajectory.

Estimating rotations correctly with visual SLAM in crowded environments with many independent moving objects is challenging without a detection of moving objects. In addition, good rotations estimates are critical for a good localization and mapping solution. For example, Figure 6.16 depicts three frames from the Atocha experiment, in which the user performs a rotation to enter again in the tropical garden area. We can observe in these images, that most of the useful features to estimate camera egomotion are located



(a) (b)

Figure 6.15: Comparison of visual SLAM estimated camera trajectories, Atocha sequence. (a) Before loop closure (b) After loop closure. Best viewed in color.

at far distances or in non-highly textured areas such as for example in the edges of the fluorescent lamps.

(a) (b) (c)

Figure 6.16: Different image views of the sparse 3D point cloud reconstruction from the Atocha railway station dataset. Each 3D point is depicted by they grey image value when the point was added to the map for first time.

Table 6.5 shows localization errors before the loop closure correction with and without the moving objects detection (MOD) module. We can observe that before the loop closure, the errors in the XZ plane are higher for the visual SLAM case without considering the detection of moving objects. We do not show any results in the Y (vertical) axis, since the motion was mainly performed in the XZ plane, and therefore we incorporated motion priors in the Y axis to aid the visual SLAM process. In Table 6.5, the error denotes the Euclidean distance between the final position and loop closure position the in the scene, i.e. $\sqrt{\epsilon_x^2 + \epsilon_z^2}$.

Camera Pose Element	**Final Position With MOD**	**Final Position Without MOD**
X (m)	29.6937	-82.2169
Z (m)	12.9205	5.3529
Error (m)	32.3843	82.3910

Table 6.5: Localization errors before loop closure correction in the XZ plane, Atocha sequence.

At the same time we grabbed the sequence, we employed a wheel odometer for estimating the total length in m of the trajectory. Then, we can compare the estimated total length of the wheel odometer with respect to the estimated trajectory lengths obtained in the visual SLAM experiments. Table 6.6 shows the estimated total length for the visual SLAM cases with and without moving objects detection compared to the estimated length by the wheel odometer. As we can observe, the estimated length of the visual SLAM algorithm with moving objects detection is very close to the ground truth length with a difference of about 1 m in a total trajectory of 647 m length. The error in the estimated length of the visual SLAM algorithm without moving objects detection is higher, approximately 6 m.

Estimated Length (m) Visual SLAM with MOD	**Estimated Length (m) Visual SLAM without MOD**	**Estimated Length (m) Wheel Odometer**
646.0712	641.3712	647.0000

Table 6.6: Estimated total length of the camera trajectory, Atocha sequence. Comparison of visual SLAM results with respect to a wheel odometer.

Figure 6.17 depicts some image views from different viewpoints of the sparse 3D point map from the Atocha railway station sequence using the visual SLAM algorithm with moving objects detection by means of residual motion likelihoods. The final sparse 3D reconstruction comprises of 65,584 3D map points and 2060 camera poses that correspond to the set of reconstructed keyframes.

(a) (b)

Figure 6.17: Different image views of the sparse 3D point cloud reconstruction from the Atocha railway station dataset. Each 3D point is depicted by they grey image value when the point was added to the map for first time.

Alcalá de Henares City-Center Experiments

For this scenario, we mapped an area of special cultural interest that we call the *route of Cervantes*. The trip starts at the facade of the University of Alcalá and finishes at the Cervantes house, passing through the Mayor street. This street is the most crowded one in the city of Alcalá and it is a very popular commercial area. Lining the street there are arcades supported on columns dating from the 19th century. Figure 6.18 depicts an aerial

view of the performed trajectory of the user with some distances of interest. The length of the distances was obtained by means of Google Earth. The total length of the route was approximately 447 m. The sequence comprises of a total number of 5,592 stereo frames and the total length in time of the experiment was approximately 10 minutes. From the facade of the University, the user passed Cervantes Square and walked through one of the sides of the square, which is an arcade supported on columns, and then the user rotated and headed to Mayor street going in a straight way for approximately 214 m. Then, the user crossed the street and finally reached Cervantes house.

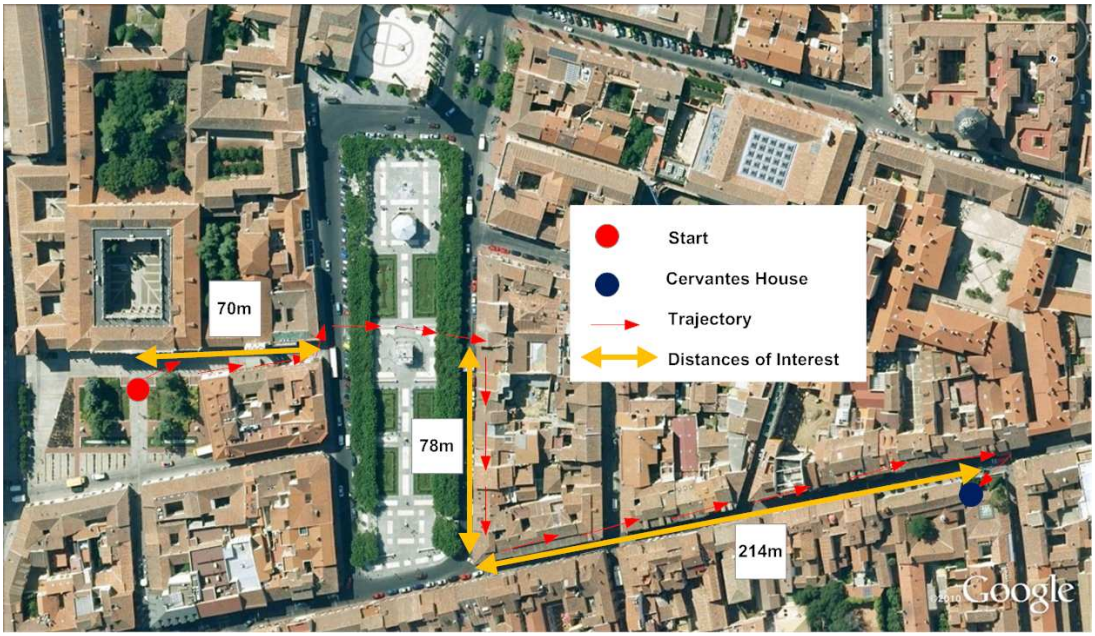
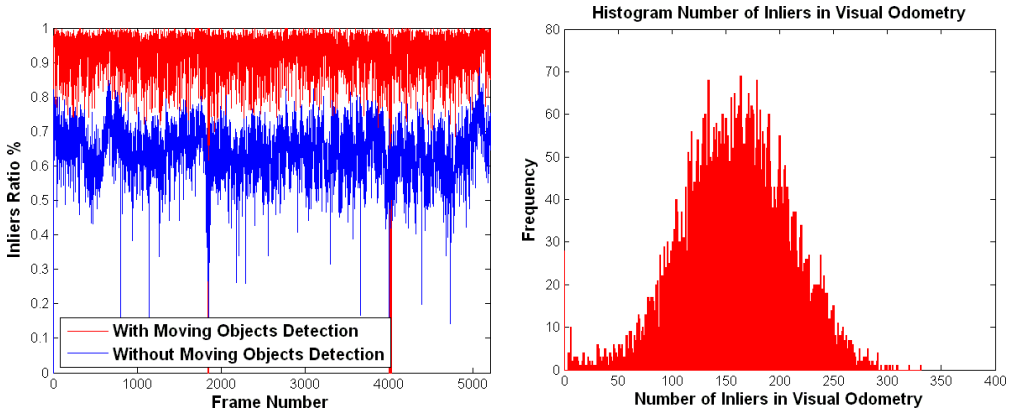


Figure 6.18: Visual SLAM experiments in the city center of Alcalá de Henares. Best viewed in color.

Figure 6.19(a) depicts the start of the route, Figure 6.19(b) depicts one image of the sequence at Mayor street and finally Figure 6.19(c) depicts Cervantes house.

Figure 6.19: (a) Start of the route: Facade of the University of Alcalá (b) Mayor street (c) End of the route: Cervantes house.

Figure 6.20(a) depicts a comparison of the inliers ratio for the stereo visual odometry algorithm in the Alcalá de Henares sequence. In general, when we incorporate the moving objects detection module into the SLAM system, the final inliers ratio in the visual odometry estimation increases considerably. However, there can be some frames where no inliers are found. In those occassions, we discard that frame from the visual odometry estimation and try to find a good solution considering the next frame.



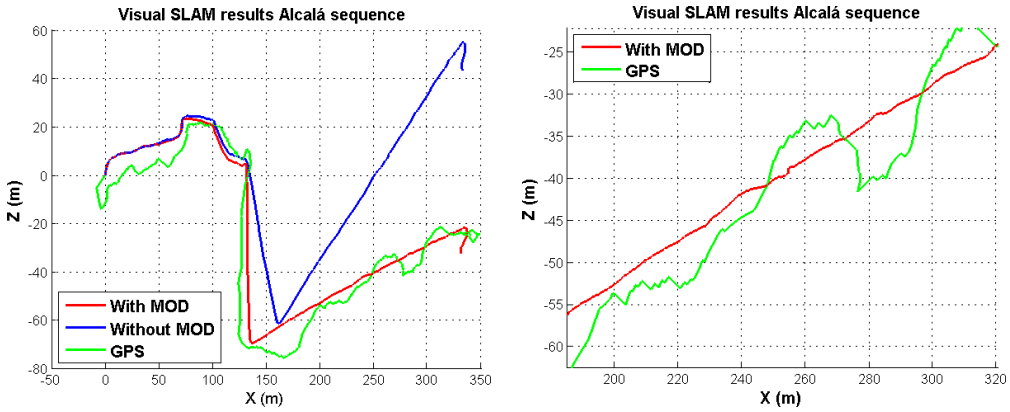
(a) (b)

Figure 6.20: Visual odometry results considering moving objects detection, Alcalá de Henares sequence. (a) Comparison of the inliers ratio in visual odometry when using the moving objects detection module (b) Histogram of the number of inliers in visual odometry with moving objects detection by residual motion likelihoods. Best viewed in color.

Figure 6.21(a) depicts the trajectory performed by the visually impaired user in the Alcalá de Henares sequence, considering the visual SLAM algorithm with (in red) and without (in blue) the moving objects detection module, and also the estimated trajectory by means of a commercial GPS (in green). As we can observe, the visual SLAM without the moving objects detection module is not able to estimate the real camera trajectory, showing higher errors when estimating the two big rotations that are present in the sequence. At the end of the sequence, we can appreciate that the errors obtained with a traditional visual SLAM algorithm are quite important. In contrast, we can appreciate how by means of the moving objects detection module, the estimated trajectory is in correspondence with the real trajectory. GPS estimated trajectory is also similar to the one obtained with our visual SLAM and moving objects detection module, however there are some places in the sequence where the standard deviation of GPS measurements is big. These situations correspond to areas where the user was walking close to buildings, or when the user was walking through the arcades in Mayor street or one of the sides of Cervantes Square. In those areas, GPS is prone to failure due to low satellite visibility conditions. Figure 6.21, depicts a zoomed area of the sequence in Mayor street, where we can appreciate with a higher level of detail the errors of GPS estimates. In ocassions, these deviations can be higher than 10 m. Those errors show why GPS solutions are not accurate and suitable enough for the navigation of visually impaired users in urban environments.

Figure 6.22 depicts a comparison between the estimated trajectories considering our visual SLAM system with the moving objects detection module with respect to GPS measurements. We superimpose the two trajectories onto an aerial image view of the sequence.

Figure 6.23 depicts three frames from the Alcalá de Henares experiment, where the visually impaired user is turning left from Cervantes Square to Mayor street. Estimating this rotation correctly is very important so as to obtain good localization results. For example, as can be observed in Figure 6.21, the visual SLAM approach without the detection of moving objects is not able to estimate this rotation properly, and then the error increases considerably after Mayor street, which is about 218 m in straight way.



(a) (b)

Figure 6.21: Comparison of visual SLAM and GPS estimated camera trajectories, Alcalá de Henares sequence. (a) Visual SLAM with and without Moving Objects Detection and GPS (b) A zoomed image of the estimated trajectory in Mayor street, where the GPS errors can be appreciated with more detail. Best viewed in color.

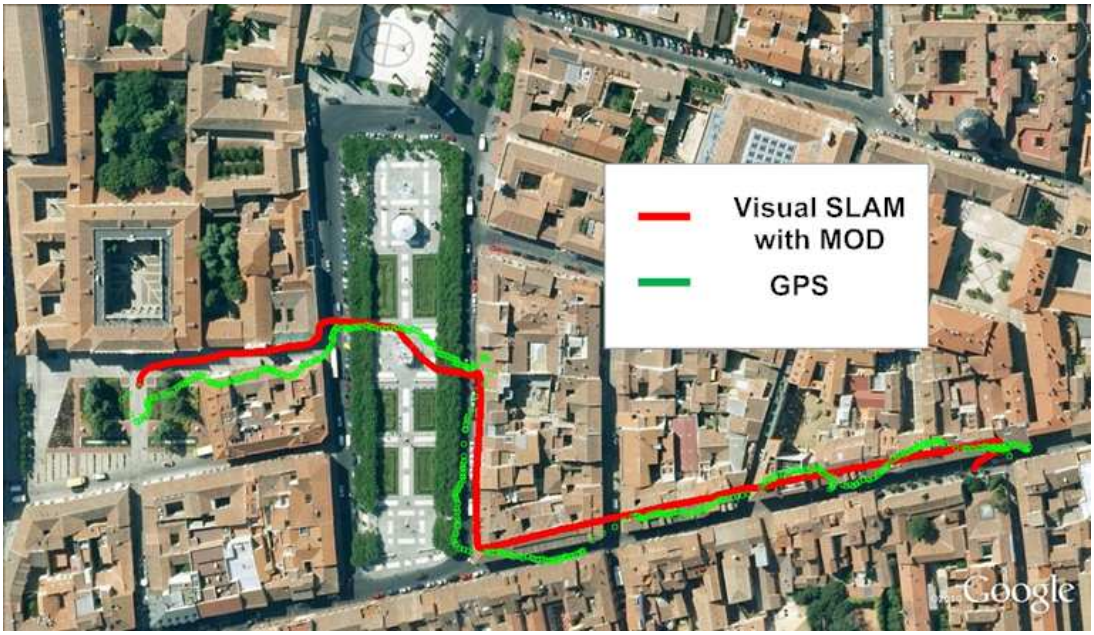


Figure 6.22: Comparison of visual SLAM with Moving Objects Detection and GPS estimated camera trajectories. Best viewed in color.

(a) (b) (c)

Figure 6.23: Different image views of the sparse 3D point cloud reconstruction from the Atocha railway station dataset. Each 3D point is depicted by they grey image value when the point was added to the map for first time.

Table 6.7 shows the estimated total length for the visual SLAM cases with and without moving objects detection compared to the estimated length by the wheel odometer. As we can observe, the estimated length of the visual SLAM algorithm with moving objects detection is again very close to the ground truth length with a difference of about 2 m in a total trajectory of 447 m length.

Estimated Length (m) Visual SLAM with MOD	**Estimated Length (m) Visual SLAM without MOD**	**Estimated Length (m) Wheel Odometer**
449.7962	451.5392	447.0000

Table 6.7: Estimated total length of the camera trajectory, Alcalá de Henares sequence. Comparison of visual SLAM results with respect to a wheel odometer.

Figure 6.24 depicts some image views from different viewpoints of the sparse 3D point map from the Alcalá de Henares city-center sequence using the visual SLAM algorithm with moving objects detection by means of residual motion likelihoods. The final sparse 3D reconstruction comprises of 64,360 3D map points and 1483 camera poses that correspond to the set of reconstructed keyframes.

(a) (b)

Figure 6.24: Different image views of the sparse 3D point cloud reconstruction from the Alcalá de Henares city-center dataset. Each 3D point is depicted by they grey image value when the point was added to the map for first time.

Timing Evaluation

As we have shown in previous sections, a dense scene flow representation of the image can yield to improved visual SLAM results considering the detection of moving objects in the

image. However, computing the dense scene flow represents a computational overhead over traditional visual SLAM methods. For example, Table 6.8 shows mean computation times of the main steps of the incremental visual SLAM algorithm for the Alcalá de Henares sequence. The most important steps in the incremental visual SLAM algorithm are feature detection and description, dense scene flow computation, visual odometry estimation and local BA. The loop closure detection, pose-graph optimization and global BA can run in parallel with the incremental visual SLAM algorithm, and some steps such as pose-graph optimization or global BA can be done at the end of the sequence or when a loop closure is detected.

Step	**Time (ms)**
Undistortion and Rectification	On-Chip
Disparity Map	On-Chip
Feature Detection (2 Scales)	16.2835
Feature Descriptors (64)	34.7035
Mean Number Descriptors per frame	400
Visual Odometry Putatives	3.0573
Visual Odometry Estimation	2.2167
Dense Scene Flow	482.3341
Mapping	14.8884
Local BA	50.9365

Table 6.8: Stereo Visual SLAM mean computation times per frame, including dense scene flow computation (Alcalá sequence). For this sequence the image resolution was 640×480.

As we can observe, the dense scene flow computation plays an important computational overhead in the incremental visual SLAM algorithm. However, we think that thanks to recent advances in dense disparity estimation [Tola et al., 2010; Geiger et al., 2010] and dense optical flow methods [Pock et al., 2007; Müller et al., 2011] and the use of powerful Graphic Processing Units (GPUs), the computational overhead of the dense scene flow representation should not be considered as a drawback. We believe that in the next future, the dense scene flow computation will become a traditional step in most of stereo visual SLAM applications, since scene flow is an essential tool to study motion in 3D scenes with many different applications.

6.4 Conclusions and Future Work

In this chapter, we have presented an algorithm for predicting the highly visible 3D points that can be used for an efficient real-time localization under indoor office-like environments. In our approach, every 3D map point models its visibility with respect to the camera poses via non-parametric distributions. We have shown the benefits of our method for monocular vision-based localization. The localization errors considering our visibility prediction are very small compared to the ground truth and also the speed-up gain compared to the other analyzed methods is very significant. In addition, we have shown that our procedure can provide accurate localization results even when a small set of training views is used. Furthermore, since our visibility prediction algorithm is based on a memory-based learning approach, the algorithm implicitly takes care of

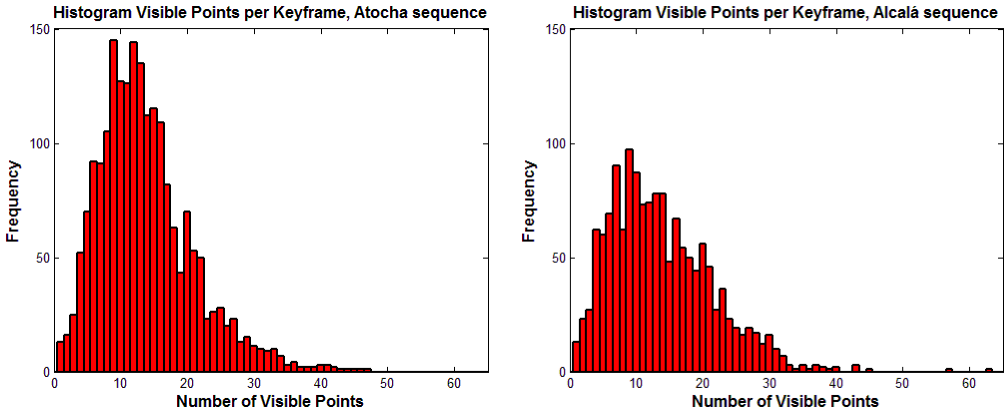
occlusions. When the visibility prediction algorithm is applied to vision-based localization applications in cluttered scenes our approach is superior compared to other heuristics that assume a transparent world, therefore introducing potential erroneous correspondences in the matching step.

Even though visual SLAM vision-based localization with a prior 3D map can be assumed to be solved in office-like environments, the problems and difficulties that can happen in real-world crowded and dynamic environments, make SLAM still an open challenge. In this chapter, we have shown that is possible to obtain accurate visual SLAM results in extemelly challenging large-scale environments with many independent moving objects. This is possible, due to the detection of moving objects in the image by means of a dense scene flow representation and from derived residual motion likelihoods. When this object detection module is added to the visual SLAM pipeline, we can improve considerably visual odometry estimates and consequently we obtain more accurate localization and mapping results in highly dynamic environments. According to our experiments and our stereo-rig setup, we are able to detect reliable moving objects up to a distance of 5 m. We think that our results can be improved considerably in the next future from better dense scene flow representations.

Dense scene flow is a challenging problem itself, since it involves the generation of dense disparity maps between the left and right images of the stereo pair and dense optical flow between two consecutive images. Most of variational optical flow methods have problems in real non-artificial scenarios in textureless regions, where a constant image flow does not correspond to the real solution. In particular the work of Müller et al. [2011] seems to be a very promising alternative to improve the capabilities of variational optical flow frameworks [Pock et al., 2007] with respect to textureless regions and large displacements. In [Müller et al., 2011], variational optical flow methods are improved for working in real scenarios by means of depth and egomotion information obtained from stereo measurements. In addition, we plan to improve scene flow estimates by means of Kalman filtering for temporal smoothness and robustness as proposed in [Rabe et al., 2010].

In this chapter, we have shown that is possible to obtain a very accurate monocular vision-based localization that can be potentially used for visually impaired users in indoor office-like environments. These environments contain many textured features and a rigid 3D model that presumably can be used for long-term localization purposes. However, in real-world environments such as cities or railway stations, obtaining an accurate monocular vision-based localization with a prior 3D map is extremelly challenging. These kind of environments contain many independent objects and also they can change considerably with time. In addition, the number of valid 3D points that are visible per keyframe in the final 3D reconstruction is very low. This is because the tracking of features in crowded environments is much more difficult and occlusions need to be taken into account. For example, Figure 6.25 depicts the histogram of visible 3D points per keyframe in the final 3D reconstructions for the Alcalá de Henares and Atocha railway station experiments. As we can observe, the number of visible 3D points per keyframe that survided the visual SLAM process and exhibit low reprojections errors after BA is very small for a great set of the keyframes. Relaxing the limits on the final reprojection error of the 3D points will not solve the problem, since pose estimates from solving the PnP problem can be erroneous if the points exhibit a high uncertainty.

According to the obtained results in Figure 6.25, we think that for large-scale crowded and dynamic environments, mantaining a robust 3D map of points that can be used later



(a) (b)

Figure 6.25: Histograms of the number of visible 3D points per keyframe for visual SLAM in crowded and dynamic environments. (a) Atocha sequence (b) Alcalá de Henares sequence.

for an efficient long-term localization is a difficult task. However, as we have shown in this chapter, visual SLAM can be used efficiently for obtaining accurate localization estimates in these difficult environments. Therefore, we think that the combination of our visual SLAM algorithm with the addition of a moving objects detection step and topological representations of the environment [Paul and Newman, 2010; Ranganathan and Dellaert, 2010] can yield to improved and robust long-term navigation of visually impaired users in real-world environments.

In the next future, we are interested in performing large-scale visual SLAM experiments in indoor and outdoor environments for aiding visually impaired users during navigation. We believe that the methods developed in this dissertation can constitute a good starting point towards the development of commercial aids for this community. However, there are still some open issues, mostly related to real-world highly dynamic environments such as outdoor urban-scenes or crowded indoor environments, e.g. railway stations. Even though, most of modern SfM or SLAM systems are capable of obtaining an accurate 3D representation of the environment, most of the approaches do not scale well with time and ignore the fact that environments such as cities or places change drastically over time. In typical SfM approaches, each 3D point in the map is normally characterized by an invariant to scale and/or rotation appearance descriptor vector [Lowe, 2004; Bay et al., 2008]. However, these appearance descriptors may change considerably between different hours of the day, or between different months of the year. Furthermore, cities change over time, and it is possible that some buildings previously mapped will not exist in the future. Therefore, we think that future lines of research can go in the direction of improving SfM algorithms to scale properly with time variations in order to obtain a long-term localization and navigation. In this context, a probabilistic temporal inference [Schindler and Dellaert, 2010; Xu et al., 2010] of the incremental 3D reconstruction can be proposed to deal with changes in time.

Chapter 7

Conclusions and Future Work

The starting point of this thesis is the development of robust and efficient visual Simultaneous Localization and Mapping (SLAM) and vision-based localization algorithms that can be applied with success to different kind of applications such as: humanoid robotics and visually impaired people. Visual SLAM and vision-based localization are challenging related problems since they involve many different tasks such as: the computation of an accurate 3D map of the environment, the description of each 3D map point by means of an appearance descriptor, the data association between a large map of 3D points and perceived 2D features in the current camera view and finally solving a non-least squares problem to estimate the spatial location and orientation of the camera in the environment.

We have developed a stereo visual SLAM algorithm that uses efficient stereo visual odometry techniques for providing good initialization estimates of the motion and the structure. In order to cope with large-scale environments we have applied a local Bundle Adjusment (BA) procedure in which a subset of the camera poses and the 3D map points are optimized simultaneously. By means of loop closure detection, we can correct the accummulated drift in the incremental 3D reconstruction using pose-graph optimization techniques [Dellaert and Kaess, 2006]. Finally, the whole set of 3D map points and camera poses can be further optimised in a global BA step.

Most of traditional visual SLAM and Structure from Motion (SfM) methods have not been evaluated in detail to work in realistic scenarios, where we can have very crowded scenes with many independent moving objects. Visual SLAM algorithms usually assume static features in the environment and that a dominant part of the scene changes only due to camera egomotion. Therefore, those algorithms are prone to failure in extremely challenging scenarios where we can have many moving objects that on occasions can almost cover the entire image view. In this thesis we used all the capabilities of stereo vision, exploiting the information of four images at once, obtaining dense disparity maps (between the left and right stereo views at each frame) and dense 2D optical fow (between two consecutive frames for a reference image of the stereo pair). Since for every pixel that has a valid disparity value we know its 3D position (with respect to the camera coordinate frame) and the associated dense 2D optical flow (with respect to a consecutive image), a dense scene flow [Vedula et al., 1999] description of the scene can be obtained, describing the 3D motion of the world points. By means of a dense scene flow representation and egomotion compensation, we have shown how to obtain motion likelihoods that can be used to detect possible moving objects in the image. According to our stereo rig settings and the small baseline used in our experiments 12 cm, we can only detect moving objects reliable up to a distance of 5 m. This moving objects detection module was incorporated into the visual SLAM system in order to aid the visual odometry estimation and to avoid

adding erroneous 3D points into the SLAM process, yielding better 3D reconstruction results.

Image matching is the problem of establishing correspondence between two images and is a core component of all sorts of computer vision systems, particularly in classic problems such as SfM [Agarwal et al., 2009], visual categorization [Csurka et al., 2004] or vision-based localization [Liu et al., 2010]. In this thesis, we have presented a new family of multiscale local descriptors named Gauge-Speeded Up Robust Features descriptors (G-SURF), a novel high performance SURF-inspired [Bay et al., 2008] set of descriptors based on gauge coordinates which are easy to implement but are theoretically and intuitively highly appealing. Image matching quality is considerably improved relative to standard SURF and other state-of-the-art techniques, especially for those scenarios where the image transformation is small in terms of change in viewpoint or the image transformation is related to blur, rotation, changes in lighting, JPEG compression or random Gaussian noise. In addition, another important conclusion is that descriptors based on gauge-derivatives can exhibit much higher performance than first-order local derivatives based descriptors. This is possible, due to the extra invariance offered by gauge-derivatives and also our G-SURF descriptors have comparable computational cost with respect to other approaches. Hence, our family of descriptors can be used efficiently in visual SLAM and vision-based localization applications. The set of G-SURF descriptors evaluated in this thesis will be publicly available in an open source library named *OpenGSURF*.

One of the most computationally expensive steps in vision-based localization is data association, in which matching candidates between a large map of 3D points and 2D features are retrieved and then usually validated by geometric constraints using RANdom SAmple Consensus (RANSAC) [Fischler and Bolles, 1981]. For the environments with highly repetitive textures, such as cities, the traditional methods mainly depend on the appearance information, which results in a very large number of matching candidates due to the ambiguities introduced by visually similar features [Schindler et al., 2008]. In this thesis, we proposed a novel method for predicting the visibility of 3D points in large-scale urban environments. In our approach, every map point models its visibility with respect to the camera poses via non-parametric distributions. By means of a combination of the Mahalanobis distance [Mahalanobis, 1936] and a Gaussian kernel, we showed how to learn a similarity metric between two camera poses in an entirely non-parametric way that is invariant to the scale of the input vectors. We have discussed the importance of the cues used in our metric and shown the benefits of adding local image histograms instead of using only camera translation and viewing direction. For very large 3D reconstructions, the number of map elements is more than a hundred times higher than the number of camera poses. Based on this observation, we think that for fast and robust vision-based localization over large 3D reconstructions, our algorithm can dramatically reduce the computation time and improve robustness, contrary to other common approaches that check the visibility of each of the 3D points from the dataset individually or pruning-out information based in geo-spatial constraints.

The visibility prediction algorithm ellaborated in this thesis has been applied successfully to two different vision-based localization applications: humanoid robots and visually impaired people. Regarding humanoid robots, we have presented a monocular vision-based localization algorithm that works in real-time (even faster than 30 Hz) and provides localization accuracy about the order of cm. For this purpose, we first build a 3D map of the environment by using stereo visual SLAM techniques, and perform visibility learning over the prior 3D reconstruction. Then, for fast vision-based localization we use

visibility prediction techniques for solving efficiently the Perspective-n-Point (PnP) problem [Lu et al., 2000; Ansar and Danilidis, 2003] and obtaining the location of the robot with respect to a global coordinate frame. We measured the accuracy of our localization algorithm by comparing the estimated trajectory of the robot with respect to ground truth data obtained by a highly accurate motion capture system. We also compared our algorithm with respect to a well-known state-of-the-art monocular visual SLAM algorithm such as the Parallel Tracking and Mapping (PTAM) [Klein and Murray, 2007] approach, showing the clear benefits of our proposed localization system.

With respect to the vision-based localization approach for the visually impaired, we have developed a vision-based localization algorithm with visibility prediction using a monocular camera as the only sensor and a prior 3D map of the environment for small office-like environments. The obtained results show that is possible to obtain a robust localization in these kind of cluttered environments with errors of few cm, even without using any appearance descriptor in the matching process, just exploiting the *smart* use of the geometric information from the prior 3D reconstruction. Our visibility prediction algorithm can deal with occlusions since the algorithm is based on a memory based learning approach and therefore we are able to *learn* occlusions. When the visibility prediction algorithm is applied to vision-based localization algorithms in cluttered scenes our approach is superior compared to other heuristics that assume a transparent world, therefore introducing potential erroneous correspondences in the matching step. In addition, we also evaluated the visual SLAM algorithm with moving objects detection in challenging crowded scenarios with many independent moving objects such as inside the Atocha railway station (Madrid, Spain) and the city-center of Alcalá de Henares (Madrid, Spain). Our results show that adding the moving objects detection module into the SLAM process yields better visual odometry and consequently more accurate localization and mapping estimates in highly crowded and dynamic environments with many independent moving objects. The obtained results show that is possible to obtain a robust localization in these kind of large-scale environments with errors of few m, improving considerably the results of GPS-based systems for outdoor environments.

7.1 Main contributions

From the results obtained in previous chapters, we consider that the main contributions of this thesis are the following:

- **Visual SLAM with moving objects detection**. In this thesis we have developed a stereo visual SLAM algorithm that can be used for mapping different types of environments, from small ones to large-scale environments by means of a hierarchical BA procedure. In addition, we have shown how to obtain a dense scene flow representation of the environment with egomotion compensation. We have also shown how to derive residual motion likelihoods from the dense scene flow representation. By means of the residual motion likelihoods we can identify possible moving objects in the image. The detection of moving objects is used to aid the SLAM process. The obtained SLAM results with the detection of moving objects are much better compared to the case where no moving objects detection is carried out. Up to the best of our knowledge, this is the first time dense scene flow has been proposed in the context of visual SLAM for aiding the SLAM estimation.

- **G-SURF descriptors**. We have introduced a new family of local descriptors that

are based on second-order multiscale gauge derivatives. According to our extensive evaluation, G-SURF descriptors exhibit much better performance than other state-of-the-art methods while exhibiting similar computational demands. G-SURF descriptors can be used efficiently for real-time SfM applications such as visual SLAM and vision-based localization. In addition, the library *OpenGSURF* that contains the set of descriptors evaluated in this thesis, will be made publicly available in the next future. Up to the best of our knowledge, this is the first open source library that allows the user to choose between different dimensional descriptors.

- **Visibility learning in large scale urban-environment**. We have proposed a memory based classification algorithm that exploits all the geometric relationships between a prior 3D map and the set of camera poses involved in the reconstruction, for predicting the visibility of the 3D points given a query camera pose. In our approach, we model the visibility of every map 3D point with respect to a query camera pose using a non-parametric distribution model. We learn these non-parametric distributions during the 3D reconstruction process, and develop efficient algorithms to predict the visibility of the 3D points during localization. With this approach, the matching process only uses those map elements with the highest visibility score, yielding a much faster algorithm and superior localization results.

- **Visual SLAM and vision-based localization for humanoid robots**. Our stereo visual SLAM algorithm was tested in typical humanoid robotics applications and obtained an accuracy about the order of cm. This accuracy was measured with respect to a highly accurate ground truth data obtained by means of a motion capture system. The presented monocular vision-based localization algorithm with visibility prediction is able to obtain an accuracy about the order of few cm and runs even faster than real-time demands (30 Hz) in laboratory-like environments. Furthermore, the localization algorithm is robust to the presence of people in the environment, different types of trajectories and changes in lighting conditions. In addition, we compared our approach with respect to well known PTAM approach, showing the clear benefits of our algorithm for humanoid robotics applications.

- **Visual SLAM and vision-based localization for the visually impaired**. In this thesis we have shown that is possible to obtain a robust vision-based localization system for aiding visually impaired users during navigation for office-like indoor environments. We have evaluated our visual SLAM algorithm with moving objects detection in very challenging scenarios with many independent moving objects and considering real visually impaired users. Our results highlight the potential benefits of the proposed techniques for the visually impaired community in the next future.

7.2 Future work

From the results and conclusions of the present work, several lines of work can be proposed:

- **Probabilitic temporal inference**. Even though, most of modern SfM or visual SLAM systems are capable of obtaining an accurate 3D representation of the environment, most of the approaches do not scale well with time and ignore the fact that environments such as cities or dynamic indoor environments change drastically over time. In typical SfM approaches, each 3D point in the map is normally characterized by an invariant to scale and/or rotation appearance descriptor vector [Lowe, 2004;

Bay et al., 2008]. However, these appearance descriptors may change considerably between different hours of the day, or between different months of the year. Furthermore, cities change over time, and it is possible that some buildings previously mapped will not exist in the future. Therefore, we think that future lines of research can go in the direction of improving SfM algorithms to scale properly with time variations. In this context, a probabilistic temporal inference [Schindler and Dellaert, 2010; Xu et al., 2010] of the incremental 3D reconstruction can be proposed to deal with changes in time.

- **Total 3D scene understanding**. We are also interested in improving the capabilities of visual SLAM and SfM approaches in order to deal with moving objects in the scene. We think that the combination of a robust dense scene flow representation plus the use of well-understood pedestrian detectors [Nedevschi et al., 2009; Enzweiler and Gavrila, 2009] and the tracking of the moving objects [Ess et al., 2009] can yield a very robust visual SLAM method that can be used in extremely challenging and crowded environments. In addition, we think that a more detailed 3D scene understanding [Geiger et al., 2011a] can be of benefit for visual SLAM and vision-based localization approaches.

- **Improvements in the computation of dense scene flow representations**. Dense scene flow is a challenging problem itself, since it involves the generation of dense disparity maps between the left and right images of the stereo pair and dense 2D optical flow between two consecutive images. Most of variational optical flow methods have problems in real non-artificial scenarios with large image displacements and in textureless regions. In those regions, the estimated optical flow is usually a constant flow that does not correspond to the real solution. In particular the work of Müller et al. [2011] seems to be a very promising alternative to improve the capabilities of variational optical flow frameworks [Pock et al., 2007] with respect to textureless regions and large displacements. In [Müller et al., 2011], variational optical flow methods are improved for working in real scenarios by means of depth and egomotion information obtained from stereo measurements. In addition, scene flow estimates can be improved by means of Kalman filtering for temporal smoothness and robustness as proposed in [Rabe et al., 2010].

- **Feature detection and description in non-linear scale spaces**. Another interesting line of research is the detection and description of invariant features considering non-linear diffusion scale spaces [Perona and Malik, 1990; Weickert et al., 1998]. In the standard scale-space paradigm the true location of a boundary at a coarse scale is not directly available in the coarse scale image. The reason for this is simply because Gaussian blurring does not respect the natural boundaries of objects. We believe that introducing new invariant features that fully exploit non-linear diffusion scale spaces (both in detection and local description of features) can represent step forward improvements on traditional image matching and object recognition applications.

- **Improving the capabilities of humanoid robotics applications**. Since we have obtained a pretty accurate, robust and real-time localization of the robot in a prior 3D map, we think that this accurate localization can be used satisfactory in related humanoid robotics applications such as control [Blösch et al., 2010], autonomous 3D object modeling [Foissote et al., 2010] or footstep planning [Perrin et al., 2010]. We

think that our real-time vision based localization can improve considerably some previous humanoid robotics applications where vision-based localization was not exploited in all its capabilities.

- **Navigation assistance of visually impaired users by means of vision-based localization**. We are interested in using our visual SLAM and vision-based localization framework for providing a complete navigation assistance system for the visually impaired. Obstacle detection and avoidance seems to be of critical importance for visually impaired users, and these algorithms can be integrated easily within our localization framework. In addition, we think that is interesting to investigate different sensorial modalities for transmitting the 3D perception information of the environment to the users, such as audio or tactile cues. In addition, we think that the combination of our metric visual SLAM with robust topological representations of the environment [Paul and Newman, 2010; Ranganathan and Dellaert, 2010] can yield to improved and robust long-term navigation of visually impaired users in real-world environments.

Bibliography

Agarwal, S., Snavely, N., Seitz, S. M., and Szeliski, R. (2010). Bundle Adjustment in the Large. In *Eur. Conf. on Computer Vision (ECCV)*.

Agarwal, S., Snavely, N., Simon, I., Seitz, S. M., and Szeliski, R. (2009). Building Rome in a Day. In *Intl. Conf. on Computer Vision (ICCV)*.

Agrawal, M., Konolige, K., and Blas, M. R. (2008). CenSurE: Center Surround Extremas for realtime feature detection and matching. In *Eur. Conf. on Computer Vision (ECCV)*.

Alcantarilla, P., Bergasa, L., and Dellaert, F. (2010a). Visual odometry priors for robust EKF-SLAM. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3501–3506, Anchorage, AK, USA.

Alcantarilla, P., Ni, K., Bergasa, L., and Dellaert, F. (2011). Visibility learning in large-scale urban environment. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Shanghai, China.

Alcantarilla, P., Oh, S., Mariottini, G., Bergasa, L., and Dellaert, F. (2010b). Learning visibility of landmarks for vision-based localization. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 4881–4888, Anchorage, AK, USA.

Angeli, A., Filliat, D., Doncieux, S., and Meyer, J. A. (2008). Fast and Incremental Method for Loop-Closure Detection using Bags of Visual Words. *IEEE Trans. Robotics*, 24:1027–1037.

Ansar, A. and Danilidis, K. (2003). Linear pose estimation from points or lines. *IEEE Trans. Pattern Anal. Machine Intell.*, 25(4):1–12.

Atkeson, C., Moore, A., and Schaal, S. (1997). Locally weighted learning. *AI Review*, 11:11–73.

Bard, Y. (1974). *Nonlinear Parameter Estimation*. Academic Press.

Bay, H., Ess, A., Tuytelaars, T., and Gool, L. V. (2008). SURF: Speeded up robust features. *Computer Vision and Image Understanding*, 110(3):346–359.

Bay, H., Fasel, B., and Gool, L. V. (2006a). Interactive Museum Guide: Fast and Robust Recognition of Museum Objects. In *Proceedings of the first international workshop on mobile vision*.

Bay, H., Tuytelaars, T., and Gool, L. V. (2006b). SURF: Speeded up robust features. In *Eur. Conf. on Computer Vision (ECCV)*.

Berg, A. C. and Malik, J. (2001). Geometric blur for template matching. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 607–614, Hawaii, USA.

Bibby, C. and Reid, I. (2007). Simultaneous localisation and mapping in dynamic environments (SLAMIDE) with reversible data association. In *Robotics: Science and Systems (RSS)*.

Bishop, C. (2007). *Pattern Recognition and Machine Learning*. Springer.

Blösch, M., Weiss, S., Scaramuzza, D., and Siegwart, R. (2010). Vision based MAV navigation in unknown and unstructured environments. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Anchorage, AK, USA.

Bohg, J., Holst, C., Huebner, K., Ralph, M., Rasolzadeh, B., Song, D., and Kragic, D. (2009). Towards grasp-oriented visual perception for humanoid robots. *Intl. J. of Humanoid Robotics*, 6(3):387–434.

Bolles, R. and Fischler, M. (1981). A RANSAC-based approach to model fitting and its application to finding cylinders in range data. In *Intl. Joint Conf. on AI (IJCAI)*, pages 637–643, Vancouver, Canada.

Bouguet, J. (2008a). The calibration toolbox for Matlab, example 5: Stereo rectification algorithm (code and instructions only).

Bouguet, J. (2008b). Documentation: Camera Calibration Toolbox for Matlab.

Broida, T., Chandrashekhar, S., and Chellappa, R. (1990). Recursive 3-D motion estimation from a monocular image sequence. *IEEE Trans. Aerosp. Electron. Syst.*, 26(4):639–656.

Broida, T. and Chellappa, R. (1991). Estimating the kinematics and structure of a rigid object from a sequence of monocular images. *IEEE Trans. Pattern Anal. Machine Intell.*, 13(6):497–513.

Brown, M., Gang, H., and Winder, S. (2011). Discriminative learning of local image descriptors. *IEEE Trans. Pattern Anal. Machine Intell.*, 33(1):43–57.

Byröd, M. and Åström, K. (2010). Conjugate gradient bundle adjustment. In *Eur. Conf. on Computer Vision (ECCV)*.

Calonder, M., Lepetit, V., and Fua, P. (2010). BRIEF: Binary Robust Independent Elementary Features. In *Eur. Conf. on Computer Vision (ECCV)*.

Caselles, V., Morel, J.-M., and Sbert, C. (1998). An axiomatic approach to image interpolation. *IEEE Trans. on Image Processing*.

Chekhlov, D., Gee, A., Calway, A., and Mayol-Cuevas, W. (2007). Ninja on a Plane: Automatic Discovery of Physical Planes for Augmented Reality Using Visual SLAM. In *IEEE and ACM Intl. Sym. on Mixed and Augmented Reality (ISMAR)*.

Civera, J., Davison, A. J., and Montiel, J. M. (2008). Inverse depth parametrization for monocular SLAM. *IEEE Trans. Robotics*.

Clemente, L. A., Davison, A. J., Reid, I., Neira, J., and Tardós, J. D. (2007). Mapping large loops with a single hand-held camera. In *Robotics: Science and Systems (RSS)*.

Clipp, B., Lim, J., Frahm, J.-M., and Pollefeys, M. (2010). Parallel, real-time visual SLAM. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan.

Csurka, G., Bray, C., Dance, C., and Fan, L. (2004). Visual categorization with bags of keypoints. In *In Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22.

Cummins, M. and Newman, P. (2008). FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *Intl. J. of Robotics Research*, 27(6):647–665.

Dalal, N. and Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Damon, J. (1995). Local Morse theory for solutions to the heat equation and Gaussian blurring. *Journal of Differential Equations*, 115(2):368–401.

Davison, A. J. and Murray, D. W. (2002). Simultaneous localization and map-building using active vision. *IEEE Trans. Pattern Anal. Machine Intell.*, 24(7).

Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Machine Intell.*, 29(6).

Dellaert, F. and Kaess, M. (2006). Square Root SAM: Simultaneous localization and mapping via square root information smoothing. *Intl. J. of Robotics Research*, 25(12):1181–1203.

Devernay, F., Mateus, D., and Guilbert, M. (2006). Multi-camera scene flow by tracking 3-D points and surfels. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Dominey, P., Mallet, A., and Yoshida, E. (2007). Progress in programming the HRP-2 humanoid using spoken language. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Rome, Italy.

Dreuw, P., Steingrube, P., Hanselmann, H., and Ney, H. (2009). SURF-Face: Face Recognition under Viewpoint Consistency Constraints. In *British Machine Vision Conf. (BMVC)*.

Durrant-White, H. and Bailey, T. (2006). Simultaneous localization and mapping SLAM: part 1. *IEEE Robotics and Automation Magazine*, 13(3):99–110.

Eade, E. and Drummond, T. (2007). Monocular SLAM as a graph of coalesced observations. In *Intl. Conf. on Computer Vision (ICCV)*.

Enzweiler, M. and Gavrila, D. M. (2009). Monocular pedestrian detection: Survey and experiments. *IEEE Trans. Pattern Anal. Machine Intell.*, 31(12):2179–2195.

Ess, A., Leibe, B., Schindler, K., and Gool, L. V. (2009). Robust multi-person tracking from a mobile platform. *IEEE Trans. Pattern Anal. Machine Intell.*, 31(1):1831–1846.

Evans, C. (2009). Notes on the OpenSURF library. Technical Report CSTR-09-001, University of Bristol.

Faber, F., Bennewitz, M., Eppner, C., Goeroeg, A., Gonsior, A., Joho, D., Schreiber, M., and Behnke, S. (2009). The humanoid museum tour guide Robotinho. In *IEEE Intl. Symposium on Robot and Human Interactive Communication (RO-MAN)*, Toyama, Japan.

Farnebäck, G. (2003). Two-frame motion estimation based on polynomial expansion. In *Proceedings of the 13th Scandinavian Conference on Image Analysis*, pages 363–370, Gothenburg, Sweden.

Feiner, S., MacIntyre, B., Höllerer, T., and Webster, A. (1997). A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. In *Intl. Symp. on Wearable Computing (ISWC)*, pages 74–81.

Fergus, R., Perona, P., and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 264–271.

Fischler, M. and Bolles, R. (1981). Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. Assoc. Comp. Mach.*, 24:381–395.

Florack, L. M. J., ter Haar Romeny, B. M., Koenderink, J. J., and Viergever, M. A. (1993). Cartesian differential invariants in scale-space. *Journal of Mathematical Imaging and Vision*, 3:327–348.

Foissote, T., Stasse, O., Wieber, P., Escande, A., and Kheddar, A. (2010). Autonomous 3D object modeling by a humanoid using an optimization-driven next-best-view formulation. *Intl. J. of Humanoid Robotics*.

Freeman, W. and Adelson, E. (1991). The design and use of steerable filters. *IEEE Trans. Pattern Anal. Machine Intell.*, 13(9):891–906.

Frese, U., Larsson, P., and Duckett, T. (2008). A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Trans. Robotics*, 21(2):196–207.

Furukawa, Y., Curless, B., Seitz, S., and Szeliski, R. (2010). Towards Internet-scale multi-view stereo. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Gee, A., Checkhlov, D., Calway, A., and W.Mayol-Cuevas (2008). Discovering higher level structure in visual SLAM. *IEEE Trans. Robotics*, 24(5):980–990.

Geiger, A., Lauer, M., and Urtasun, R. (2011a). A generative model for 3d urban scene understanding from movable platforms. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, USA.

Geiger, A., Roser, M., and Urtasun, R. (2010). Efficient large-scale stereo matching. In *Asian Conf. on Computer Vision (ACCV)*, Queenstown, New Zealand.

Geiger, A., Ziegler, J., and Stiller, C. (2011b). Stereoscan: Dense 3d reconstruction in real-time. In *IEEE Intelligent Vehicles Symposium (IV)*, Baden-Baden, Germany.

Gil, A., Mozos, O., Ballesta, M., and Reinoso, O. (2009). A comparative evaluation of interest point detectors and local descriptors for visual SLAM. *Machine Vision and Applications*.

Gill, P., Murray, W., and Wright, M. (1981). *Practical Optimization*. Academic Press.

Goesele, M., Curless, B., and Seitz, S. (2006). Multi-view stereo revisited. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2402–2409, New York, USA.

Goesele, M., Snavely, N., Curless, B., Hoppe, H., and Seitz, S. (2007). Multi-view stereo for community photo collections. In *Intl. Conf. on Computer Vision (ICCV)*, pages 14–20, Rio de Janeiro, Brasil.

Gool, L. V., Moons, T., and Ungureanu, D. (1996). Affine/photometric invariants for planar intensity patterns. In *Eur. Conf. on Computer Vision (ECCV)*, pages 642–651.

Grasa, O. G., Civera, J., and Montiel, J. (2011). EKF Monocular SLAM with Relocalization for Laparoscopic Sequences. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*.

Grisetti, G., Kümmerle, R., Stachniss, C., Frese, U., and Hertzberg, C. (2010). Hierarchical optimization on manifolds for online 2D and 3D mapping. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*.

Hähnel, D., Triebel, R., Burgard, W., and Thrun, S. (2003). Map building with mobile robots in dynamic environments. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 1557–1563, Taipei, Taiwan.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Proc. Fourth Alvey Vision Conference*, pages 147–151.

Hartley, R. (1999). Theory and practice of projective rectification. *International Journal of Computer Vision*, 35:115–127.

Hartley, R. and Zisserman, A. (2000). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

Heikkila, J. and Silven, O. (1997). A four-step camera calibration procedure with implicit image correction. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1106–1112.

Hesch, J. A. and Roumeliotis, S. I. (2010). Design and analysis of a portable indoor localization aid for the visually impaired. *Intl. J. of Robotics Research*, 29(11):1400–1415.

Hirschmuller, H. (2006). Stereo vision in structured environments by consistent semi-global matching. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2386–2393.

Hornung, A., Wurm, K., and Bennewitz, M. (2010). Humanoid robot localization in complex indoor environments. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan.

Hua, G., Brown, M., and Winder, S. (2007). Discriminant embedding for local image descriptors. In *Intl. Conf. on Computer Vision (ICCV)*, Rio de Janeiro, Brazil.

Kaess, M., Ni, K., and Dellaert, F. (2009). Flow separation for fast and robust stereo odometry. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Kobe, Japan.

Kaess, M., Ranganathan, A., and Dellaert, F. (2008). iSAM: Incremental Smoothing and Mapping. *IEEE Trans. Robotics*.

Kagami, S., Nishiwaki, K., Kuffner, J., Thompson, S., Chestnutt, J., Stilman, M., and Michel, P. (2005). Humanoid HRP2-DHRC for autonomous and interactive behavior. In *Proc. of the Intl. Symp. of Robotics Research (ISRR)*, pages 103–117.

Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., Akachi, K., and Isozumi, T. (2004). Humanoid robot HRP-2. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 1083–1090, New Orleans, USA.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *IEEE and ACM Intl. Sym. on Mixed and Augmented Reality (ISMAR)*, Nara, Japan.

Koenderink, J. (1984). The structure of images. *Biological Cybernetics*, 50:363–370.

Konolige, K. (1997). Small vision system: Hardware and implementation. *Proc. of the Intl. Symp. of Robotics Research (ISRR)*, pages 111–116.

Konolige, K. and Agrawal, M. (2008). FrameSLAM: from bundle adjustment to real-time visual mapping. *IEEE Trans. Robotics*, 24(5):1066–1077.

Kuijper, A. (2009). Geometrical PDEs based on second-order derivatives of gauge coordinates in image processing. *Image and Vision Computing*, 27(8):1023–1034.

Kulyukin, V., Gharpure, C., Nicholson, J., and Pavithran, S. (2004). RFID in robot assisted indoor navigation for the visually impaired. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*.

Kwak, N., Stasse, O., Foissotte, T., and Yokoi, K. (2009). 3D grid and particle based SLAM for a humanoid robot. In *IEEE-RAS Intl. Conference on Humanoid Robots*, pages 62–67, Paris, France.

Lenz, P., Ziegler, J., Geiger, A., and Roser, M. (2011). Sparse scene flow segmentation for moving object detection in urban environments. In *IEEE Intelligent Vehicles Symposium (IV)*, Baden-Baden, Germany.

Lewis, D. (1998). Naive (bayes) at forty: The independence assumption in information retrieval. In *Eur. Conf. on Machine Learning (ECML)*, pages 4–15.

Li, L., Socher, R., and Fei-Fei, L. (2009). Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Miami, FL, USA.

Lindeberg, T. (1998). Feature detection with automatic scale selection. *Intl. J. of Computer Vision*, 30(2):77–116.

Liu, J., Phillips, C., and Daniilidis, K. (2010). Video-based localization without 3D mapping for the visually impaired. In *Workshop on Computer Vision Applications for the Visually Impaired as part of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Liu, R. and Wang, Y. (2009). SAR Image Matching Based on Speeded Up Robust Feature. In *WRI Global Congress on Intelligent Systems*.

Llorca, D., Sotelo, M., Parra, I., Ocaña, M., and Bergasa, L. (2010). Error analysis in a stereo vision-based pedestrian detection sensor for collision avoidance applications. *Sensors*, 10:3741–3758.

Loomis, J., Golledge, R. G., and Klatzky, R. L. (2001). GPS-based navigation systems for the visually impaired. *Fundamentals of Wearable Computers and Augmented Reality*, pages 429–446.

Lourakis, M. (2004). levmar: Levenberg-Marquardt nonlinear least squares algorithms in C/C++. [web page] `http://www.ics.forth.gr/~lourakis/levmar/`.

Lourakis, M. A. and Argyros, A. (2009). SBA: A software package for generic sparse bundle adjustment. *ACM Trans. Math. Software*, 36(1):1–30.

Lovegrove, S. and Davison, A. (2010). Real-time spherical mosaicing using whole image alignment. In *Eur. Conf. on Computer Vision (ECCV)*, Crete, Greece.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *Intl. J. of Computer Vision*, 60(2):91–110.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Intl. Conf. on Computer Vision (ICCV)*, pages 1150–1157, Corfu, Greece.

Lu, C., Hager, G., and Mjolsness, E. (2000). Fast and globally convergent pose estimation from video images. *IEEE Trans. Pattern Anal. Machine Intell.*, 22(6):610–622.

Lucas, B. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Intl. Joint Conf. on AI (IJCAI)*, pages 674–679.

Álvarez, L., Guichard, F., Lions, P., and Morel, J. M. (1993). Axioms and fundamental equations of image processing. *Arch. for Rational Mechanics*, 123(3):199–257.

Álvarez, L., Lions, P., and Morel, J. (1992). Image selective smoothing and edge detection by nonlinear diffusion. *SIAM Journal on Numerical Analysis (SINUM)*, 29:845–866.

Mahalanobis, P. (1936). On the generalised distance in statistics. In *Proceedings National Institute of Science, India*, pages 49–55.

Marquardt, D. (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics (SIAP)*, 11(2):431–441.

Mei, C., Benhimane, S., Malis, E., and Rives, P. (2008). Efficient homography-based tracking and 3-D reconstruction for single viewpoint sensors. *IEEE Trans. Robotics*, 24(6):1352–1364.

Mei, C., Sibley, G., Cummins, M., Newman, P., and Reid, I. (2010). REAL: A system for large-scale mapping in constant-time using stereo. *Intl. J. of Computer Vision*.

Michel, P., Chestnutt, J., Kagami, S., Nishiwaki, K., Kuffner, J., and Kanade, T. (2007). GPU-accelerated Real-Time 3D Tracking for Humanoid Locomotion and Stair Climbing. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 463–469, San Diego, CA, USA.

Mikolajczyk, K. and Schmid, C. (2004). Scale and affine invariant interest point detectors. *Intl. J. of Computer Vision*, 60:63–86.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Machine Intell.*, 27(10):1615–1630.

Milford, M. and Wyeth, G. (2008). Mapping a suburb with a single camera using a biologically inspired SLAM system. *IEEE Trans. Robotics*, 24(5):1038–1053.

Morel, J. and Yu, G. (2009). ASIFT: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences (SIIMS)*, 2(2):438–469.

Mountney, P., Stoyanov, D., Davison, A. J., and Yang, G. Z. (2006). Simultaneous stereoscope localization and soft-tissue mapping for minimally invasive surgery. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*.

Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F., and Sayd, P. (2009). Generic and real-time structure from motion using Local Bundle Adjustment. *Image and Vision Computing*, 27:1178–1193.

Müller, T., Rannacher, J., Rabe, C., and Franke, U. (2011). Feature-and depth-supported modified total variation optical flow for 3D motion field estimation in real scenes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Nedevschi, S., Bota, S., and Tomiuc, C. (2009). Stereo-based pedestrian detection for collision-avoidance applications. *IEEE Trans. on Intelligent Transportation Systems*, 10(3):380–391.

Newcombe, R. and Davison, A. (2010). Live dense reconstruction with a single moving camera. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Ni, K., Kannan, A., Criminisi, A., and Winn, J. (2008). Epitomic location recognition. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Ni, K., Steedly, D., and Dellaert, F. (2007). Out-of-Core Bundle Adjustment for Large-Scale 3D Reconstruction. In *Intl. Conf. on Computer Vision (ICCV)*.

Nistér, D., Naroditsky, O., and Bergen, J. (2004). Visual Odometry. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Nistér, D. and Stewénius, H. (2006). Scalable recognition with a vocabulary tree. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Oh, S., Tariq, S., Walker, B., and Dellaert, F. (2004). Map-based priors for localization. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*.

Oliva, A. and Torralaba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *Intl. J. of Computer Vision*, 42:145–175.

Ozawa, R., Takaoka, Y., Kida, Y., Nishiwaki, K., Chestnutt, J., and Kuffner, J. (2007). Using visual odometry to create 3D maps for online footstep planning. In *IEEE Intl. Conference on Systems, Man and Cybernetics*, pages 2643–2648, Hawaii, USA.

Paul, R. and Newman, P. (2010). FAB-MAP 3D: Topological mapping with spatial and visual appearance. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Anchorage, AK.

Paz, L. M., Guivant, J., Tardós, J. D., and Neira, J. (2007). Data association in O(n) for divide and conquer SLAM. In *Robotics: Science and Systems (RSS)*, Atlanta, USA.

Paz, L. M., Piniés, P., Tardós, J. D., and Neira, J. (2008). Large scale 6DOF SLAM with stereo-in-hand. *IEEE Trans. Robotics*, 24(5).

Perona, P. and Malik, J. (1990). Scale-space and edge detection using annisotropic diffusion. *IEEE Trans. Pattern Anal. Machine Intell.*, 12(7):1651–1686.

Perrin, N., Stasse, O., Lamiraux, F., and Yoshida, E. (2010). Approximation of feasibility tests for reactive walk on HRP-2. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 4243–4248, Anchorage, AK.

Petrie, H., Johnson, V., Strothotte, T., Raab, A., Fritz, S., and Michel, R. (1996). MoBIC: Designing a travel aid for blind and elderly people. *Jnl. of Navigation*, 49(1):45–52.

Platel, B., Balmachnova, E., Florack, L., and ter Haar Romeny, B. (2006). Top-Points as interest points for image matching. In *Eur. Conf. on Computer Vision (ECCV)*.

Pock, T., Urschler, M., Zach, C., Reinhard, B., and Bischof, H. (2007). A duality based algorithm for TV-L1-Optical-Flow image registration. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*.

Pollefeys, M., Nistér, D., Frahm, J., Akbarzadeh, A., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Kim, S., Merrell, P., Salmi, C., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewenius, H., Yang, R., Welch, G., and Towles, H. (2008). Detailed real-time urban 3D reconstruction from video. *Intl. J. of Computer Vision*, 78(2):143–167.

Pradeep, V., Medioni, G., and Weiland, J. (2010). Robot vision for the visually impaired. In *CVAVI10, IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA.

Pretto, A., Menegatti, E., Bennewitz, M., Burgard, W., and Pagello, E. (2009). A visual odometry framework robust to motion blur. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Kobe, Japan.

Rabe, C., Müller, T., Wedel, A., and Frank, U. (2010). Dense, robust, and accurate motion field estimation from stereo image sequences in real-time. In *Eur. Conf. on Computer Vision (ECCV)*, pages 582–595.

Ranganathan, A. and Dellaert, F. (2009). Bayesian surprise and landmark detection. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Kobe, Japan.

Ranganathan, A. and Dellaert, F. (2010). Online probabilistic topological mapping. *Intl. J. of Robotics Research*.

Rothwell, C., Zisserman, A., Forsyth, D., and Mundy, J. (1992). Canonical frames for planar object recognition. In *Eur. Conf. on Computer Vision (ECCV)*, pages 757–772.

Royer, E., Lhuillier, M., Dhome, M., and Chateau, T. (2005). Localization in urban environments: monocular vision compared to a differential gps sensor. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Russell, S. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Pearson Education.

Saéz, J. M. and Escolano, F. (2008). Stereo-based aerial obstacle detection for the visually impaired. In *European Conference on Computer Vision (ECCV) / Workshop on Computer Vision Applications for the Visually Impaired (CVAVI)*, Marselle, France.

Saéz, J. M., Escolano, F., and Peñalver, A. (2005). First steps towards stereo-based 6DOF SLAM for the visually impaired. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, San Diego, USA.

Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Intl. J. of Computer Vision*, 47:7–42.

Schindler, G., Brown, M., and Szeliski, R. (2007). City-scale location recognition. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Schindler, G. and Dellaert, F. (2010). Probabilistic temporal inference on reconstructed 3D scenes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA.

Schindler, G., Krishnamurthy, P., Lublinerman, R., Liu, Y., , and Dellaert, F. (2008). Detecting and matching repeated patterns for automatic Geo-tagging in urban environments. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Schleicher, D., Bergasa, L. M., Ocaña, M., Barea, R., and López, E. (2009). Real-time hierarchical outdoor SLAM based on stereovision and GPS fusion. *IEEE Trans. on Intelligent Transportation Systems*, 10(3):440–452.

Schmid, C. and Mohr, R. (1995). Matching by local invariants. Technical report, INRIA.

Schmid, C. and Mohr, R. (1997). Local grayvalue invariants for image retrieval. *IEEE Trans. Pattern Anal. Machine Intell.*, 19(5):530–535.

Schweighofer, G. and Pinz, A. (2006). Fast and globally convergent structure and motion estimation for general camera models. In *British Machine Vision Conf. (BMVC)*.

Schweighofer, G. and Pinz, A. (2008). Globally optimal O(n) solution to the PnP problem for general camera models. In *British Machine Vision Conf. (BMVC)*.

Sivic, J. and Zisserman, A. (2003). Video Google: A text retrieval approach to object matching in videos. In *Intl. Conf. on Computer Vision (ICCV)*.

Snavely, N., Seitz, S. M., and Szeliski, R. (2006). Photo Tourism: Exploring image collections in 3D. In *SIGGRAPH*.

Stachniss, C., Bennewitz, M., Grisetti, G., Behnke, S., and Burgard, W. (2008). How to learn accurate grid maps with a humanoid. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Pasadena, CA, USA.

Stasse, O., Davison, A., Sellaouti, R., and Yokoi, K. (2006). Real-time 3D SLAM for a humanoid robot considering pattern generator information. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 348–355, Beijing, China.

Stasse, O., Larlus, D., Lagarde, B., Escande, A., Saidi, F., Kheddar, A., Yokoi, K., and Jurie, F. (2007). Towards Autonomous Object Reconstruction for Visual Search by the Humanoid Robot HRP-2. In *IEEE-RAS Intl. Conference on Humanoid Robots*, pages 151–158, Pittsburg, USA.

Stasse, O., Verrelst, B., Wieber, P., Vanderborght, B., Evrard, P., Kheddar, A., and Yokoi, K. (2008). Modular architecture for humanoid walking pattern prototyping and experiments. *Advanced Robotics, Special Issue on Middleware for Robotics–Software and Hardware Module in Robotics System*, 22(6):589–611.

Strasdat, H., Montiel, J., and Davison, A. (2010a). Real-time monocular SLAM: Why filter? In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Anchorage, USA.

Strasdat, H., Montiel, J., and Davison, A. (2010b). Scale-drift aware large scale monocular SLAM. In *Robotics: Science and Systems (RSS)*, Zaragoza, Spain.

Tariq, S. and Dellaert, F. (2004). A multi-camera 6-DOF pose tracker. In *IEEE and ACM Intl. Sym. on Mixed and Augmented Reality (ISMAR)*, pages 296–297.

ter Haar Romeny, B. M. (2003). *Front-End Vision and Multi-Scale Image Analysis. Multi-Scale Computer Vision Theory and Applications, written in Mathematica*. Kluwer Academic Publishers.

Tola, E., Lepetit, V., and Fua, P. (2010). DAISY: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Trans. Pattern Anal. Machine Intell.*, 32(5):815–830.

Torralba, A., Murphy, K., Freeman, W., and Rubin, M. (2003). Context-based division system for place and object recognition. In *Intl. Conf. on Computer Vision (ICCV)*, pages 273–280.

Treuillet, S., Royer, E., Chateau, T., Dhome, M., and Lavest, J. (2007). Body mounted vision system for visually impaired outdoor and indoor wayfinding assistance. In *Conference and Workshop on Assistive Technologies for People with Vision and Hearing Impairments. Assistive Technology for All Ages*.

Triggs, B., McLauchlan, P., Hartley, R., and Fitzgibbon, A. (1999). Bundle adjustment – a modern synthesis. In Triggs, W., Zisserman, A., and Szeliski, R., editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag.

Vedaldi, A. (2007). An open implementation of the SIFT detector and descriptor. Technical Report 070012, UCLA CSD.

Vedaldi, A. and Fulkerson, B. (2008). VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`.

Vedula, S., Baker, S., Rander, P., Collins, R., and Kanade, T. (1999). Three-dimensional scene flow. In *Intl. Conf. on Computer Vision (ICCV)*.

Vedula, S., Baker, S., Rander, P., Collins, R., and Kanade, T. (2005). Three-dimensional scene flow. *IEEE Trans. Pattern Anal. Machine Intell.*, 27(3):475–480.

Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *Intl. J. of Computer Vision*, 57(2):137–154.

Walker, B. N. and Lindsay, J. (2005). Using virtual reality to prototype auditory navigation displays. *Assistive Technology Journal*, 17(1):72–81.

Walker, B. N. and Lindsay, J. (2006). Navigation performance with a virtual auditory display: Effects of beacon sound, capture radius, and practice. *Human Factors*, 48(2):265–278.

Wang, C., Thorpe, C., Thrun, S., Hebert, M., and Durrant-Whyte, H. (2007). Simultaneous localization, mapping and moving object tracking. *Intl. J. of Robotics Research*, 26:889–916.

Wangsiripitak, S. and Murray, D. (2009). Avoiding moving outliers in visual SLAM by tracking moving objects. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 375–380.

Wedel, A., Brox, T., Vaudrey, T., Rabe, C., Franke, U., and Cremers, D. (2011). Stereoscopic scene flow computation for 3D motion understanding. *Intl. J. of Computer Vision*, 35(1):29–51.

Wedel, A., Meißner, A., Rabe, C., Franke, U., and Cremers, D. (2009). Detection and segmentation of independently moving objects from dense scene flow. In *Proceedings of the 7th International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, pages 14–27.

Weickert, J., ter Haar Romeny, B. M., and Viergever, M. A. (1998). Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Trans. Image Processing*, 7(3).

Weinberger, K. Q. and Tesauro, G. (2007). Metric learning for kernel regression. In *In Proceedings of the Eleventh International Workshop on Artificial Intelligence and Statistics (AISTATS)*, Puerto Rico.

Wendel, A., Irschara, A., and Bischof, H. (2011). Natural landmark-based monocular localization for MAVs. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 5792–5799, Shanghai, China.

Williams, B., Klein, G., and Reid, I. (2007). Real-time SLAM relocalisation. In *Intl. Conf. on Computer Vision (ICCV)*.

Williams, B. and Reid, I. (2010). On combining visual SLAM and visual odometry. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3494–3500, Anchorage, AK, USA.

Wolberg, J. (1967). *Prediction Analysis*. Van Nostrand.

Wuest, H., Pagani, A., and Stricker, D. (2007). Feature management for efficient camera tracking. In *Asian Conf. on Computer Vision (ACCV)*.

Xu, J., Wang, Q., and Yang, J. (2010). Modeling urban scenes in the spatial-temporal space. In *Asian Conf. on Computer Vision (ACCV)*, pages 374–387, Queenstown, New Zealand.

Yebes, J., Alcantarilla, P., and Bergasa, L. (2011). Occupant monitoring system for traffic control based on visual categorization. In *IEEE Intelligent Vehicles Symposium (IV)*.

Yuen, D. C. K. and MacDonald, B. A. (2005). Vision-based localization algortihm based on landmark matching, triangulation, reconstruction, and comparison. *IEEE Trans. Robotics*, 21(2):217–226.

Zhu, Z., Oskiper, T., Samarasekera, S., Kumar, R., and Sawhney, H. (2008). Real-time global localization with a pre-built visual landmark database. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.