TESIS DE LA UNIVERSIDAD DE ZARAGOZA

2015

11

Alejandro Rituerto Sin

Modeling the environment with egocentric vision systems

Departamento

Informática e Ingeniería de Sistemas

Director/es

Murillo Arnal, Ana Cristina

Guerrero Campo, José Jesús

ISSN 2254-7606

Prensas de la Universidad Universidad Zaragoza

Universidad Zaragoza
1542

Tesis Doctoral

MODELING THE ENVIRONMENT WITH EGOCENTRIC VISION SYSTEMS

Autor

Alejandro Rituerto Sin

Director/es

Murillo Arnal, Ana Cristina
Guerrero Campo, José Jesús

UNIVERSIDAD DE ZARAGOZA

Informática e Ingeniería de Sistemas

2014

Universidad Zaragoza

Instituto Universitario de Investigación de Ingeniería de Aragón Universidad Zaragoza

Modeling the environment with egocentric vision systems

Alejandro Rituerto Sin

Ph.D. DISSERTATION

Instituto de Investigación en Ingeniería de Aragón
Dpto. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza

Supervisors: Dr. José Jesús Guerrero Campo
Dr. Ana Cristina Murillo Arnal

October 2014

Modeling the environment with egocentric vision systems

Alejandro Rituerto Sin

Supervisors

José Jesús Guerrero Campo	Universidad de Zaragoza, Spain
Ana Cristina Murillo Arnal	Universidad de Zaragoza, Spain

Dissertation Committee

José María Armingol	Universidad Carlos III, Spain
Walterio W. Mayol-Cuevas	University of Bristol, United Kingdom
Gonzalo López Nicolás	Universidad de Zaragoza, Spain

International Reviewers

Roberto Manduchi	University of California Santa Cruz, California
Henrik Andreasson	Örebro Universitet, Sweden

Acknowledgments

These lines are dedicated to everyone who helped me during these years, whether they were directly or indirectly involved in the development of this thesis.

First of all, I would like to thank my supervisors, Josechu Guerrero and Ana C. Murillo. First for giving me the opportunity to work with them, and second for their support, patience and good advice during these years. I would like to thank also Luis Puig, who helped and guided me during my first research work. Also many thanks to the committee members: José María Armingol, Walterio W. Mayol-Cuevas and Gonzalo López, and to the international reviewers: Roberto Manduchi and Henrik Andreasson, for spending their time evaluating my work and giving me great advice.

During my time as a graduate student, I have had the opportunity of learning and developing interesting work for my thesis, but most important, I have had the opportunity of meeting a lot of incredible people. I would like to thank Achim Lilienthal and Henrik Andreasson, from Örebro University. They hosted me and helped me during my stay in their lab in 2011, I learned a lot with them. I thank also Roberto Manduchi, for giving me the opportunity of collaborating with him at the Computer Vision Lab at UCSC in Santa Cruz, not only once but twice. Thanks also to all the friends I met during these stays, they made me feel like being at home and we shared many good moments. I would like to thank the Spanish Ministry of Economy and Competivity for the FPI scholarship (BES-2010-030299), which allowed the development of this thesis.

Finally, I would like to thank my family and friends for all their support, they have always been there for the bad and the good moments.

Resumen

Cada vez más sistemas autónomos, ya sean robots o sistemas de asistencia, están presentes en nuestro día a día. Este tipo de sistemas interactúan y se relacionan con su entorno y para ello necesitan un modelo de dicho entorno. En función de las tareas que deben realizar, la información o el detalle necesario del modelo varía. Desde detallados modelos 3D para sistemas de navegación autónomos, a modelos semánticos que incluyen información importante para el usuario como el tipo de área o qué objetos están presentes. La creación de estos modelos se realiza a través de las lecturas de los sensores disponibles en el sistema. Actualmente, gracias a su pequeño tamaño, bajo precio y la gran información que son capaces de capturar, las cámaras son sensores incluidos en todos los sistemas autónomos. El objetivo de esta tesis es el desarrollar y estudiar nuevos métodos para la creación de modelos del entorno a distintos niveles semánticos y con distintos niveles de precisión. Dos puntos importantes caracterizan el trabajo desarrollado en esta tesis:

• El uso de cámaras con punto de vista egocéntrico o en primera persona ya sea en un robot o en un sistema portado por el usuario (*wearable*). En este tipo de sistemas, las cámaras son solidarias al sistema móvil sobre el que van montadas. En los últimos años han aparecido muchos sistemas de visión *wearables*, utilizados para multitud de aplicaciones, desde ocio hasta asistencia de personas.

• El uso de sistemas de visión omnidireccional, que se distinguen por su gran campo de visión, incluyendo mucha más información en cada imagen que las cámara convencionales. Sin embargo plantean nuevas dificultades debido a distorsiones y modelos de proyección más complejos.

Esta tesis estudia distintos tipos de modelos del entorno:

• *Modelos métricos:* el objetivo de estos modelos es crear representaciones detalladas del entorno en las que localizar con precisión el sistema autónomo. Esta tesis se centra en la adaptación de estos modelos al uso de visión omnidireccional, lo que permite capturar más información en cada imagen y mejorar los resultados en la localización.

• *Modelos topológicos:* estos modelos estructuran el entorno en nodos conectados por arcos. Esta representación tiene menos precisión que la métrica, sin embargo presenta un nivel de abstracción mayor y puede modelar el entorno con más riqueza. Esta tesis se centra en la creación de modelos topológicos con información adicional sobre el tipo de área de cada nodo y conexión (pasillo, habitación, puertas, escaleras...).

• *Modelos semánticos:* este trabajo también contribuye en la creación de nuevos modelos semánticos, más enfocados a la creación de modelos para aplicaciones en las que el sistema interactúa o asiste a una persona. Este tipo de modelos representan el entorno a través de conceptos cercanos a los usados por las personas. En particular, esta tesis desarrolla técnicas para obtener y propagar información semántica del entorno en secuencias de imágenes.

Abstract

More and more intelligent systems, both robots and wearable systems, are present in our everyday life. This kind of systems interact with the environment so they need suitable models of their surrounding. Depending on the tasks that they have to perform, the information or detail required in those models changes: from highly detailed 3D models for autonomous navigation systems, to semantic models including important information for the user such as the kind of area being traversed or the presence of objects. These models are created from the sensory data provided by the system. Cameras are an important sensor included in any intelligent system, thanks to their small size, cheap prices and the great amount of information that they provide. This thesis studies and develops new methods to create models of the environment with different semantic levels and different precision levels. There are two key-points in the subsequent approaches presented:

• The use of egocentric vision systems. All the vision systems and image sequences used in this thesis characterize for a first-person (egocentric) point of view, where the camera moves with the system they are mounted on. Many wearable vision systems have appeared in the last years. These systems are used in many applications, from leisure to human assistance.

• The use of omnidirectional vision. This kind of vision systems provide much more information that conventional cameras thanks to their wide field of view. However, they present additional difficulties due to distortions and require more complex projection models.

This work studies different kinds of models of the environment:

• *Metric models:* the objective of these models is to create accurate representations of the environment to localize the system with precision. This thesis focuses on the adaptation of this kind of models to use omnidirectional cameras, allowing to capture more information on each image and improve the localization results.

• *Topological models:* these models segment the environment in nodes related by links connecting them. These models have lower precision than metric models, however, they show higher abstraction and can include additional information about the environment, e.g. the kind of area of each node, the location of important objects or the kind of connection of each link. This thesis focuses on the creation of topological models including enhanced information about the kind of area of the different nodes and links (e.g. room, corridor, door, stairs)

• *Semantic models:* this thesis also contributes on the creation of semantic models of the environment and is focused on the creation of models for applications where the autonomous system interacts with a person. This kind of models represent the environment using concepts close to human concepts. Particularly, this thesis develops techniques to get and propagate semantic information of the environment in sequences of images.

Contents

Chapter 1

Introduction

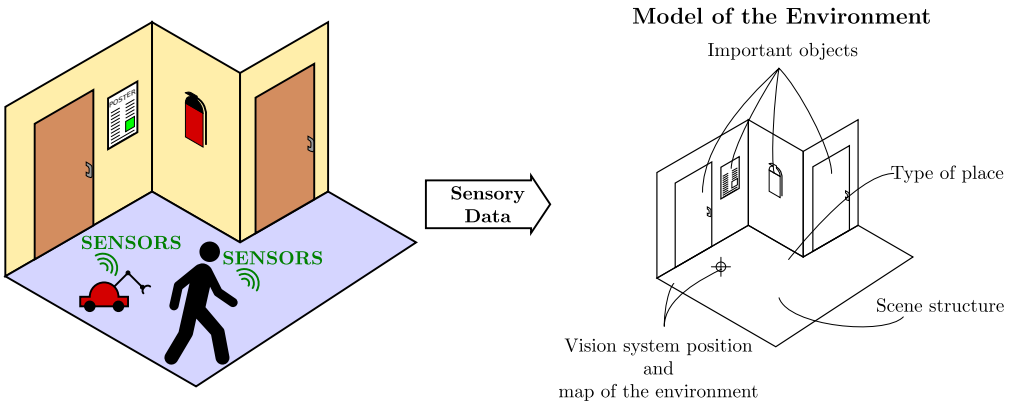


Figure 1.1: Intelligent systems need models of their environment to perform the tasks they are designed for. These models are the result of translating this data into useful information for the system.

1.1 Computer vision to model the environment

In the last years, the presence of intelligent or autonomous systems has become something usual in people daily life. Cleaning robotic systems [iRobot, 2002], tele-presence sysems [Double Robotics, 2011], surveillance [SDR Tactical Robots, 2000] or aerial photo&videography systems [Ascending Technologies, 2013] are widely used. At the same time, smart-phones have become a commodity that people carries everyday, and we see how each day these devices get new functionalities and sensors [Google, 2014b] that make them more powerful, and closer to what we consider intelligent systems. More recently, we are witnessing how many and different wearable platforms are appearing. Wristbands and watches [Garmin, 2014, pebble, 2012], glasses [Google, 2014a, Recon Instruments, 2014, OrCam Technologies, 2010] or devices to hang from your neck [Microsoft Research, 2004, Autographer, 2013], are getting more and more interest from the general public. A common issue of any of these intelligent systems, is the need of a suitable representation that allows them to move, interact and understand its surroundings.

The best suited representation or model of the environment depends on the task that each intelligent system is designed to perform. For example, in the case of robotic platforms, the model typically requires a high precision representation to facilitate accurate localization of the system within the environment. These highly detailed models are usually obtained using Simultaneous Localization and Mapping (SLAM) [Thrun et al., 2005] or Structure from Motion (SfM) [Hartley and Zisserman, 2003] techniques. These approaches have shown extraordinary mapping results [Davison et al., 2007, Scaramuzza et al., 2006b]. However, the general drawback of this type of approaches is that they tend to fail for big environments or long operation times [Hochdorfer and Schlegel, 2009].

Topological models are one way to deal with some of the drawbacks of metric approaches. They organize the environment into nodes and links connecting them, where each node may represent a location or region of the environment. Metric and topologi-

cal models can be used together, as their information is complementary. Sometimes, an environment may be too large and complex to represent it with a single highly detailed model. In these cases, we can consider the use of a topological model. A topological model can organize the environment in smaller areas or nodes. Then, a metric model, can be used to create detailed representations of each of these areas. This framework is usually known as hierarchical modeling, a kind of environment representation where different modeling techniques work together [Kuipers, 2000, Murillo et al., 2007].

A key feature of topological models is the ability to integrate higher abstraction levels, such as the properties of the different areas of the environment [Friedman et al., 2007] or the location of important scene elements [Viswanathan et al., 2011]. This makes topological models suitable representations to design systems for human-computer interaction, where the model of the environment needs to include information useful for a person. Knowing the kind of area traversed by the system, the presence of objects or people can be very useful for robotic platforms, but it is mandatory for systems designed for human-computer interaction or human assistance.

In order to obtain models of the environment better suited for human-computer interaction, researchers work to enrich these models with concepts that can be understood by people. The use of priors about the environment presents a powerful tool to integrate higher abstraction concepts in our models. Even the human brain makes assumptions about the environment based on visual experience. For example, we assume that light comes from above and that objects are not viewed from below; we recognize faces if we see them upright and assume that all the borders are convex. Our visual perception is the result of this kind of unconscious inferences.

Usual priors that can be used for intelligent systems are, for example, assumptions on the type of environment [Coughlan and Yuille, 1999], restricting the camera position and motion [Tsai and Kuipers, 2012] or where the different elements of the environment are and how they are sensed by the system [Yang and Newsam, 2011]. This prior knowledge about the environment can be added to the process of building the model in such a way that the resulting models integrate information closer to human concepts.

Intelligent systems relate with their environment trough sensors. Many different sensors have been used for modeling the environment: i.e. laser, cameras, RGB-D cameras, odometry, different sensors mean different advantages and different challenges. This thesis studies and proposes novel ways to model the environment using vision sensors. Lately, cameras are included as sensor in any platform. Their low price and small size make cameras easy to integrate in any platform. At the same time, the increasing resolution allows to capture more and more details and useful information in each image.

Computer vision has been shown to be a useful tool to model the environment. From the creation of high precision models [Davison et al., 2007] or topological models [Cummins and Newman, 2011] to the detection and recognition of particular objects [Tuytelaars et al., 2010] and places [Valgren and Lilienthal, 2010], computer vision helped to improve the creation of many kinds of representations. The rich information provided in each acquired image, makes cameras a great sensor to create semantic representations of the environment. Particularly the use of omnidirectional cameras, where the amount

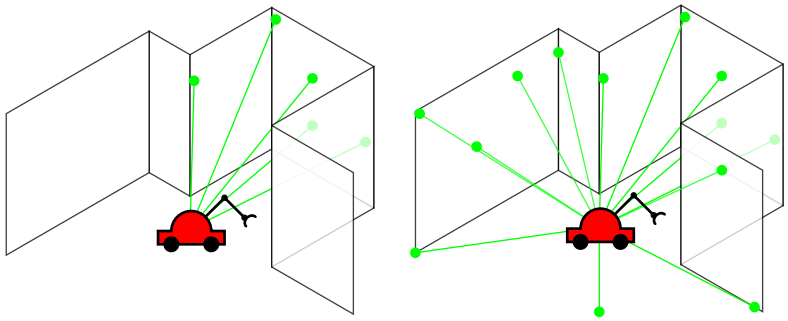


Figure 1.2: Conventional versus Omnidirectional cameras. The use of omnidirectional cameras present a great advantage over conventional cameras when creating metric models of the environment. A metric model is compound by the different scene 3D key-points (green) and the system 3D position. Key-points all around the environment are visible with an omnidirectional camera.

of information included in each image is bigger than in conventional cameras due to the large field of view. The whole surrounding scene of the camera can be captured in just one image. However, a large field of view means higher distortions and more complex projection models must be considered.

A key element of this thesis is the use of wearable sensors, particularly egocentric cameras. Egocentric cameras move together with the system. This is the typical framework in robotics and the natural one when addressing human assistance intelligent systems.

Egocentric vision systems and datasets. For the different modeling approaches presented in this thesis, different camera systems and image datasets have been used, all of them are further detailed in Appendix A. A shared characteristic among all these datasets, which is actually one of the main focus of this work, is that they have been acquired from an ego-centric point of view, with a robotic platform or a wearable system. About the vision systems, we have focused in omnidirectional cameras both catadioptric, composed by a curved mirror and a conventional camera, and dioptric in the form of cameras with fish-eye lenses. We have built different wearable prototypes with cameras mounted in backpacks and in a helmet. Additionally we have used other publicly available datasets: robotic acquired sequences from the Rawseeds Project, hand held sequences from the dataset presented in [Furlan et al., 2013] and fish-eye sequences from the Wearable Computer Vision Systems dataset presented for the Workshop on Wearable Computer Vision Systems (WWCV 2013) held with ICCV.

1.1.1 Metric models

An initial and important problem to solve by any autonomous system is the localization. The system needs to define his own location and the location of important objects in

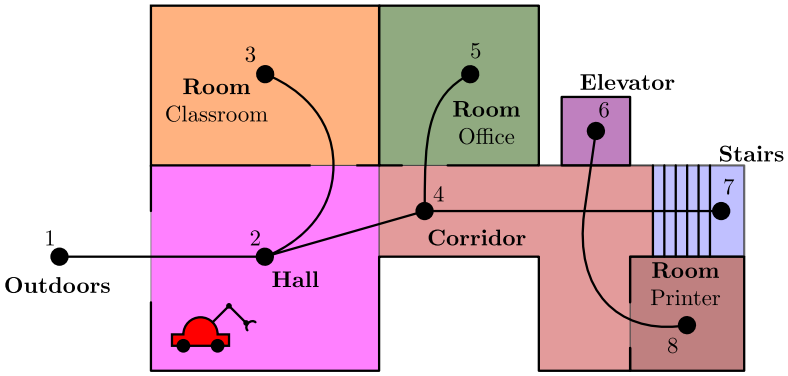


Figure 1.3: A topological model represents the environment with higher abstraction levels. This kind of models can be easily enhanced by adding information about the kind of area of each node (i.e. room, corridor, hall,...) or how these areas are connected (i.e. door, elevator, stairs,...).

the environment. The detail level needed for this localization depends on the task to be performed and the capacity of the system to move by itself. An autonomous robot requires high detailed models of the environment to be able to avoid collisions, compute the best trajectory to his destination and, at the same time, keep track of the environment and its elements. The problem of metric localization has been studied for long both in robotics [Thrun et al., 2005] and in computer vision [Hartley and Zisserman, 2003].

Typically, visual metric models use conventional cameras, however, the use of omnidirectional cameras can greatly improve the performance of such techniques. Being able to observe the whole scene in one image allows to keep track of the scene key-points for longer time than with conventional cameras, Fig. 1.2. This leads to more robust solutions. Another point where omnidirectional cameras present an advantage over conventional cameras is under the presence of untextured areas in the scene. With a conventional camera, the small field of view may cause the loss of scene key-points when the camera is pointing directly to a surface without texture. This will lead to localization errors and the system can lose track of his position or the map. However, with an omnidirectional camera, key-points in visible textured parts of the scene appear in the image, even when facing untextured surfaces.

1.1.2 Enhanced topological models

As previously mentioned, different tasks need different models. In some cases, detailed metric models may not be needed or may not be powerful enough to model large environments. In those cases the use of topological models is the right choice. Topological models describe the environment as a set of nodes and links between them. This kind of models are not as precise as metric models, but present a higher abstraction level. They can be used together with a metric model so both models complement their respective weaknesses: the error in the localization for large environments of the metric models and

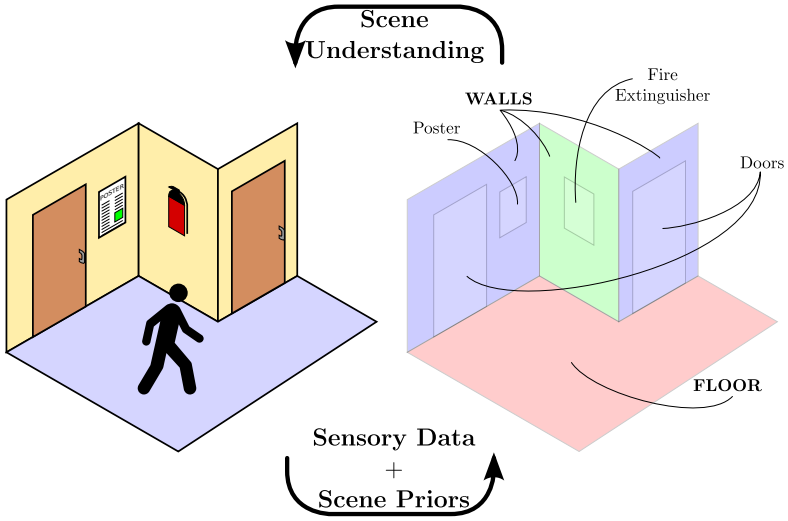


Figure 1.4: Semantic models include high level information of the environment. The integration of prior knowledge about the system surrounding in the environment model allows to better understand the sensory data and create interpretations closer to human concepts.

the lack of precision of the topological models. However, a topological model can be the only model of a system that does not requires high precision.

An key feature of topological models is that they can include higher information levels than metric maps, Fig. 1.3. Nodes can represent the different positions of important elements of the scene (i.e. important objects, places). They can also inform about the different areas of an environment, even detail properties of these areas such as if they are an office, a kitchen or a corridor, or how these areas are connected, through a door, an elevator or stairs. All this information is very useful for a robot, since it can determine how to move from one area to another, and it is very informative for human-assistance systems since those concepts are used also by people.

1.1.3 Semantic models for environment understanding

Next objective is to create models representing the environment as close to human concepts as possible. A common approach to include such concepts is to rely in priors about the scene to include such concepts in our environment model. Previous knowledge about the environment is mostly used when working indoors. Assumptions about the scene structure can be made (i.e. Manhattan World [Coughlan and Yuille, 1999]). Also restrictions about the position of important and characteristic elements of the environment (i.e. light sources on the ceiling, signs in the walls, obstacles in the floor). Another restriction that can be really helpful for this purpose is the spatio-temporal restriction imposed by a video sequence. When working with cameras on autonomous systems, data usually comes in the from of video sequences. The changes between one image and the next one

are usually small, providing multiple measurements of a same scene and allowing to track scene objects. Integrating this constraints about the scene in the process of building a model, the resulting model will better represent the environment with concepts closer human understanding, Fig. 1.4.

One possible approach to integrate these restrictions to build a model is to keep track of important scene elements [Sivic and Zisserman, 2003]. By saving the appearance and the position of such elements, the system will be able to recognize them later. The environment can then be described by the occurrences of these scene elements that represent real objects of the environment. A different approach is to use the environment constraints to infer information about the environment and create interpretations of the surrounding scene. With this intuition, we find many papers that infer the spatial layout of a scene from just one image [Delage et al., 2006]. These approaches rely on assumptions about man made environments and how this kind of scenes are projected in images.

1.2 Goals and contributions

This thesis studies how computer vision can be used to improve, adapt or design different models of the environment. To test our proposals, different cameras have been used, both in robotic and wearable platforms.

Omnidirectional cameras for visual SLAM. The SLAM (Simultaneous Localization and Mapping) problem is one of the essential challenges for the current robotics. Chapter 2 presents our approach to integrate the Spherical Camera Model for catadioptric systems in a Visual-SLAM application. The proposal uses the inverse depth parameterization of the measurements and SIFT [Lowe, 2004] points as image key-points. The Spherical Camera Model is a projection model that unifies the projection of central catadioptric and conventional cameras. To integrate this model into the Extended Kalman Filter-based SLAM we require to linearize the direct and the inverse projection. The comparison of the performance of a visual SLAM system using monocular omnidirectional and conventional vision confirms that the latter produces better trajectory and orientation estimation than the conventional vision system.

Associated publications:

- [Rituerto et al., 2010b] Rituerto, A., Puig, L., and Guerrero, J.J. (2010b). Visual SLAM with an omnidirectional camera. In *International Conference on Pattern Recognition (ICPR)*, pages 348–351.

- [Rituerto et al., 2010a] Rituerto, A., Puig, L., and Guerrero, J.J. (2010a). Comparison of omnidirectional and conventional monocular systems for visual SLAM. In 10^{th} *Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS) (Best paper award).*

Semantic labeling for indoor topological mapping using a wearable catadioptric system. Richer semantic representations of the environment may allow autonomous systems to perform higher level tasks and provide better human-robot interaction. This issue is analyzed in Section 3. A new omnidirectional vision based scene labeling approach for augmented indoor topological mapping is presented. Our proposal includes novel ideas in order to augment the semantic information of a typical indoor topological map: we pay special attention to the semantic labels of the different types of transitions between places, and propose a simple way to include this information to build a topological map, as part of the criteria to segment the environment. This approach is built on an efficient catadioptric image representation based on the Gist descriptor, which is used to classify the acquired views into types of indoor regions. The basic types of indoor regions considered are *Place* and *Transition*, farthest divided into more specific subclasses, e.g *Transition* into *Door*, *Stairs* or *Elevator*. Besides using the result of this labeling, the proposed mapping approach includes a probabilistic model to account for spatio-temporal consistency. All the proposed ideas have been evaluated in a new indoor

dataset acquired with our wearable catadioptric vision system, showing promising results in a realistic prototype.

Associated publications:

- [Rituerto et al., 2014c] Rituerto, A., Murillo, A., and Guerrero, J.J. (2014c). Semantic labeling for indoor topological mapping using a wearable catadioptric system. *Robotics and Autonomous Systems, Special Issue Semantic Perception, Mapping and Exploration*, 62(5):685–695.

Line Image Signature, global descriptor for Scene Understanding. Pursuing similar objectives, section 4 proposes a novel line-based image global descriptor that encloses the structure of the scene observed. This descriptor is designed with omnidirectional imagery in mind, where observed lines are longer than in conventional images. Experiments using two public datasets analyze the performance of the descriptor for scene categorization. The descriptor has been tested with two omnidirectional systems: a catadioptric camera and panoramic images. The different parameters of the descriptor have been analyzed. Experiments show how the proposed descriptor can be used for indoor scene recognition with comparable results to state-of-the-art global descriptors. Additional advantages of particular interest for wearable vision systems are higher robustness to rotation, compactness, and easier integration with other scene understanding steps using the observed lines and contours.

Associated publications:

- [Rituerto et al., 2013] Rituerto, A., Murillo, A. C., and Guerrero, J.J. (2013). Line image signature for scene understanding with a wearable vision system. In *International SenseCam & Pervasive Imaging Conference*, pages 16–23.

- [Rituerto et al., 2014e] Rituerto, A., Murillo, A. C., and Guerrero, J.J. (2014e). Line-based global descriptor for omnidirectional vision. In *IEEE International Conference on Image Processing (ICIP)*.

Building a hierarchical vocabulary from an image sequence. Vision based recognition approaches frequently use quantized feature spaces, commonly know as Bag of Words (BoW) or vocabulary representations. A drawback using standard BoW approaches is that conceptual or semantic information is not considered as criteria to group the visual features into words. To solve this challenging task, section 5 studies how to leverage the standard vocabulary construction process to obtain a more meaningful visual vocabulary for particular applications using image sequences. We take advantage of spatio-temporal constraints and prior-knowledge about the position of the camera. The key contribution of our approach is to define a new method to incorporate tracking information to the process of vocabulary construction, and to add geometric cues to the appearance descriptors. Motivated by long-term indoor robotic applications, we focus in a robot camera pointing to the ceiling, which facilitates the capture of more stable regions of the environment, improving long term operation and the discovery of

repetitive and representative elements. The experimental validation shows how our vocabulary models the environment in more detail than standard vocabulary approaches, while keeping comparable recognition performance. We show different robotic tasks that could benefit of the use of our visual vocabulary approach, such as place recognition or object discovery. For this validation we use a publicly available dataset fitting the requirements of our approach.

Associated publications: This work was mainly developed during my 3 months stay in Örebro Sweden where I collaborated with Professor Achim Lilienthal and PhD Doctor Henrik Andreasson from the Mobile Robotics and Olfaction Lab in Örebro University.

- [Rituerto et al., 2014a] Rituerto, A., Andreasson, H., Murillo, A., , Lilienthal, A.J., and Guerrero, J.J. (2014a). Building a hierarchical vocabulary from an image sequence. *Pattern Recognition (Under review).*

3D Spatial layout propagation. Indoor scene understanding from monocular images has been widely studied, and a common initial step to solve this problem is the estimation of the layout of the scene, the basic 3D structure. Many previous approaches obtain the layout of a single image, however, section 6 addresses the problem of scene layout propagation along a video sequence. Our approach uses a Particle Filter framework to propagate the scene layout obtained using a state-of-the-art technique on the initial frame. We propose how to generate, evaluate and sample new layout hypotheses for the scene on each of the frame. The intuition we follow is that we can obtain better layout estimation at each frame through propagation than running separately at each image. The experimental validation is run on a publicly available indoor dataset and shows promising results for the layout computed using our approach, without the need of estimating accurate 3D maps. Additionally, they demonstrate how this layout information can be used to improve detection tasks, in particular sign detection, by easily rejecting false positives.

Associated publications: An important part of this work was developed during two stays of 3 and 4 months in Santa Cruz, California, working on the Computer Vision Lab at UCSC collaborating with Professor Roberto Manduchi.

- [Rituerto et al., 2014b] Rituerto, A., Manduchi, R., Murillo, A.C., and Guerrero, J.J. (2014b). 3D spatial layout propagation in a video sequence. In *International Conference on Image Analysis and Recognition (ICIAR).*

- [Rituerto et al., 2014d] Rituerto, A., Murillo, A. C., and Guerrero, J.J. (2014d). 3D layout propagation to improve object recognition in egocentric videos. In *Assistive Computer Vision and Robotics (ACVR).*

Additionally, I have collaborated with my supervisor in the direction of one undergraduate thesis project that resulted in the following publication:

- [Gutiérrez et al., 2011] Gutiérrez, D., Rituerto, A., Montiel, J., and Guerrero, J.J. (2011). Adapting a real-time monocular visual SLAM from conventional to omni-

directional cameras. In *International Conference on Computer Vision Workshops (ICCVW)*, pages 343–350.

I have also collaborated with other researchers of our group in the integration of different modeling methods to work together for next publication:

- [Murillo et al., 2012] Murillo, A. C., Gutiérrez-Gómez, D., Rituerto, A., Puig, L., and Guerrero, J.J. (2012). Wearable omnidirectional vision system for personal localization and guidance. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 8–14.

Modeling for metric localization

Chapter 2

Omnidirectional vision for visual SLAM

The SLAM (Simultaneous Localization and Mapping) problem is one of the essential challenges for robotics. This problem can be solved using computer vision, usually by integrating camera measurements in a Kalman Filter [Davison, 2003]. These solutions usually utilize conventional cameras with limited Field of View (FoV) as sensor. This chapter shows how omnidirectional cameras, particularly a catadioptric vision system, can be used to solve the SLAM problem and how they provide a better solution than conventional cameras. The projection of scene points in the image in a catadoptric camera can be modeled with the Spherical Camera Model [Geyer and Daniilidis, 2000], a projection model that unifies central catadioptric and conventional cameras. Section 2.3 details how this projection model can be integrated in a SLAM application based in the Extended Kalman Filter. Section 2.4 shows how the use of a monocular omnidirectional system greatly improves the estimation of the camera trajectory.

2.1 Introduction and related work

Robotics applications can greatly benefit of using omnidirectional cameras as vision systems thanks to the wide Field of View (FoV) of this kind of cameras. Many different types of omnidirectional systems can be found:

- rotating camera, a conventional camera with a rotating mechanic system that allows to capture images all around the scene;
- multicamera systems, compound of many cameras pointing to different parts of the surrounding;
- dioptric systems that use special lenses with wide FoV such as fish-eye lens;
- catadioptric cameras, composed by a curved mirror and a conventional camera and usually found in robotic applications.

This last type of systems has been used in many applications such as surveillance [Scotti et al., 2005], robot navigation [Chahl and Srinivasan, 2000] and localization [Guerrero et al., 2008], tele-presence [Nagahara et al., 2006] or 3D reconstruction [Lhuillier, 2007]. This chapter makes use of a hyper-catadioptric monocular system composed by a conventional camera and an hyperbolic mirror.

A requirement for any autonomous system is to be able to navigate in unknown environments. SLAM [Thrun et al., 2005] techniques try to solve this problem by building a map of the surrounding and localizing the autonomous robot relative to this map using partial measurements of the environment. There exist several types of sensors used to obtain this information from the environment, e.g., radar, laser, sonar or vision. Among the visual SLAM approaches that use conventional vision we can find Davison et al. [Davison et al., 2007] where they present a real-time algorithm to recover 3D trajectory of a camera moving through an unknown scene. Pupilli et al. [Pupilli and Calway, 2006] describe a particle filtering SLAM extension which provides resilience to erratic motion. However just a few SLAM approaches use omnidirectional systems. Gaspar et al. [Gaspar et al., 2000] transform the omnidirectional images into bird's eye images and propose two navigation modalities: Topological navigation and Visual Path following. Huang et al. [Huang and Song, 2008] propose a switching method of visual reference scans that can reduce the computation complexity of the Extended Kalman Filter. Kim et al. [Kim and Chung, 2003] present a SLAM algorithm based on a vision sensor robust to the matching problem with cooperation of structure from motion and stereo vision. Mičušík et al. [Mičušík and Pajdla, 2006] present a method for fully automatic and robust estimation of two-view geometry, auto-calibration, and 3D metric reconstruction from point correspondences in images taken by cameras with wide circular FoV.

When a vision system is used as measurement sensor a projection model is needed. Through this projection model we can obtain geometric information from the images. There are different projection models that describe central catadioptric systems [Kang,

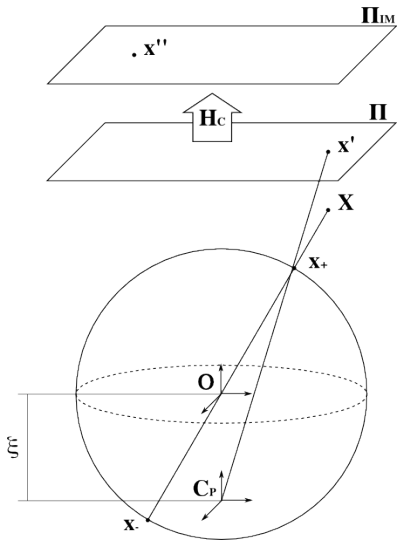


Figure 2.1: Spherical Camera Model Projection

2000], [Svoboda and Pajdla, 2002], [Scaramuzza et al., 2006b], [Toepfer and Ehlgen, 2007], however the most used is the Spherical Camera Model [Geyer and Daniilidis, 2000], [Barreto and Araujo, 2001]. An important property of this model is that models the projection of any central vision system , which allow us to develop a single application that works with both omnidirectional and conventional cameras.

The omnidirectional systems can capture in just one image the 360^o of a scene. That makes possible to keep track of image points in all directions, while conventional systems only keep track of image points inside their narrow FoV. In this chapter we present the integration of the Spherical Camera Model in an EKF-based SLAM. Besides, we perform a comparison of omnidirectional and conventional cameras for SLAM that confirms the advantages of omnidirectional systems over conventional ones.

2.2 The Spherical Camera Model

The work in [Geyer and Daniilidis, 2000] presents a unified projection model for any catadioptric system with a single viewpoint. Model extended later by Barreto et al. in [Barreto and Araujo, 2001]. This projection is known as the Spherical Camera Model and it is widely used to model omnidirectional vision systems.

The projection of a 3D point in the image is divided in three steps (Fig. 2.1). We assume that the world reference system coincides with the camera reference system. The first step is the projection of the scene point $\mathbf{X}$ onto a unit sphere centered in the origin $\mathbf{O}$. There are two intersection points, $\mathbf{x}_+$ and $\mathbf{x}_-$, but just one is physically true. These points are projected to a virtual projection plane π through the virtual projection center

$\mathbf{C}_P = (0,\ 0,\ -\xi)^{\mathsf{T}}$ into point $\mathbf{x}'$. These two steps are coded in one equation:

$$\mathbf{x}' = \hbar(\mathbf{X}) = \begin{pmatrix} x \\ y \\ z + \xi\sqrt{x^2+y^2+z^2} \end{pmatrix}. \tag{2.1}$$

The next step transforms the virtual plane π in the image plane π_{IM} through a homographic transformation H_C.

$$\mathbf{x}'' = \mathsf{H}_C\mathbf{x}' \tag{2.2}$$

$$\mathsf{H}_C = \mathsf{K}_C\mathsf{R}\mathsf{M}_C \tag{2.3}$$

$$\mathsf{K}_C = \begin{pmatrix} f_x & s_{skew} & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.4}$$

$$\mathsf{M}_C = \begin{pmatrix} \psi - \xi & 0 & 0 \\ 0 & \xi - \psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.5}$$

where K_C includes the camera parameters, M_C includes the system parameters [Barreto and Araujo, 2001] and R is the rotation matrix of the camera with respect to the mirror. Finally, the image coordinates are estimated by the next function:

$$\mathbf{u} = \begin{pmatrix} u \\ v \end{pmatrix} = f_u(\mathbf{x}'') = \begin{pmatrix} \frac{x''}{z''} \\ \frac{y''}{z''} \end{pmatrix} \tag{2.6}$$

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

To estimate the 3D ray generated by an image point, the inverse projection model is used. From the point image coordinates, $\mathbf{u} = (u,\ v)^{\mathsf{T}}$, being $\mathbf{x}'' = (u,\ v,\ 1)^{\mathsf{T}}$. The equations of the inverse projection model are:

$$\mathbf{x}' = {\mathsf{H}_C}^{-1}\mathbf{x}'' \tag{2.7}$$

$$\mathbf{x} = \hbar(\mathbf{x}')^{-1} = \begin{pmatrix} x' \\ y' \\ z' - \frac{\xi(x'^2+y'^2+z'^2)}{\xi z' + \chi} \end{pmatrix} \tag{2.8}$$

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

Since the Spherical Camera Model deals with catadioptric systems and conventional cameras, an SLAM application including this model will be able to consider both vision systems just modifying the camera calibration parameters.

2.3 SLAM: Simultaneous Localization and Mapping

By using SLAM techniques a robot can build a map of the environment keeping track of its pose in the map. In Visual SLAM these measurements are usually characteristic points in the images.

The most used SLAM algorithms are based on the Kalman Filter, a filter that predicts the state of linear systems. As the geometry imposes non-linear relations, the Extended Kalman Filter (EKF) [Thrun et al., 2005] is used. The EKF linearize non-linear functions by approximating them to its first order Taylor series. The EKF is divided in two phases:

- *Prediction*, the new state of the system is estimated from the previous time step state through the motion model.
- *Update*, uses the measurements of the environment to improve the new state prediction.

In EKF we assume that every state variable has a Gaussian distribution and it is represented by its mean and its covariance.

The system state in our EKF algorithm is coded in $\mathbf{x}$,

$$\mathbf{x} = (\underbrace{\mathbf{r},\ \mathbf{q},\ \mathbf{V},\ \omega}_{\text{Camera state}}\ ,\ \underbrace{x_i,\ y_i,\ z_i,\ \theta_i,\ \phi_i,\ \rho_i}_{\text{3D points}}\ ...)^{\mathsf{T}} \tag{2.9}$$

where $\mathbf{r}_{(3\times1)}$ is the camera pose, $\mathbf{q}_{(4\times1)}$ is the quaternion of its orientation and $\mathbf{V}_{(3\times1)}$ and $\omega_{(3\times1)}$ are its linear and angular velocities, respectively. The scene points are coded by inverse depth parameterization [Civera et al., 2008], $(x_i,\ y_i,\ z_i)$ is the pose of the camera the first time it observes the point i, θ_i and ϕ_i are the angles that determine the corresponding 3D ray, and ρ_i is the inverse depth of the point. As correlation must be considered, if $\mathbf{x}$ has n dimensions the state covariance matrix C is a squared $n \times n$ matrix.

2.3.1 Image key-points matching

An essential step of the *Update* phase of the EKF is the measurement matching. We use SIFT [Lowe, 2004] points to detect and match image features. The SIFT descriptor is orientation and scale invariant, required properties for multiview matching. The matching is made in two steps for each new image. First, we match these 3D points with the new image features. The matching only uses the 3D points which were predicted to be inside the FoV of the current image. An 3D point is matched to a new image feature if the distance between their descriptors multiplied by a threshold is not greater than the distance from the 3D point descriptor to all the other new image features descriptor. The value of this threshold is 1.5. In second step the matched point must be inside the 95% confidence region predicted by the SLAM algorithm to be considered valid.

2.3.2 The Spherical Camera Model for the EKF

The EKF algorithm requires the first derivative of the measurement equation. In order to adapt the Spherical Camera Model to the EKF algorithm we have formulated the model derivatives, for the direct and the inverse projection.

Jacobian of the Spherical Camera Model direct projection

Trough the Jacobian of the model direct projection, we can estimate the covariance of the image projection of a 3D point given his 3D covariance.

$$\mathsf{J} = \mathsf{J}_{\mathsf{f_u}} \mathsf{H}_C \mathsf{J}_\hbar \tag{2.10}$$

$$\mathsf{J}_{\mathsf{f_u}} = \begin{pmatrix} \frac{1}{z''} & 0 & -\frac{x''}{z''^2} \\ 0 & \frac{1}{z''} & -\frac{y''}{z''^2} \end{pmatrix} \tag{2.11}$$

$$\mathsf{J}_\hbar = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{\xi x}{\rho} & \frac{\xi y}{\rho} & 1 + \frac{\xi z}{\rho} \end{pmatrix} \tag{2.12}$$

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

Jacobian of the Spherical Camera Model inverse projection

To initialize new features we require a way to estimate the covariance of the new feature from the covariance of the image point. This estimation can be done through the Jacobian of the Spherical Camera Model inverse projection.

$$\frac{\partial \mathbf{x}}{\partial \mathbf{x}''} = \frac{\partial \mathbf{x}}{\partial \mathbf{x}'} \frac{\partial \mathbf{x}'}{\partial \mathbf{x}''} \tag{2.13}$$

$$\frac{\partial \mathbf{x}'}{\partial \mathbf{x}''} = \mathsf{H_C}^{-1} \tag{2.14}$$

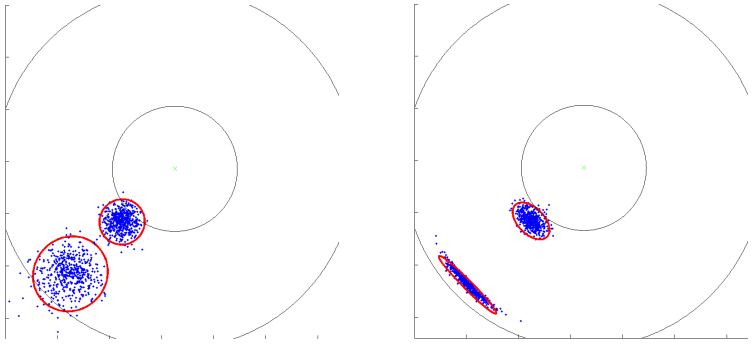
$$\frac{\partial \mathbf{x}}{\partial \mathbf{x}'} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{\partial z}{\partial x'} & \frac{\partial z}{\partial y'} & \frac{\partial z}{\partial z'} \end{pmatrix} \tag{2.15}$$

$$\frac{\partial z}{\partial x'} = -\frac{2\xi x'(\xi z' + \chi) - \frac{(1-\xi^2)\xi x' \rho^2}{\chi}}{(\xi z' + \chi)^2} \tag{2.16}$$

$$\frac{\partial z}{\partial y'} = -\frac{2\xi y'(\xi z' + \chi) - \frac{(1-\xi^2)\xi y' \rho^2}{\chi}}{(\xi z' + \chi)^2} \tag{2.17}$$

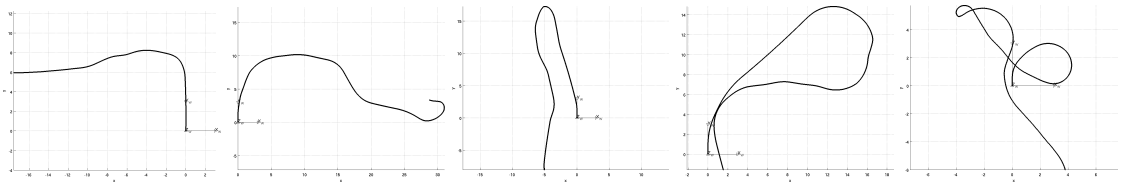
$$\frac{\partial z}{\partial z'} = -\frac{2\xi z'(\xi z' + \chi) - (\xi + \frac{z'}{\chi})\xi \rho^2}{(\xi z' + \chi)^2} \tag{2.18}$$

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



(a) (b)

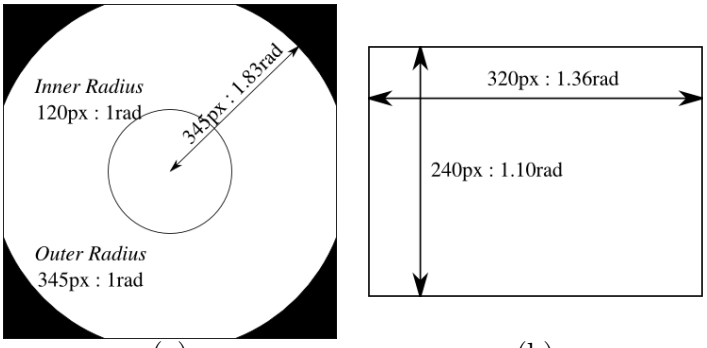
Figure 2.2: Simulation of the projection of a cloud of 3D points in a omnidirectional image. The black asterisks are the projection of the 3D central points, and the blue points are the projection of the cloud points around them. The red ellipses are the 95% confidence zone of the point according to the linearized model in two cases (a) and (b).



Trajectory 1 Trajectory 2 Trajectory 3 Trajectory 4 Trajectory 5

Figure 2.3: Selected trajectories

To verify the developed Jacobians and to show the size and shape of the uncertainty in the omnidirectional images we perform the next simulation. We simulate central 3D points and a Gaussian cloud of 500 points around each of them. After that we project those points through the Spherical Camera Model in a omnidirectional image. Two cases are considered in Fig. 2.2. In both cases two 3D points situated at the same distance from the camera are considered. In Fig. 2.2(a) the points have a 3D uncertainty, so the cloud of points form a sphere. In Fig. 2.2(b) a 2D uncertainty in a horizontal plane perpendicular to the optical axis is considered, to simulate points extracted from the ground plane. In both cases the uncertainty near the center of the image is smaller than in the outer part of the image because of the non-homogeneous resolution of the omnidirectional image. In the case of the 2D uncertainty the width of the ellipse in the radial direction is smaller than in tangential direction, specially in the outer part of the image.



(a) (b)

Figure 2.4: Angular resolution of both the omnidirectional, (a), and the conventional, (b) cameras.

Table 2.1: Mean of the orientation error in radians for different values of the framerate, Trajectory 4

Framerate	Omnidirectional camera	Conventional camera
30 FPS	–	0.22
15 FPS	0.07	0.63
10 FPS	0.07	0.26
5 FPS	0.10	0.54

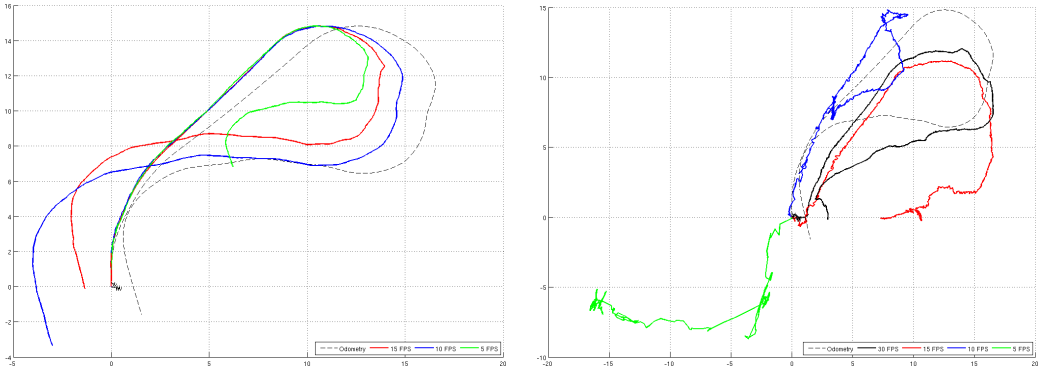
2.4 Comparison of omnidirectional and conventional cameras for visual SLAM

2.4.1 Experimental settings

The SLAM approach used for the next experiments is based on a basic visual SLAM[1]. This basic code uses the EKF algorithm and inverse depth parametrization [Civera et al., 2008]. We have integrated the Spherical Camera Model in the measurement equation and SIFT features are used as image points. A bucketing is also used to allocate features all over the image. We use two egocentric image sequences provided by The Rawseeds Project (see Appendix A). They have been acquired by a robot with a hyper-catadioptric camera and a conventional one. The ground truth of the robot trajectory is given by an improved odometry. To better analyze the behavior of our approach we have split the whole trajectory in five partial trajectories (Fig. 2.3) where some particular difficulties appear. All of these trajectories have been acquired outdoor with natural lighting and include people and vehicles moving.

The resolution of the images is 640x640 pixels for the omnidirectional camera and

[1]Code: `http://www.robots.ox.ac.uk/~SSS06/Website/Practicals/~SSS06.Prac2.MonocularSLAM.tar.gz`



(a) (b)

Figure 2.5: Framerate analysis of the trajectory 4 for the omnidirectiona, (a), and conventional, (b), systems.

320x240 pixels for the conventional one. The values of the angular resolution of both cameras (Fig. 2.4) are 235.05 $\frac{pixels}{rad}$ and 218.26 $\frac{pixels}{rad}$ for the conventional camera in horizontal and vertical direction respectively, and a mean of 232 $\frac{pixels}{rad}$ in tangential direction and 188.25 $\frac{pixels}{rad}$ in radial direction for the omnidirectional camera. We consider both vision sytems equivalent in angular resolution.

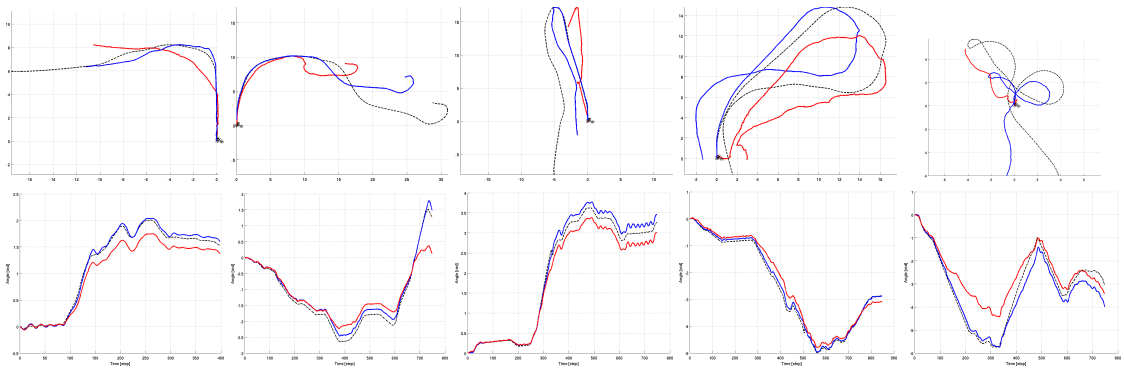
The calibration parameters for both systems are also provided by The Rawseeds Project. For the conventional camera we add the Spherical Camera Model parameter $\xi = 0$ to the original calibration. For the hyper-catadioptric system we have performed a new calibration of the system using the approach in [Bastanlar et al., 2008] for better results.

Initially we tested the effect of the framerate for both visual systems. An example of the estimated trajectories are shown in Fig. 2.5. The Mean Absolute Error (MAE) of the orientation for each experiment is shown in Table 2.1. The best results for both systems are the ones obtained with the maximum framerate. From now we use a framerate of 15 FPS for omnidirectional camera and 30 FPS for conventional camera in order to obtain the best results.

2.4.2 Qualitative analysis of the result for each trajectory

Using monocular vision we cannot estimate the real scale of the resulting trajectory. We decide to use the orientation angle to analyze the SLAM results since camera translation and 3D coordinates of points are affected by unknown scale. The estimated results are compared to those given by the odometry. In order to plot the real and the estimated trajectories in just one figure we compute a scale coefficient for each experiment. Fig. 2.6 shows the trajectory and orientation results for each tested trajectory.

At first sight the results corresponding to the omnidirectional camera are better than the conventional camera results. The estimation follows the real movement, but it does not fit the real dimension of the real trajectory. That is clear for the fifth section where



Trajectory 1 Trajectory 2 Trajectory 3 Trajectory 4 Trajectory 5

Figure 2.6: The upper row shows the x-y trajectory results of the experimentation and the lower row shows the orientation results for the five selected trajectories. The blue and red line are the estimated data with the omnidirectional camera and the conventional one, respectively. The black dashed line is the real data given by the odometry.

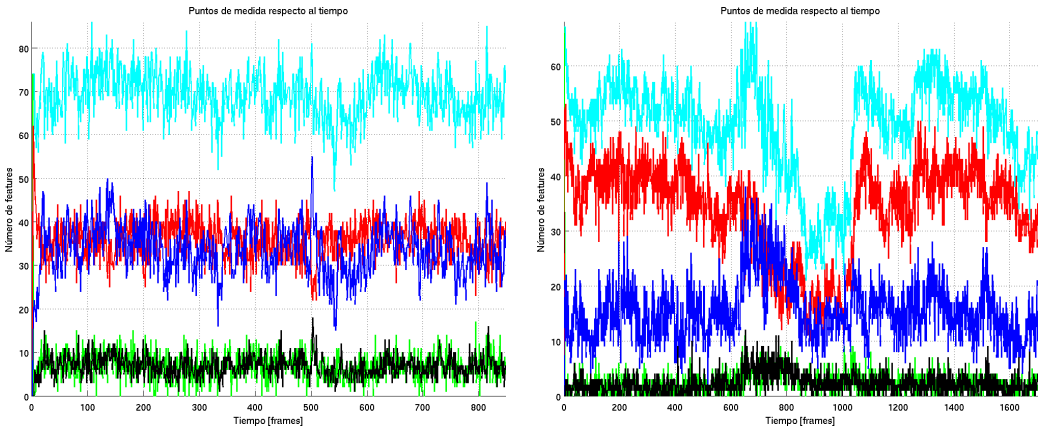
Table 2.2: Mean of the orientation error in radians

Trajectory	Omnidirectional camera	Conventional camera
1	0.04	0.16
2	0.12	0.26
3	0.11	0.15
4	0.07	0.22
5	0.29	0.62

the omnidirectional estimated trajectory takes all the turns of the real trajectory, but the distances are not the same. The conventional camera estimation for this case does not follow the real trajectory, however the estimation of the orientation is similar to the omnidirectional one.

For the third trajectory the scale drifts along the sequence. The estimated trajectory with the omnidirectional camera goes up from the start point following the real trajectory, but after the 180^o turn the estimation travels a shorter distance than the real trajectory. The scale coefficient has been computed for the upper point of the trajectory but it changes after this turn.

The values of the mean absolute error for the orientation are shown in Table 2.2. The omnidirectional camera gives much better orientation estimation than using conventional camera, even using double number of frames per second with the conventional camera.



(a) (b)

Figure 2.7: Features history for the trajectory 4. The cyan line is the number of active points in the state vector. The red line is the number of features matched and the blue one the non-matched features. The green line is the number of initialized features and the black line the number of removed features in each frame. (a) using omnidirectional camera, and (b) using conventional camera.

2.4.3 Influence of the Field of View in feature history

To study the advantages of the FoV of the omnidirectional cameras over the conventional cameras we analyze the features history of the SLAM results. Fig. 2.7 shows the features history over the time corresponding to trajectory 4. We can observe how the number of active features decreases significantly between the steps 600 and 800. The number of features removed from the state vector in this interval increases. Our SLAM approach removes features after 4 consecutive steps of not being matched. If we look at the omnidirectional camera feature history we observe that for the correspondent interval, 300 to 400 frame steps, the number of active features does not decrease. To better analyze what is happening here we study the particular images, Fig. 2.8. In the conventional images we see how the camera approximates to an untextured wall that hides the further houses and trees. The visible floor and sky are also untextured. Previous features disappear behind the wall, while the untextured scene makes hard to initialize new ones. The omnidirectional images do not suffer of this problem, it also gets closer to the wall, but the great FoV allows to keeps track of features behind the robot.

2.4.4 Analysis of the image bucketing

The bucketing is used to initialize features all over the image, avoiding the problem of having most of the features features concentrated in a small area of the image. We test two bucketing alternatives for each vision system, a polar bucketing and a rectangular one. Fig. 2.9 shows the two bucketing alternatives tested for both vision systems. In Table 2.3 shows the mean and maximum values of features life time for each trajectory

Table 2.3: Mean (μ) and maximum values of life of the features in seconds

(a) Omnidirectional Camera

	Traj. 1		Traj. 2		Traj. 3		Traj. 4		Traj. 5		All
	μ	max	μ	max	μ	max	μ	max	μ	max	μ
Polar buck.	0.69	10.53	0.70	9.40	0.83	11.47	0.74	16.00	0.80	16.07	0.75
Rect. buck.	0.67	10.73	0.72	12.40	0.85	10.93	0.72	12.00	0.77	12.27	0.75

(b) Conventional Camera

	Traj. 1		Traj. 2		Traj. 3		Traj. 4		Traj. 5		All
	μ	max	μ	max	μ	max	μ	max	μ	max	μ
Polar buck.	1.10	16.80	0.84	14.33	0.96	17.83	0.78	14.73	0.60	8.37	0.86
Rect. buck.	0.69	17.77	0.52	14.10	0.60	17.10	0.55	20.27	0.48	8.00	0.57

using both bucketing systems.

It has been argued that omnidirectional vision can maintain tracked points for a longer time due to its wide FoV. This happens with the rectangular bucketing where the mean feature life time is bigger than for the conventional one. However, with the polar bucketing the mean life for the conventional camera is bigger than for the omnidirectional system. The reason is that with a circular bucketing for conventional vision there are a lot of features in the center of the image which remains visible while the camera moves in the direction of the focal axis. In the fifth trajectory the camera takes many turns so it does not move in the direction of the focal axis which explains the differences. This makes hard to the conventional camera to track the features, while the omnidirectional camera can keep track of the features all over the scene. However we have tested that a bigger mean life of the features does not make a better SLAM solution.

2.4.5 Scene reconstruction

We do not have ground truth for the 3D structure of the scene. To illustrate these results, Fig. 2.10 shows the cenital view of the reconstruction of the scenes for the trajectories 3 and 4. The features showed are those with a life bigger than 20 frames. We can observe how the features form a straight line parallel to the trajectory in its right when the omnidirectional camera is used. This line corresponds to a wall present in the real scene. The conventional camera features reconstruction does not form that wall so clearly. Similar differences can be observed for the fourth trajectory. The 3D reconstruction using the omnidirectional camera is much better than the other.

2.4.6 Analysis of the vertical drift

Our SLAM approach estimates the 3D motion of the camera, but the camera moves on a planar ground, so the vertical coordinate should be zero. Both cameras have been calibrated by the Rawseeds Project to be aligned with the acquisition platform axis. The MAE of the estimation of z coordinate of the camera motion is shown in Table 2.4. In

Table 2.4: Mean of the absolute error of the estimation of the vertical coordinate z in meters

Trajectory	Omnidirectional camera	Conventional camera
1	0.18	0.19
2	0.26	0.34
3	0.18	0.44
4	0.11	0.34
5	0.10	0.38

all the experiments the estimation using the omnidirectional camera is much better than using the conventional one.

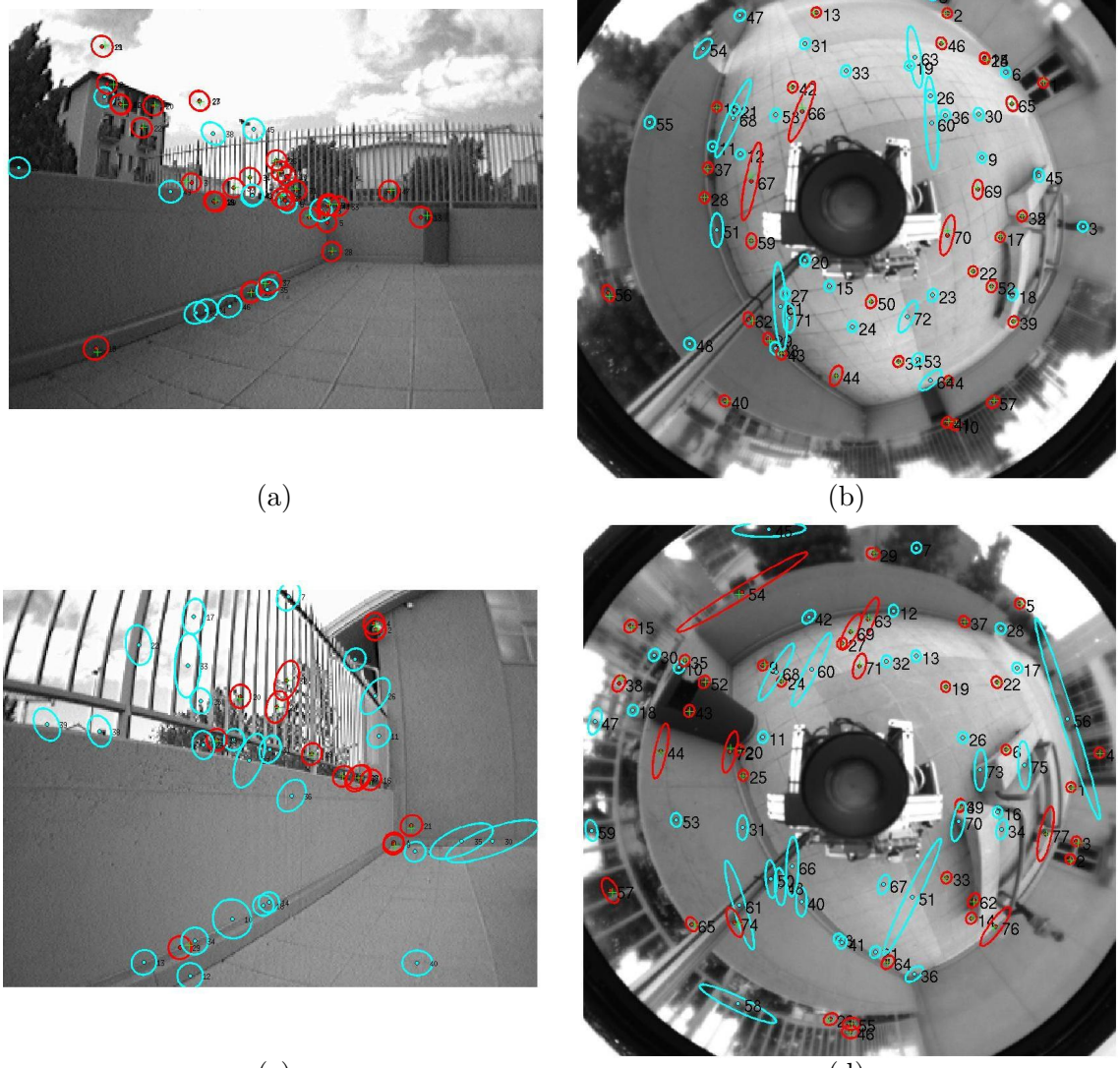
2.5 Conclusions

This section has shown how the Spherical Camera Model can be integrated in an EKF-based SLAM application. The Spherical Camera Model unifies the projection of vision central systems. Its integration in a SLAM application allows to use conventional and omnidirectional central cameras just varying the camera calibration. The different parameters of the integration have been tested and evaluated with real images.

A comparison of the performance of the SLAM application using a catadioptric system and a conventional camera has shown how omnidirectional cameras produce better estimations of trajectory and orientation than with conventional cameras. As well as better 3D reconstructions of the scene.

Conventional camera

Omnidirectional camera



(a) (b)

(c) (d)

Figure 2.8: Frames of interest of the fourth trajectory execution. The cyan ellipses correspond to the confidence zone of not matched features, and the red ones correspond to the matched features. (a) and (b) correspond to the initial image of the analyzed zone for the conventional and the omnidirectional system, respectively. The (c) and (d) images are the last frames for this analyzed zone for both systems.

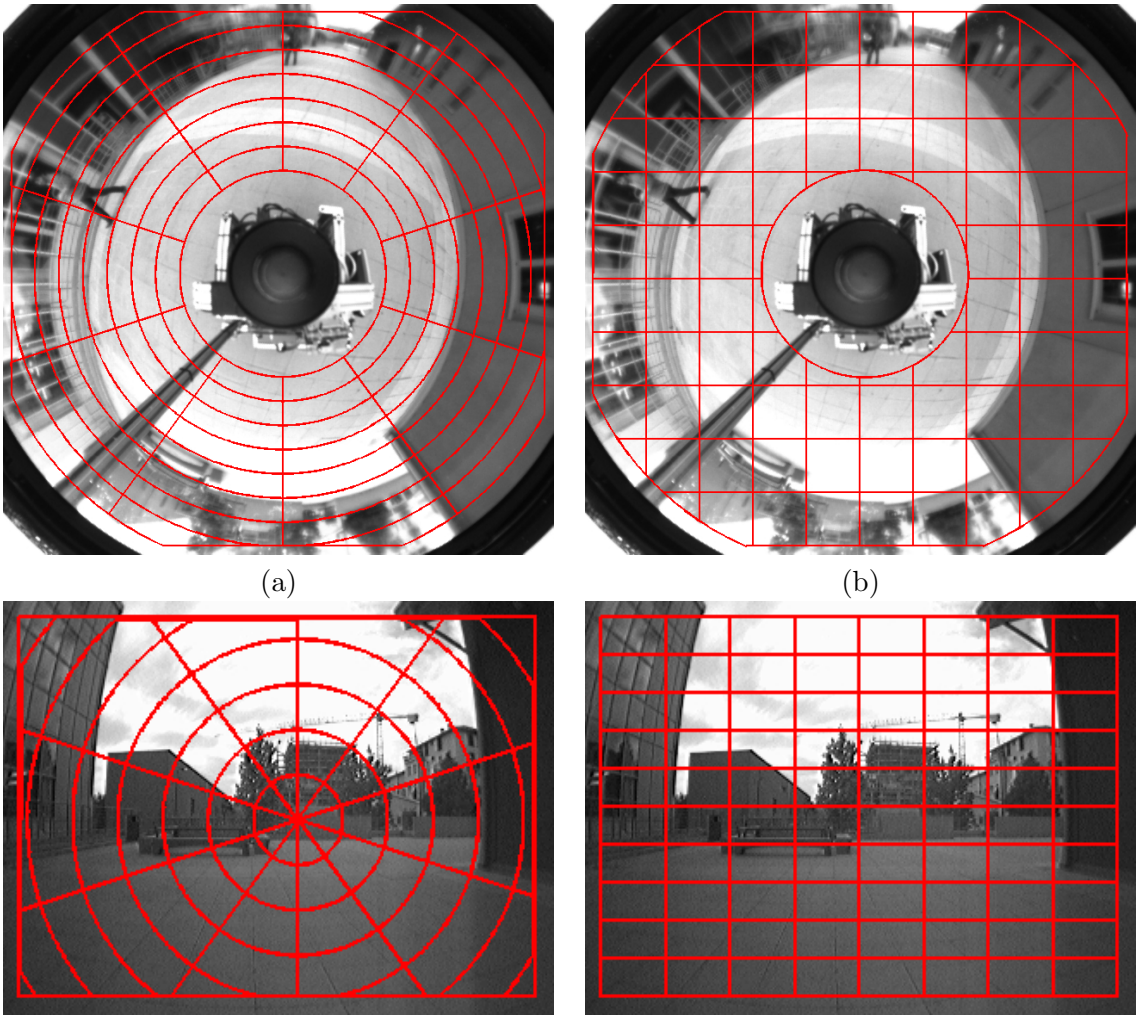


Figure 2.9: Bucketing systems used for the omnidirectional and the conventional camera. Polar, (a), and rectangular, (b), bucketing for the omnidirectional camera. For the conventional images (c) and (d) shows the polar and rectangular bucketing, respectively.

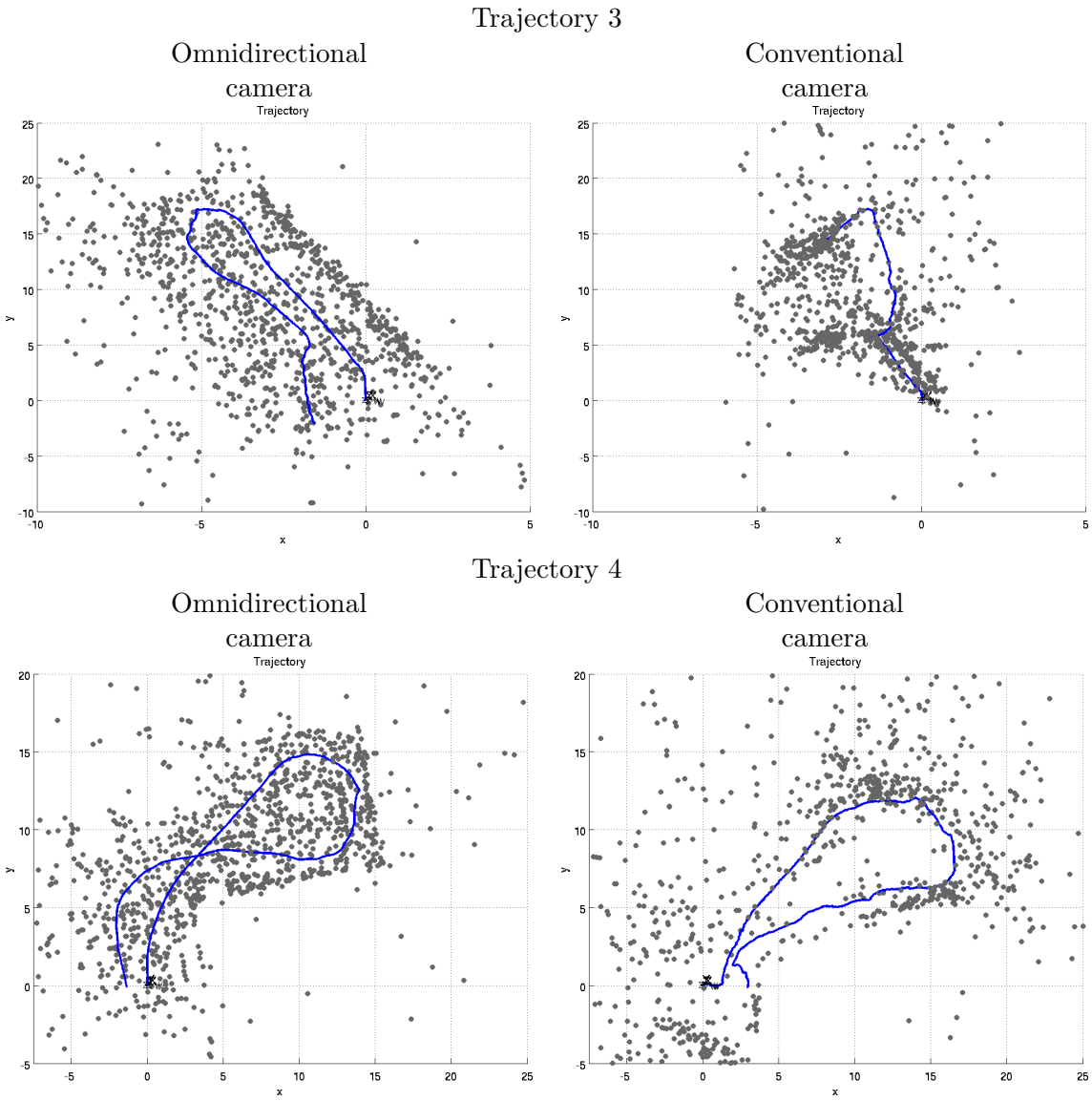


Figure 2.10: Cenital view of the 3D reconstruction along trajectories 3 and 4. The blue line is the estimated trajectory and the gray points are estimated 3D points using the omnidirectional and conventional cameras.

Enhanced topological models

Visual topological models segment the environment into nodes according to their different appearances. Current research on appearance based modeling goes towards richer semantic representations of the environment, which may allow autonomous systems to perform higher level tasks and provide better human-robot interaction. For this purpose, omnidirectional vision systems are of particular interest because they allow to have more compact and efficient representation of the environment. Next, we study how to augment indoor topological modeling using omnidirectional vision. This chapter studies how to augment indoor topological modeling using omnidirectional vision. Chapter 3 presents novel ideas to augment the semantic information of a typical indoor topological model: we pay special attention to the labels of the different types of transitions between places, and propose a simple way to include this semantic information to build a topological map, as part of the criteria to segment the environment. This proposal is built on efficient catadioptric image representation based on the Gist descriptor, which is used to classify the acquired views into types of indoor regions.

Chapter 4 proposes a novel line-based image global descriptor that encloses the structure of the scene observed. This descriptor is designed with omnidirectional imagery in mind, where observed lines are longer than in conventional images. The new descriptor can handle images acquired with different omnidirectional sensors (omnidirectional cameras, panoramic images).

Chapter 3

Place labeling for indoor topological modeling with omnidirectional vision

This chapter presents a new omnidirectional vision based scene labeling approach for augmented indoor topological mapping. Omnidirectional vision systems are of particular interest because they allow us to have more compact and efficient representation of the environment. Our proposal includes novel ideas in order to augment the semantic information of a typical indoor topological map: we pay special attention to the semantic labels of the different types of transitions between places, and propose a simple way to include this semantic information to build a topological map, as part of the criteria to segment the environment. This work is built on efficient catadioptric image representation based on the Gist descriptor, which is used to classify the acquired views into types of indoor regions. The basic types of indoor regions considered are *Place* and *Transition*, divided into more specific subclasses, e.g., *Transition* into *Door*, *Stairs* and *Elevator*. Besides using the result of this labeling, the proposed mapping approach includes a probabilistic model to account for spatio-temporal consistency using a Hidden Markov Model (HMM). All the proposed ideas have been evaluated in a new indoor dataset developed for this work. This dataset has been acquired with our wearable catadioptric vision system, showing promising results in a realistic prototype.

3.1 Introduction

For most autonomous tasks, one of the initial steps consist on obtaining a suitable representation of the environment. In order to obtain it, the system interprets the data acquired with different sensors on-line or in exploration phases to build different types of models depending on the tasks to be performed. Focusing on vision sensors, this modeling consists of arranging the acquired images into a visual memory or reference map. Data should be organized efficiently but more importantly, in a way as useful as

possible to be used later. In many cases, big and accurate metric maps are not necessary or not informative enough, therefore higher abstraction level maps can be built, such as topological or object-based maps, such as [Tapus and Siegwart, 2005, Vasuvedan et al., 2007, Zender et al., 2008, Topp and Christensen, 2010].

The general goal of this work is to build a useful semantic-topological map for indoor environments using a wearable catadioptric vision system for human assistance. We propose how to include interesting semantic information on indoor topological models. In particular we design a simple approach to segment the environment into semantically meaningful clusters using omnidirectional images acquired with our wearable system. We represent the catadioptric images following the approach described in [Murillo et al., 2010], which is based on an adaptation of the global Gist descriptor [Oliva and Torralba, 2006] to omnidirectional images. Our long term goal is to merge the presented semantic model with a set of small metric maps of each topological region, which could be obtained with standard visual odometry or slam algorithms [Rituerto et al., 2010b]. In this work we focus on a new scene classification method for topological mapping. Our proposal includes the following two novel ideas with regard to other related works.

First, an improved criteria to define environment clusters. On-line topological mapping approaches usually segment the environment regions by evaluating the similarity within consecutive images and establishing different criteria to decide when and where to segment the trajectory. We define how to easily include the labeling from our semantic indoor region classifier as part of the criteria to organize the topology of the environment. This classifier is based on a model previously built from a few given examples of the different classes to be recognized. We find that most of the approaches for semantic indoor scene labeling try to label types of *Places* [Stachniss et al., 2005, Saurer et al., 2010, Pronobis et al., 2010b, Ranganathan, 2010].

Second, additionally to the types of *Places*, we perform a detailed analysis of the semantic information included in the types of *Transitions* (such as door, elevator or stairs) between these *Places*. This information can be of great interest for autonomous systems working indoors, since depending on the transitions we may be or not be able to traverse from one *Place* to another. Knowing the type of transition may allow us to choose a suitable robot team member to go to a particular destination, or give appropiate instructions in case of human assistance systems.

Two other interesting properties of our method, that are not novel themselves but their integration is essential in our proposal, are the following: first, the fact of using only global descriptors, with the corresponding improvement in efficiency with regard to the use of local features; second, the inclusion of a probabilistic model to keep the spatio-temporal consistency of the labeling along the trajectory. In spite of the simplicity of the image representation, the proposal gets to partition the environment into semantic meaningful areas for humans, as it can be seen later in the experimental validation, using the presented catadioptric dataset.

Additionally to our approach, this chapter presents a new indoor dataset, used to evaluate our proposals. This dataset has been acquired indoors with a wearable system composed by an omnidirectional camera, an IMU and a GPS. The use of wearable sys-

tems is mostly oriented to create human assistance applications, and adds new difficulties to the work.

3.2 Related work

Topological modeling of the environment is a subject already studied for long [Thrun and Bucken, 1996,Santos-Victor et al., 1999]. Initially, these models presented huge possibilities due to their lower computational requirements with regard to accurate metric maps. More recently, these models have gained interest due to the possibilities of augmenting them with semantic concepts [Nüchter and Hertzberg, 2008], such as information about places [Tapus and Siegwart, 2005] and/or objects [Vasuvedan et al., 2007]. Topological maps are many times built on top of a hierarchy of different map levels [Kuipers, 2000], e.g., a global topological map that connects smaller local metric maps [Tomatis et al., 2003]. An extensively used solution to achieve different efficiency and accuracy results at the different levels is to use global and local image features through the different steps of the hierarchy [Murillo et al., 2007].

Augmenting topological maps with semantic information makes them more suitable for human-robot interaction [Zender et al., 2008,Topp and Christensen, 2010] and allows us to achieve more complicated goals [Galindo et al., 2008]. Semantic mapping provides new opportunities to increase the autonomy and reasoning skills of intelligent systems, both for outdoor and indoor applications.

In outdoor settings, many of the recent and impressive approaches are achieved by combining multi-sensor information, typically vision and laser sensors [Cummins and Newman, 2009, Douillard et al., 2011], to build topological models that include place or object recognition information. In the work [Cummins and Newman, 2009], which deals with place recognition, the authors present an approach for appearance based mapping using extremely large datasets (1000 km) that efficiently recognizes previously visited places. The work in [Douillard et al., 2011] is focused on objects rather than places, it recognizes and labels objects in large urban environments proposing a Conditional Random Field based framework.

Focusing on the framework of this work, indoor environments, we also find proposals using different types of sensors to interpret semantic information that will be included in a topological map. Initial proposals were typically achieved using range data, to learn a room-doorway-hallway structure indoors in [Martínez Mozos and Burgard, 2006] or [Friedman et al., 2007]. We also find proposals using a combination of range and vision cues, for example in [Zender et al., 2008] they combine place and object recognition in exploration and semantic mapping approach. The work in [Pronobis et al., 2010b] suggests a Support Vector Machine (SVM) scheme that learns how to optimally combine and weight each cue. In [Rottmann et al., 2005] boosting is used to learn a classifier with different place labels, using vision and range sensors.

Proposals only based on vision sensors are closer to our approach. Although they usually provide more detailed labels than only range data approaches, most of these approaches still include semantic labels only regarding Places (e.g., office, corridor,

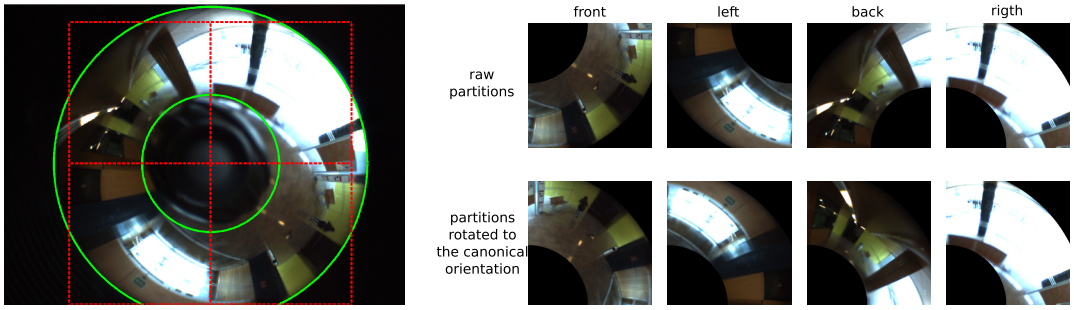
kitchen...) [Pronobis et al., 2010a], considering all transitions as just connections between places. We find different types of approaches that try to classify types of places, with multiple proposals of how to learn the labels to be recognized and how to represent the images.

Regarding how to learn the environment model, some proposals are constantly trained and, sometimes, simultaneously run with human supervision to achieve a representation closer to human concepts [Topp and Christensen, 2010,Nieto-Granda et al., 2010]. Others use weaker human supervision to learn a model from a few initial labeled samples, such as the work in [Saurer et al., 2010]. This approach learns the representation of problematic locations (e.g. images showing only zoomed wall areas, without any information about the actual indoor region) from a few given examples. This helps to detect when those problematic cases occur and to avoid giving incorrect or noisy labeling.

Our augmented topological mapping approach makes use of human supervision, but only in the initial training phase, to provide sample labeled images of the types of indoor scenes of interest. Besides labels for places, our approach includes semantic information about the types of transitions. This is of particular interest for multiple-floor buildings, where depending on who is using the map a transition (elevator, stairs, closed door,...) may be feasible to traverse or not. We find other recent works also paying attention to transitions [Nieuwenhuisen et al., 2010, Ranganathan, 2010]. The first work only detects doors, proposing to dynamically model the environment to react if a transition is suddenly closed. The second work applies the generic label *transition* to all areas that are not detected as being of a known type of place, so no knowledge of the kind of transition is included.

Besides the described augmented mapping approaches, we find additional closely related works regarding the more general problem of place or scene recognition indoors [Quattoni and Torralba, 2009]. This work also points the idea of plenty of indoor areas usually not considered in classification approaches. Among the big set of types they study we can find elevators and stairs. In this work we include all type of indoor regions that can actually be considered as a transition between places (elevators, stairs and areas under doors or under jambs). Another common point with that work is the use of the Gist descriptor [Oliva and Torralba, 2006].

Attending to the image representation, some works propose to work with local features, such as the robust vision-based robot localization using combinations of local features from [Ramisa et al., 2009], or the work in [Viswanathan et al., 2011] that presents an integration of object detection, using local features, and global image properties for place classification. In this work we use global features. Our global image descriptor is based only on the global Gist descriptor, following the ideas initially presented in [Murillo and Kosecka, 2009]. The Gist descriptor was initially presented for classifying outdoor scenes [Oliva and Torralba, 2001] and used in more recent work together with additional cues for indoor scene recognition [Quattoni and Torralba, 2009]. Global descriptors are known to be more efficient and compact, but usually less robust and discriminative, than local features. However, in the current work promising results pointed that this weakness can be compensated to a certain extent by the powerful scene representation contained



(a) (b)

Figure 3.1: (a) Raw image acquired with our catadioptric vision system. The green line shows the mask limits used to avoid the artifacts of the image. The dashed red line defines the limits of the four image parts. (b) Top row shows these four raw parts, while bottom row shows the four parts rotated to the canonical orientation.

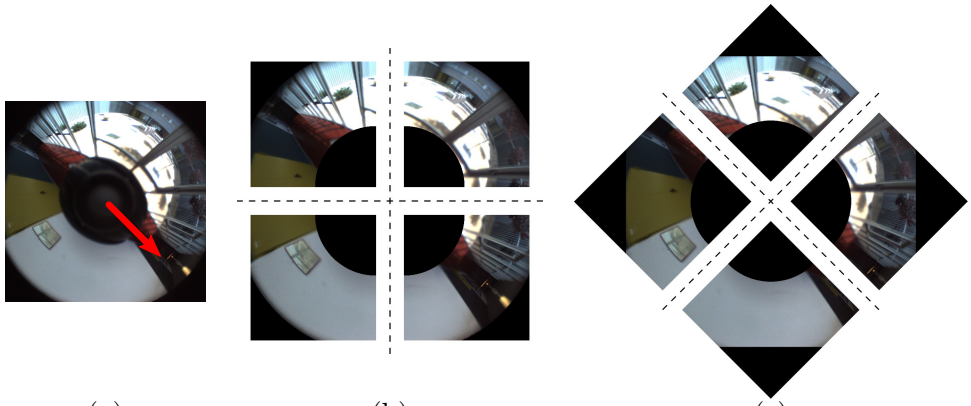
in omnidirectional images.

The use of omnidirectional images is another key characteristic of our proposal. Some proposals take advantage of wide field of view cameras to acquire more compact visual models, e.g., in [Rybski et al., 2003, Puig et al., 2010] panoramic cameras are used for indoor topological map building and [Goedemé et al., 2007] presents an approach for topological mapping and navigation using a catadioptric vision system. We use this second type of images, acquired with a catadioptric vision system, usually smaller and with lower cost that the panoramic cameras. However, these cameras present additional issues to deal with, such as big image distortion, noise and parts of the vision system self-reflected in the views. These issues together with the fact that we are using a wearable system, requires a carefully designed image representation detailed in the following section.

3.3 Image representation and similarity

Visual descriptors that capture image information as a whole are known as global descriptors, while those that capture a specific interest region are called local descriptors. It has been typically shown that local descriptors are more accurate for visual localization than global descriptors, but also have much larger memory and processing requirements [Deselaers et al., 2008]. Therefore, to deal with large quantities of images for tasks where efficiency is an issue and it is not required a detailed analysis of image content, a global representation is preferred.

In this work we use the Gist descriptor [Oliva and Torralba, 2001], a holistic image representation or global image feature. In particular, it is a low dimensional representation of the scene captured in an image which corresponds to the mean response to steerable filters at different scales and orientations computed over 4x4 sub-windows. The descriptor consists of a vector of 320 components for each color band used, so in a



(a) (b) (c)

Figure 3.2: (a) Omnidirectional image acquired with the Wearable OmniCam system partitioned using the two methods analyzed: (b) Direct and (c) Rotated. The red arrow in the first image shows the front direction of the helmet.

RGB image it has 960 components. This global feature was presented and applied as a successful tool for scene recognition, with the big computational saving of bypassing the segmentation and the processing of individual objects or regions. Approaches using this descriptor typically work with squared conventional images, most of the time assuming frontal scene views acquired with the camera focal axis parallel to the ground plane, since the descriptor is not rotation invariant.

In the case of omnidirectional cameras the image contains 360^o degrees field of view around the camera. This presents a problem when facing the same scene with different direction of travel, i.e., same location but camera rotated around the vertical axis. This situation can generate apparently different scene view, although it is just a matter of reorganization (shift), of the scene parts. To handle this problem and try to make our image representation invariant to the camera vertical rotation we split the omnidirectional images in four parts, similar to the method presented in [Murillo et al., 2010]. Each image is split in four parts, each part is rotated to a canonical orientation (Fig. 3.1 shows how this rotation is done), and he Gist descriptor is computed for each part. We need to mask out parts of the image where artifacts appear, mostly produced by the reflection of catadioptric system elements in its own mirror. With this representation, the omnidirectional image Gist $\mathbf{g}$ is composed by four conventional Gist descriptors, one computed for each image part (front, left, back and right): $\mathbf{g} = [g_f, g_l, g_b, g_r]$.

We have analyzed how to split the omnidirectional images. Fig. 3.2 shows two partition possibilities: Direct partition (Fig. 3.2(a)), that just splits the image into four equal squares, and Rotated partition (Fig. 3.2(b)), where the parts are extracted from the 45^o rotated image. Due to the camera orientation, the parts extracted in the Direct partition correspond to the main directions of the scene according to the Manhattan World Assumption. The use of these two partitions is analyzed in detail in subsection 3.3.1.

The similarity between two images using this representation is obtained based on the

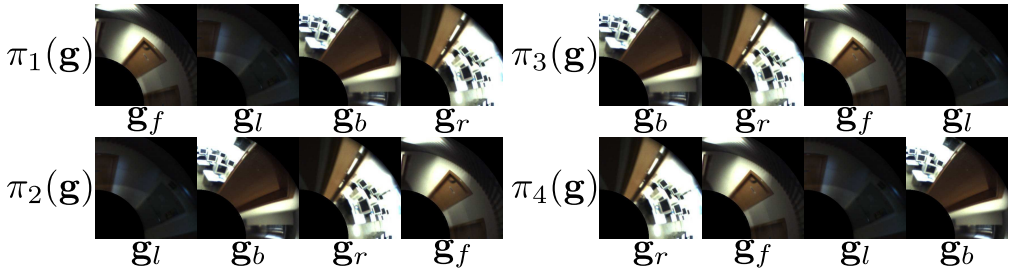


Figure 3.3: Graphic representation of the circular permutations of the Gist descriptor.

Euclidean distance between the descriptors. We compute the minimum distance that can be obtained between one image and the four possible permutations of the four sections of the second image. These permutations correspond with the four possible alignments of the sectors of the image and will hopefully provide us with the best alignment of the two evaluated images. Being $\mathbf{g}$ and $\mathbf{g}'$ the descriptors of two images, the distance between them is:

$$dist(\mathbf{g}, \mathbf{g}') = \min_m (d_e(\mathbf{g}, \pi_m(\mathbf{g}'_{flbr})), \tag{3.1}$$

where $\pi_m(\mathbf{g}'_{flbr})$ is the m^{th} circular permutation of the descriptor $\mathbf{g}'$ component vectors ($m = 1, 2, 3, 4$) and d_e the Euclidean distance between the Gist descriptors of two omnidirectional images. Fig. 3.3 shows the circular permutations in a graphic way.

3.3.1 Rotation invariance analysis

To analyze in detail the rotation invariance issues described above we have performed two experiments.

With the first experiment we want to prove the rotation invariance achieved with this image representation. We get 36 images equally distributed along a 360^o camera rotation movement without translation, around the vertical camera axis. Using the Direct partition we achieve invariance to vertical rotation at angles multiple of 90^o and using both the Direct and the Rotated partition together we get invariance to rotation at angles multiple of 45^o. This rotation invariance is not robust to all kind of movements, but the Manhattan assumption seems a reasonable one to work with man-made environments, where the possible directions of travel on a particular location usually fit these restrictions. Each image from this test set corresponds to a rotation of 10^o with regard to the previous image. We extract the Gist descriptor of all images with the two partitioning methods, so for each image we have two descriptors ($\mathbf{g}_{Direct}$ and $\mathbf{g}_{Rotated}$). We compare the Gist of the Direct partition ($\mathbf{g}_{Direct}$) of the reference 0^o image with both the Direct ($\mathbf{g}_{Direct}$) and the Rotated ($\mathbf{g}_{Rotated}$) Gist descriptors of all the following images. Using a perfect rotation invariant representation all images would get exactly the same descriptor and the distance (3.1) would be 0. Figure 3.4(a) shows the results of this test. The red line represents the distance between $\mathbf{g}_{Direct}$ in the test images and the initial reference image. It shows the higher distance values (less similar images according to

our representation) at rotations of 45^o, 135^o and 225^o; while, as expected, the minimum distances appear at rotations of 0^o, 90^o, 180^o, 270^o and 360^o. The blue line represents the distance between $\mathbf{g}_{Direct}$ in the test images and $\mathbf{g}_{Rotated}$ in the reference image. The black line represents the minimum value of red and blue line. This merged result confirms that using both partition methods we achieve better invariance to rotation, in particular to rotation at angles multiple of 45^o.

Second experiment is designed to show the influence of using one or two of the described partition methods while moving indoors. We have chosen a subsequence of the dataset where the camera moves along a corridor and returns the same way but from opposite direction (180^o rotation). The test consists of comparing the $\mathbf{g}_{Direct}$ of all images against the $\mathbf{g}_{Direct}$ and $\mathbf{g}_{Rotated}$ of the reference image. The image used as reference is the first image of the sequence. Ideally, we would like to observe how the distance between images increases as we get the test image farthest from the initial image. Figure 3.4(b) shows the results of this second experiment. The frames 50 and 240 correspond to frames where the camera is rotating. Points 190 and 305 correspond to the highest Gist distance, points of the trajectory that are almost at the farthest location from the reference. This suggest that the Gist distance variations are bigger due to the translation along the corridor than due to the rotation. The distance $\mathbf{g}_{Direct} - \mathbf{g}_{Rotated}$, blue line, is usually higher than the $\mathbf{g}_{Direct} - \mathbf{g}_{Direct}$, red line. Points where Direct versus Rotated distance is smaller than Direct versus Direct distance correspond to parts of the trajectory where the camera is rotating.

Therefore, as already mentioned, we can see small improvements using the duplicate Gist partition while navigating indoors. Even if it is more robust to rotations (to all angles multiple of 45^o instead of only multiples of 90^o), the increase in the Gist distance due to the camera rotation issues is small compared to those that appear due to translation. Therefore, the experiments in the rest of this work were performed using only the Direct partition method.

3.4 Augmented topological map with semantic labels of indoor scenes

All the steps of our augmented topological mapping approach are detailed now. First, we propose a simple classification to identify basic indoor scene classes of interest (*Places* and *Transitions*) to discover the topology of the environment. Then, we evaluate the classification into more detailed types of scenes and finally integrate it in a proposal for augmented topological map building.

3.4.1 Labeling of *Places* and *Transitions*

Classification of indoor areas into *Places* and *Transitions* is natural and easy for humans when navigating through a building, and they represent the basis to build a topological map of the environment. *Places* are the nodes of the model and *Transitions* correspond to the edges between nodes. The main objective in this part of the work is to develop a

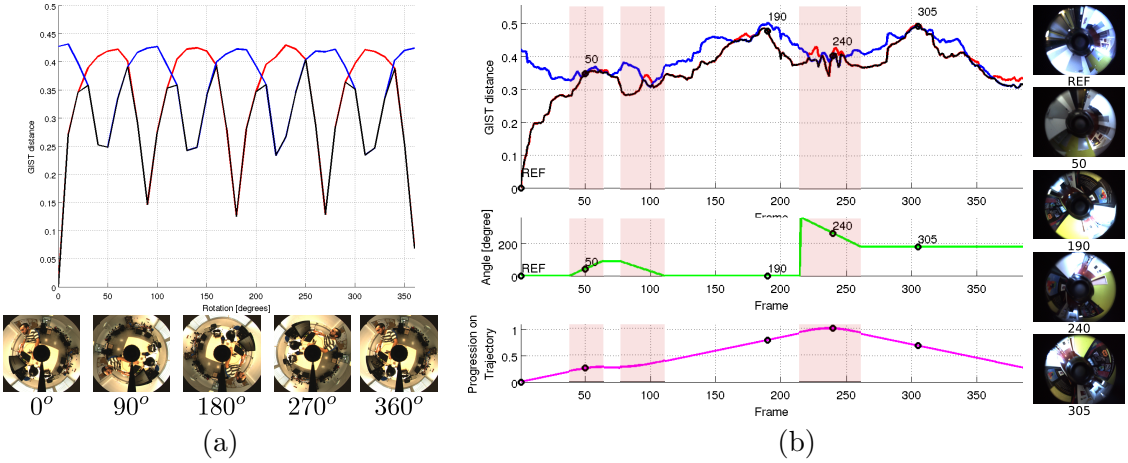


Figure 3.4: Rotation invariance of the image representation proposed (a), and influence of translation vs. rotation (b). The red line shows the distance $\mathbf{g}_{Direct} - \mathbf{g}_{Direct}$ and the blue one the distance $\mathbf{g}_{Direct} - \mathbf{g}_{Rotated}$. The black line shows the minimum value of both distances. The pink areas in (b) mark the parts of the trajectory where the camera is rotating. The omnidirectional images on the right of the figure correspond to the frames marked in the plots. The green line shows the rotation respect to the reference frame. The pink line shows the translation of the camera in the trajectory, the distance from the reference frame.

method to automatically classify the images of a sequence into *Places* and *Transitions* to build an initial map with this information. Additionally we evaluate how to label the scene captured in each omnidirectional image into subclasses: the images classified as *Places* are farthest labeled into corridors and rooms of different sizes (big, medium and small rooms) and the images classified as *Transitions* are labeled as doors, jambs, stairs and elevators (Table 3.1).

Both subclasses provide the model with augmented semantic information, but of particular interest, and different from other approaches, is the fact of analyzing in detail the types of transitions. Indeed, the actions and movements required to traverse each of them are significantly different both for a human or robot navigating the map. For example, climbing stairs is not the same as traversing a door, or the type of movement to be generated may be different in a corridor and in a big room.

We describe next how to perform this classification based on the image representation and similarity evaluation described in previous section.

The Environment Model

A basic step in our method is obtaining the environment model, to use it later as reference to classify new occurrences. This environment model is created using a set of training reference images. It is composed of representative descriptors of each class and subclass given with this training data. In the dataset used in this work, detailed in next section, all

Table 3.1: Classes and subclasses considered in this work.

Classes	Subclasses
Places (P)	Corridor ($P1$) Big Room ($P2$) Medium Room ($P3$) Small Room ($P4$)
Transitions (T)	Door ($T1$) Jamb ($T2$) Stairs ($T3$) Elevator ($T4$)

the images have been manually classified and grouped in clusters of consecutive images belonging to the same semantic class/subclass. To build the model in a systematic way we consider as training data the first n_i clusters of each class. The value n_i for each class i is computed as $\frac{C_i}{4}$, being C_i the number of clusters of class i in the environment. To obtain a more homogeneous sampling, the value n_i is quantized into the following set of values: $n_i \in [1, 2, 5, 10]$. The model of each class, M_i, initially consists of all $\mathbf{g}_{Direct}$ descriptors of the training images. Note that typically *Place* clusters would include more images than *Transition* clusters, since the time spent traversing a corridor is longer than the time spent crossing a door, so more images of that type are acquired. To avoid that this fact leads to unbalanced models towards *Place*, we use a standard k-means method to find the k Gist descriptors that better represent each class. Then, all classes have the same amount of reference data in the model, the k Gist descriptors that correspond to the centroids of the obtained clusters. More formally, the environment model is:

$$M_i = \{M_{i,j} | j = 1..m\} \text{ with } M_{i,j} = \{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{k_{Model}}\} \tag{3.2}$$

where M_i is the model of class i ($i \in [P, T]$), m is the number of subclasses of class i, and $M_{i,j}$ is the model of subclass j from class i that is composed by its k representative descriptors.

Label new occurrences according to the environment model

To label new images, we use a simple nearest neighbor based classifier. To measure the likelihood of a new image being of a particular class, we compute the following likelihood function (3.3) based on the Gist descriptor distance, and assign the maximum likelihood solution as label for the new image.

$$p(I_t | S_t = i) = \frac{K\, e^{\frac{-d_i}{\sigma^2}}}{\sum_{j=P,T} K\, e^{\frac{-d_j}{\sigma^2}}} \tag{3.3}$$

$S_t = i$ is the event of being in an area of class i at time t, when the image I_t is acquired. Then $p(I_t | S_t = i)$ is the likelihood of acquiring image I_t at time t being in an area of

class i. Parameters K and σ^2 are user defined gain and variance respectively, in this work we use $K = 1$ and $\sigma^2 = 0.2$, adjusted after some initial experiments. d_i and d_j are computed comparing the current image Gist with the environment model:

$$d_c = \min_{\mathbf{g}' \in M_c} (dist(\mathbf{g}_t, \mathbf{g}')) \tag{3.4}$$

d_c is the similarity between image I_t and the class c. The Gist descriptor $\mathbf{g}_t$ is the descriptor of the image I_t and $\mathbf{g}'$ is one of the representative descriptors of class c included in the model (M_c). The distance between these descriptors is computed following equation (3.1).

3.4.2 Semantic sequence segmentation with temporal consistency

The complete approach for labeling sequential information is based on the image classifier described in previous subsection. The idea is that this semantic labeling of a sequence can be of great help to obtain more meaningful topological representations of the environment captured in that sequence.

Generally, with catadioptric images, if we only pay attention to consecutive image descriptors distance to decide where to split the topological regions, what we obtain is far from a semantic segmentation that a human would do, containing for example lots of small clusters. This is because even consecutive catadioptric images may present big visual differences, due to big image distortions and image changes. This effect is specially pronounced when objects and scene elements are close to the camera (as it usually happens indoors). Our proposal uses semantic labels as basic criteria to obtain semantic meaningful clusters in the topological model as detailed next.

First, we enclose the classifier described before in a framework that allows us to include spatio-temporal coherence in the model. We expect this coherence to improve the classification on sequential data: if the current image is very likely to belong to a transition area, next image is also likely to be part of it. We model these ideas using a Hidden Markov Model (HMM) following the approach presented in [Angeli et al., 2008]. A HMM is a dynamic Bayesian network that represents a sequence of variables. At each instance of time the state is a random variable which can take one of the just two values: P (*Place*) or T (*Transition*). Let S_t be the random variable that represents the event of being in *Place* or *Transition* area at time t and I_t the image at this time. Then, the problem of detecting the kind of area j being crossed can be formulated as the search of j that satisfies:

$$j = \arg\max_{i \in \{T,P\}} \; p(S_t = i | I_t). \tag{3.5}$$

The posterior probability $p(S_t = i|I_t)$ is the probability of the event $S_t = i$ given the image I_t, which can be decomposed using the Bayes rule and the Markov property:

$$\begin{aligned} p(S_t = i|I_t) =& \alpha \;\; p(I_t|S_t = i)p(S_t = i|I_{t-1}) = \\ =& \alpha \;\; p(I_t|S_t = i) \sum_{j=T,P} p(S_t = i|S_{t-1} = j)p(S_{t-1} = j|I_{t-1}), \end{aligned} \tag{3.6}$$

where α is a normalization term, and the conditional probability $p(I_t|S_t = i)$ is the likelihood function (eq. (3.3)) modeling the likelihood of the current image I_t being of type i. The term $p_(S_t = i|S_{t-1} = j)$ is the state transition probability for observing the event $S_t = i$ given $S_{t-1} = j$, i.e., having an image of type i when previous image was of type j. This term models the probability of all possible changes in the state from time $t-1$ to t. We need to model four possible state transitions: $p(S_t = i|S_{t-1} = j)$, with $i, j \in T, P$. In practice, we set empirically the value of the probabilities of repeating the same event occurred at time $t-1$ in time t, $p(S_t = i|S_{t-1} = i)$, so the rest can be computed as $p(S_t = j|S_{t-1} = i) = 1 - p(S_t = i|S_{t-1} = i)$, with $j \neq i$.

Algorithm 1 Semantic sequence segmentation method.

Input: Omnidirectional image sequence and environment model
Output: Semantic sequence segmentation

```
n = Number of the current cluster
th = Similarity threshold of the minSize filter
M_i = Model of class i, with i ∈ [P, T]
g_n = Gist of the first image of the current cluster n
g_t = Gist of the new image I_t
P_{t-1} = Probabilities at previous step
while not end of sequence do
  // New image I_t
  g_t = OmnidirectionalGist(I_t)
  // Compute similarity with the current cluster
  d = dist(g_t, g_n);
  // Compute probability of being transition or place
  [p_P, p_T] = HMMEnvironmentModel(g_t, M_P, M_T, P_{t-1})
  if p_P > p_T then
    state = P
  else
    state = T
  end if
  if d > th & state ≠ state_{ncluster} then
    CreateNewCluster(I_t, n + 1, state)
    n = n + 1
  else
    IncludeImageInCluster(I_t, n)
  end if
  P_{t-1} = [p_P, p_T]_{t-1}
end while
```

Algorithm 1 details the proposed semantic trajectory segmentation method. For each new image the probability of being *Transition* or *Place* is estimated using the HMM. Consecutive images of the same class are grouped into the same cluster until the criteria to start a new cluster is fired. This criteria is based on the likelihoods estimated from the described HMM, but to prevent the appearance of too small clusters a criteria based on the similarity with the first image of the current cluster (*minSize* filter) is included. If this distance is below the similarity threshold (th) established, the new image is included

in the current cluster, even if classification results according to HMM likelihood would label it as a different class than current images in the cluster. We will see the differences of using one or both of this criterion to build the topological map in next section.

This first step just segments the input sequence into clusters of images labeled as *Places* or *Transitions*, the classification into subclasses is performed next. We try to take advantage of this first level classification as a prior for the more detailed classification into subclasses. Once an image is labeled as *Transition* or *Place*, we look for the subclass with the reference descriptor most similar to the current image. We already know the class assigned to the image, so we only evaluate the subclasses of that class.

Note that at the end of the process, we want to assign a unique class and subclass to all members of each cluster. However, during the process images labeled as different subclasses may had ended up together. We consider this is noise due to the fact that descriptors of some subclasses are pretty close to each other and difficult to separate sometimes (doors and jambs for instance, are hard to distinguish for a human observer as well). Then, to assign the most likely subclass label to the whole cluster, we compute the mode of the subclass label assigned to each image in the cluster.

3.5 Experiments

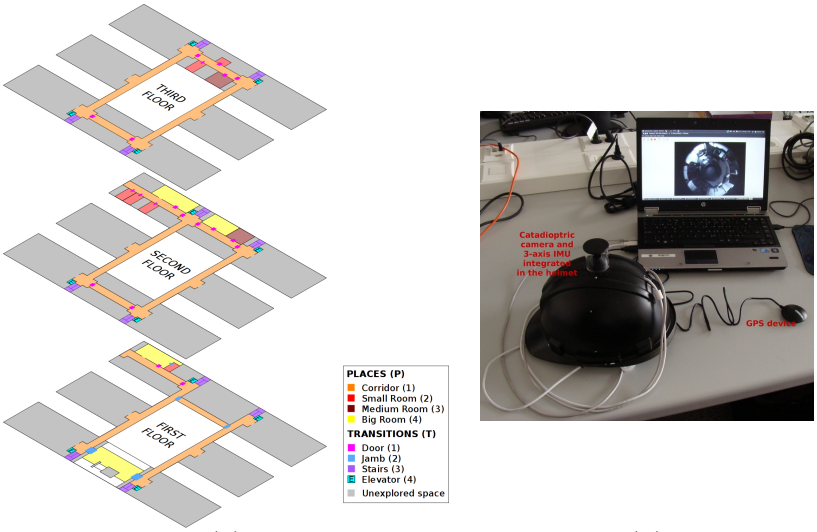
Now we present the new dataset acquired as part of this work and the results of the experimental validation of our proposed method.

3.5.1 The Wearable OmniCam Dataset

The catadioptric image dataset presented in this work has been acquired with our Wearable OmniCam acquisition system. This system includes a small hyper-catadioptric camera mounted on the top of a helmet (Fig. 3.5(a)), a 3-axis IMU (compass, gyroscope and accelerometer) and a GPS device. The three sensors are synchronized and the camera has been calibrated using the approach described in [Puig et al., 2011]. However, the presented data has been acquired indoors, so GPS is deactivated. IMU data is also not used in this work, that is a purely vision based approach, but could be used for future works and included in the published data.

The use of wearable sensing is mainly intended for applications of human assistance. There are a lot of sensors used in different ways to help persons: GPS for localization and guidance, IMU for movement supervision, cameras for object or place recognition, range sensors for obstacle avoidance. In this work we have focused in omnidirectional vision as a wearable system, so we find some problems that do not exist when using omnidirectional cameras for robots. In our case, placing the camera in a helmet, make us face the head movements of a person walking. Up to our our knowledge this is the first dataset published form indoor environments with a wearable omnidirectional camera.

The dataset acquisition has been performed inside one building at our Campus at the University of Zaragoza, Spain. The building has three floors and includes areas of different types: corridors, research laboratories, offices, classes, etc. The acquisition has



(a) (b)

Figure 3.5: (a) Map of the building where the dataset has been acquired, different colors mean different type of area traversed. (b) Acquisition system: an omnidirectional catadioptric camera mounted on a helmet.

been performed by a person wearing the helmet, so the dataset suffers the typical motion of a person walking. A long trajectory covering as much areas as possible was performed (many areas are locked or with restricted access so it was not possible to cover all regions in the building). Figure 3.5(b) shows the map of the three floors of the building highlighted with different colors, depending on the type of area traversed during the acquisition. The gray areas are parts not included in the dataset.

The visual part of the dataset consists of 20905 omnidirectional images at 1024x768 pixels resolution acquired at a frame rate of 10 FPS. The ground truth labeling of the building areas has been made according to our objective of separating *Places* and *Transitions*. We consider the main spaces of a building, like corridors or rooms, as *Places*. *Transitions* label comprises all the areas joining different *Places*: doors, stairs, elevators, etc. The more detailed classification in type of *Places* or type of *Transitions* has been chosen to adequately describe the environment of acquisition. *Places* are classified as Big, Medium and Small Rooms and Corridors. Typically small rooms correspond to offices, medium to classes and big to halls or laboratories, for simplicity we classify them according to their size despite their different uses. *Transitions* are classified as Doors, Jambs, Stairs and Elevators. The areas labeled as *Transitions* starts about 0.5 meters before and ends about 0.5 meters after the *Transition* has been crossed.

All images have been manually labeled with the type of area where acquired and its position. Consecutive images labeled with the same type of area have been grouped into clusters. Table 3.2 shows the number of clusters and between parentheses the number of images of each type.

Table 3.2: Number of clusters of each class in the dataset. Values between parentheses are the number of images of that class/subclass.

Places	TOTAL 56 (16522)	Corridor 38 (12577)	Big Room 7 (1559)	Medium Room 3 (1021)	Small Room 8 (1365)
Transitions	TOTAL 55 (4382)	Door 40 (1268)	Jamb 9 (514)	Stairs 4 (1933)	Elevator 2 (667)

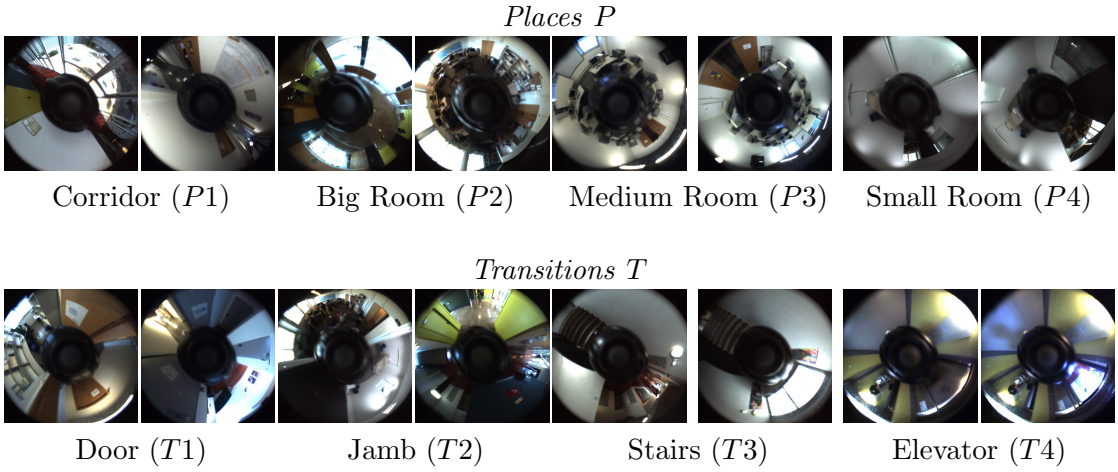


Figure 3.6: Examples of images labeled in the ground truth as elements of the different classes and subclasses.

3.5.2 Image representation evaluation

This first set of experiments is designed to evaluate how suitable and discriminative for our problem the image representation described is. These experiments evaluate different environment models and how they work classifying the rest of the images into *Places* and *Transitions*, as well as the detailed classification into subclasses.

As said before a key element of our labeling process is the reference model used. Then, we have tried to build this model automatically to avoid any bias with hand made selections. The basic model created from the dataset, let us name it *One*-Cluster-Model, includes only the first cluster of each subclass found in the sequence. The second model evaluated, named n-Cluster-Model, includes a variable amount of clusters considered as reference for each subclass, depending on the occurrence of each class and subclass.

In practice, the value of n_i used for the n-Cluster-Model is 5 for Corridor, 2 for Big room, 1 for Medium room and 1 for Small room in the case of *Places*. In the case of *Transitions* the value of n_i is 10 for Door, 2 for Jamb, 1 for Stairs and 1 for Elevator. 4351 images are used to build this model, while the amount of images used to create the *One*-Cluster-Model is about half of that value (2208 images). The amount of representative descriptors per class k is set to 25 after some initial evaluations of the approach. Then, the environment model is formed by 200 representative Gists (2 Classes $\times$ 4 Subclasses/Class

$\times$ k).

The test images used to evaluate the approach in the following experiments are all images in the dataset not used to create the model. We run a Naive Bayes Classifier based on the likelihood function described in eq. (3.3), that assigns a label to each image independently of the rest of images. It is a simple probabilistic classifier based on applying Bayes' theorem under independence assumptions. The formulation of the Naive Bayes Classifier in our case and following the nomenclature used for the formulation of the Hidden Markov Model is:

$$p(S_t = i|I_t) = \alpha \;\; p(S_t = i) \prod_{j=1}^{n} p(I_t|S_t = i), \tag{3.7}$$

again, $p(S_t = i|I_t)$ is the posterior probability of the event $S_t = i$ given the image I_t, $p(S_t = i)$ is the prior probability of the class i and $p(I_t|S_t = i)$ is the likelihood function (eq. 3.3). We set the same prior probability for each class: $p(S_t = i) = 0.5$ with $i \in [P, T]$.

The results of this classification using the *One*-Cluster-Model can be seen in Table 3.3a and Table 3.3b shows the results using the n-Cluster-Model. Each row contains the percentage of tests corresponding to a label correctly classified or wrong labeled with the other type. The accuracy is computed as the sum of all the correct classifications divided by the total number of classifications. The classification using any of the models works better for *Places* (P) than for *Transitions* (T) and, as it could be expected, the simple model is less powerful to represent the environment that the n-Cluster-Model, with around 5% higher accuracy. There are additional reasons to use the second model: first, indoor environments use to include more areas of some classes than others, e.g., in the building of the tests there are more doors than stairs or elevators; second, some areas of the same subclass can be very different, e.g., the hall of the building and a research laboratory are both classified as Big Rooms. The n-Cluster-Model is kept for the rest of the experiments as reference model.

Besides the basic *Place/ Transition* segmentation, we want to test how the proposed image representation works to classify the images into the considered subclasses. Following a similar approach, using our hand labeled ground truth, we classify all the images from each class (P or T) into the corresponding subclasses ($P1/P2/P3/P4$ or $T1/T2/T3/T4$). Tables 3.3c and 3.3d show the results of this experiment. Looking to the results for *Places* we can observe acceptable average values for the accuracy in the labeling, however there are big differences in the results at different subclasses (almost all corridor images ($P1$) are well classified, but only 34.05% of small rooms ($P4$) were labeled correctly. The misclassified rooms (about 30% for each room subclass) are usually classified as corridors. The poor results obtained for the classification among different rooms means that the descriptor is not discriminative enough to distinguish well between these subclasses. Table 3.3d shows the results for the classification of *Transitions* with also heterogeneous results for different subclasses but acceptable average accuracy above 80%. Conclusion after these results is that the representation proposed gives acceptable results to augment topological representations, but there are chances of better performance if we achieve a more discriminative representation for particular subclasses.

Table 3.3: Labeling results evaluating each test independently from the rest of the sequence with a Naive-Bayes Classifier. Top: results for *Place* (P) - *Transition* (T) classification using two different reference models. Bottom: subclasses classification results using the best performing reference model.

(a) *One*-Cluster-Model

	P	**T**
P	75.02	24.98
T	43.79	56.21
Accuracy: 71.51		

(b) n-Cluster-Model

	P	**T**
P	80.89	19.11
T	39.89	60.11
Accuracy: 76.82		

(c) *n-Cluster-Model for* Place *subclasses*

	P1	**P2**	**P3**	**P4**
P1	93.04	1.78	0.46	4.72
P2	28.48	61.13	10.31	0.08
P3	32.26	0.98	42.46	24.30
P4	43.90	20.00	2.05	34.05
Places Accuracy: 82.95				

(d) *n-Cluster-Model for* Transition *subclasses*

	T1	**T2**	**T3**	**T4**
T1	69.31	2.01	3.41	25.28
T2	15.42	44.71	25.33	14.54
T3	0.00	0.00	100.00	0.00
T4	0.54	0.00	1.09	98.37
Transitions Accuracy: 82.41				

3.5.3 Testing the mapping method

Previous subsection shows the accuracy of the labeling classifier: around 76% when classifying into *Places* or *Transitions* and around 82% when labeling one of the basic classes into one of its subclasses. This subsection summarizes our experiments to validate the whole mapping method proposed in Section 3.4.2.

First we evaluate the effect of including temporal consistency on the label assignment along the sequence. We compare results using the Hidden Markov Model (HMM) to decide the most likely class/subclass instead of the Naive Bayes Classifier evaluation. The HMM requires to adjust the probability of a transition to happen. In [Rottmann et al., 2005] the authors propose a system to automatically adjust the value of this probability based in the training data. We test this system, what give us values of $p(S_t = i|S_{t-1} = j)$ higher than 0.99 when $j = i$. The effect of this values is a benefit in the detection of *Places* to the detriment of the detection of *Transistions*. We set the value of $p(S_t = i|S_{t-1} = j)$ to 0.9 when $j = i$ and $p(S_t = i|S_{t-1} = j)$ to 0.1 when $j \neq i$ to obtain results more adequate to our objective. Using the HMM probability evaluation to assign the labels, *Places* and *Transitions*, we can see a slight improvement, as can be seen in Table 3.4a compared to previous results in Table 3.3b.

Secondly, experiments summarized in table 3.4, compare results including or not the $minSize$ filter explained in Section 3.4.2, in the tables 3.4a and 3.4b respectively. This filter compares the Gists distance between the first and the last images on the current

Table 3.4: Labeling results evaluating the probability of each class according to the HMM including or not the *minSize* filter. Top: results for *Place* (P) - *Transition* (T) classification. Bottom: subclasses classification results.

(a) P/T Classification without *minSize* filter

	P	**T**
P	82.87	17.13
T	39.86	60.14
Accuracy: 78.42		

(b) P/T Classification including *minSize* filter

	P	**T**
P	78.07	21.93
T	32.27	67.73
Accuracy: 76.04		

(c) Subclasses classification including *minSize* filter

	P1	**P2**	**P3**	**P4**	**T1**	**T2**	**T3**	**T4**
P1	76.61	0.70	0.09	1.63	7.48	0.56	4.13	8.79
P2	22.18	43.94	0.00	0.00	15.06	18.82	0.00	0.00
P3	47.77	0.00	49.16	0.00	0.56	0.00	2.51	0.00
P4	21.95	16.10	0.00	30.97	20.41	9.74	0.00	0.82
T1	33.30	0.00	10.53	1.71	41.12	1.71	0.00	11.63
T2	60.35	11.01	0.00	8.37	0.00	20.26	0.00	0.00
T3	2.31	0.00	0.00	13.38	0.00	0.00	84.31	0.00
T4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	100.00

cluster and check it with a similarity threshold, the distance must be over this threshold (th) to create a new cluster.

The images are classified with HMM and, as explained, they are grouped into clusters according to the assigned class: consecutive images fitting the conditions are grouped together. We can appreciate similar average accuracy in the P/T classification with or without taking into account the *minSize* filter. However the fact of avoiding too small clusters turns into a more meaningful semantic partition of the environment as detailed later. Detailed results of classification into subclasses are only shown for the complete approach, including the *minSize* filter, in table 3.4c. Results without using this filter were very similar, slightly better for subclasses of Places but slightly worse for subclasses of Transitions.

Table 3.5 shows the size and number of clusters we generate with the two options and the cluster arrangement done manually as ground truth when labeling the images. We have set empirically the *minSize* threshold to 0.275 for all tests. Note that using the *minSize* threshold most of the extremely small clusters are eliminated and the map obtained is more similar to the manually labeled map. Besides, as described before, this option provided better accuracy for Transitions, that is the particular labels we are interested the most.

Another interesting comparison run was to analyze the usefulness of doing jointly the semantic labeling and the topological clustering. We evaluated the results of the

Table 3.5: Number of clusters generated with the mapping approach with and without *minSize* filter.

	HMM	HMM + *minSize*	GT
# clusters	267	180	111
Minimum cluster size (# images)	1	7	19

Table 3.6: Map areas automatically detected (of # in the Ground Truth)

Places	$P1$	$P2$	$P3$	$P4$	TOTAL
	30 (of 38)	5 (of 7)	2 (of 3)	4 (of 8)	41 (of 56)
Transitions	$T1$	$T2$	$T3$	$T4$	TOTAL
	22 (of 40)	3 (of 9)	4 (of 4)	2 (of 2)	31 (of 55)

individual location labeling with or without getting a common sub-class label for all images in each cluster. We obtained improvements in the labeling results running both steps simultaneously and assigning a common label to all components in a topological cluster. This is not surprising, since by grouping images we take into account the subclass of all the images in the cluster as a group, so we filter some misclassification errors.

Finally, summarizing the experimental validation, Fig. 3.7 shows the trajectory of the sequence with the mapping results. This result is obtained with the whole sequence to obtain a representation of the whole environment. Then as the images used to estimate the model are included now, we observe higher accuracy values: 81.83% for the classification into *Places* and *Transitions*, 71.70% for the classification into *Places* subclasses and 74.37% for the classification into *Transitions* subclasses. Fig. 3.7(a) shows the manual segmentation into clusters and their ground truth class label, and Fig. 3.7(b) shows the segmentation after running our approach. Comparing both segmentations we can see where errors occur. Regarding *Places* detection, as previously observed, corridors are much clearly recognized than the different types of rooms. In the case of *Transitions*, the higher errors occur for Jambs (blue), that are present only in the first floor and are not detected, so the corridors that should be separated by them are joined in one cluster. Some errors also occur in the classification of corridors due to the creation of inexistent transitions. These errors may be happening because of rapid illumination changes that produce big appearance changes and artifacts in the images.

All previous classification evaluations have been estimated considering the individual labeling of each image. However, the objective when creating a semantic map is to correctly detect the different areas of the environment. Despite some mistakes, the map created captures the distribution of the areas of the building. Table 3.6 shows the number of areas detected according to their class and subclass. We consider an area detected by our approach when 50% of the images in that area have been correctly labeled. Usually the problem is that the generated clusters are still smaller than the ground truth annotated ones, that is why we consider correct detections even if only a part of the hand labeled images in the region are correctly classified.

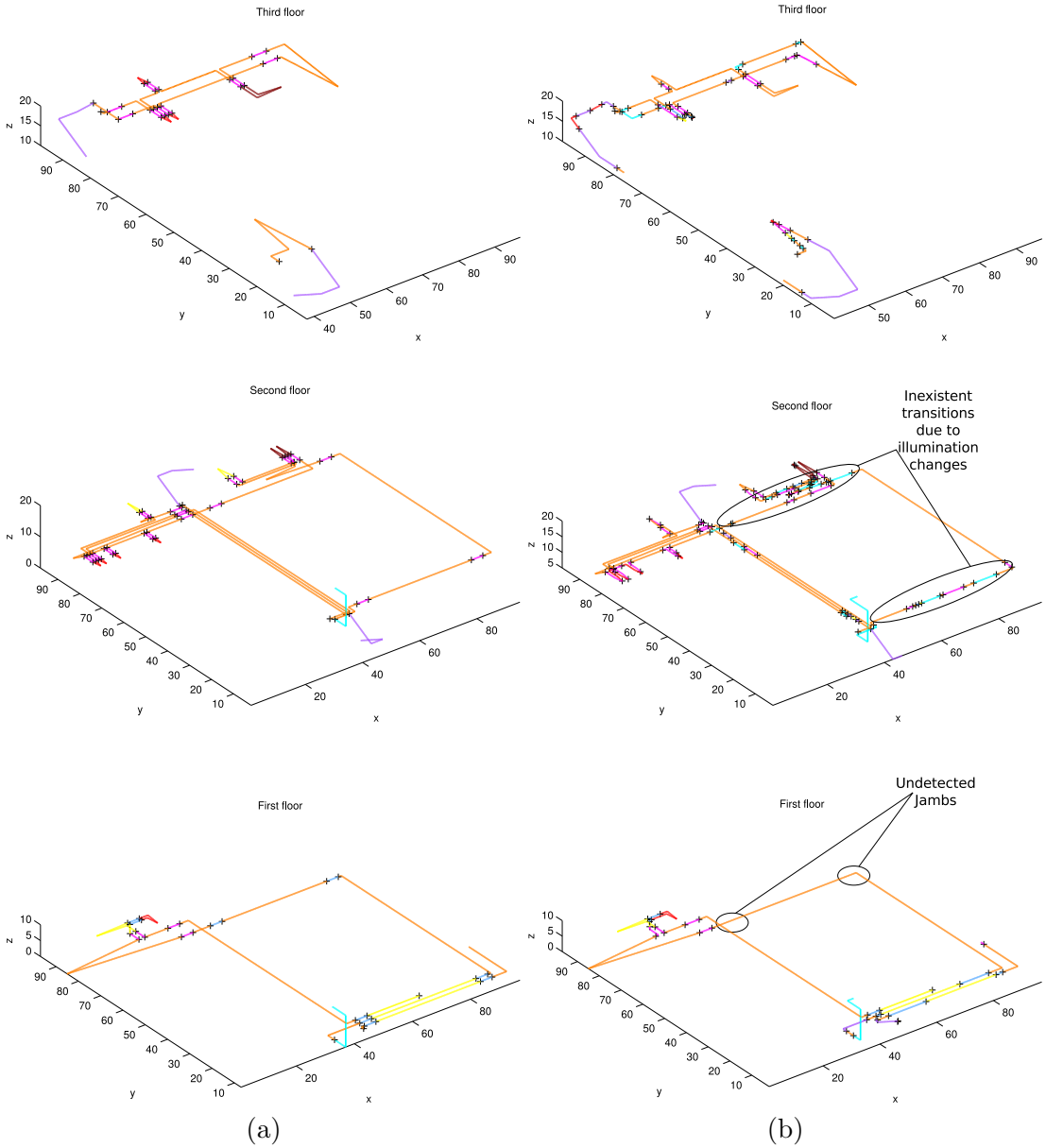


Figure 3.7: Segmentation of the trajectory in clusters of subclasses: (a) Manual, (b) Complete approach. The start position of each cluster is marked with a black cross. One color for each subclass:
Places: (Orange) Corridor, (Yellow) Big Room, (Brown) Medium Room, (Red) Small Room;
Transitions: (Pink) Door, (Blue) Jamb, (Purple) Stairs, (Light Blue) Elevator.

3.6 Conclusions

We have presented a novel indoor semantic place labeling method that includes information of the indoor scenes. The method uses catadioptric images and the adaptation of the Gist global descriptor to represent these images. The general idea proposed is to simultaneously run a topological map building approach and the classifier to label the different types of indoor scenes as *Places* or *Transitions*. The result is a semantic-topological model, where the nodes are *Places* and the edges are *Transitions* between these *Places*, including information about different types of *Places* (Big, Medium or Small room, Corridors) and *Transitions* (Door, Jamb, Stairs, Elevator). A detailed semantic analysis of the types of transitions is not common although could provide important information to later uses of the map. Our approach is based on this semantic classification of the images, using a simple environment model, integrated with a Hidden Markov Model framework to add spatio-temporal consistency.

Chapter 4

Line Image Signature: a global descriptor for scene labeling

This section propose a novel line-based image global descriptor that encloses the structure of the observed scene. This descriptor is designed with omnidirectional imagery in mind, where observed lines are longer than in conventional images. Experiments show how the proposed descriptor can be used for indoor scene recognition comparing its results to state-of-the-art global descriptors and how it can be easily adapted to work with different omnidirectional camera systems. Besides, we demonstrate additional advantages of particular interest of the new descriptor: higher robustness to rotation, compactness, and easier integration with other scene understanding steps.

4.1 Introduction

This section is focused on the problem of scene understanding on indoor environments, where lines are known to play an important role, see Fig. 4.1. People can easily guess the 3D structure of a scene represented by line sketches. Line and contour cues have been extensively used to analyze images since they provide very useful information. Contours occur as boundaries of objects, helping to detect them, or as frontiers between surfaces, encoding the structure of the scenes. Analyzing contours in the images has been shown useful for tasks such as object recognition [Belongie et al., 2002], 3D scene reconstruction [Hedau et al., 2009] or image registration [Russell et al., 2009].

We propose a novel line-based scene descriptor. This descriptor is obtained as follows: first, scene lines are extracted from the omnidirectional images and classified in the three scene dominant directions; then, the descriptor is built as a histogram that encloses the distribution of these lines at different image regions. The descriptor is intended for omnidirectional images, where the whole scene can be captured in just one image. Lines appearing in theses images are longer than in conventional images and the vanishing points appear in the image. However they present high distortion, making line detection more complicated.

The designed descriptor is compact and invariant to rotations around the vertical axis,

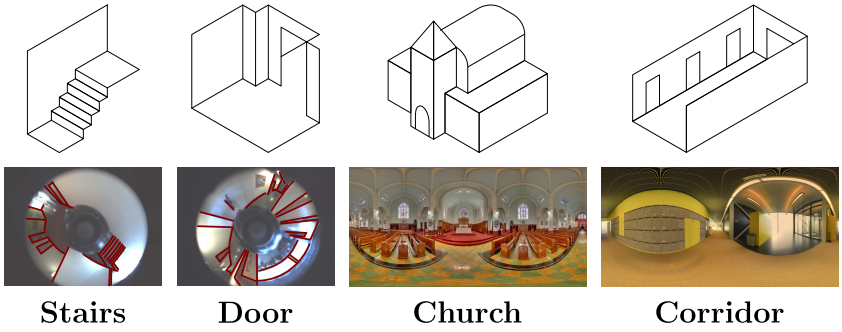


Figure 4.1: Line drawings are easily recognized by any person. The lines can be detected in images. Long lines if we work with omnidirectional images as the ones shown in the figure.

important requirements when working with robots or autonomous systems. Besides, the proposed descriptor extracts and processes scene lines in a way that can be used for other scene analysis techniques, such as 3D layout recovery.

The growing interest and developments on wearable computer vision systems are facilitating new systems and technologies for human assistance. Our goal is to provide a wearable indoor navigation assistance system with semantic information about the environment traversed by the user. The prototype developed consists of a helmet-mounted omnidirectional camera aimed for indoor navigation assistance. We use a catadioptric vision system able to capture a 360^o field of view, therefore observed lines have the advantage of being longer that in conventional images.

Our experimental validation shows a detailed analysis of the parameters of the descriptor computation and demonstrate its performance for indoor scene recognition on realistic datasets. The place recognition capabilities of the proposed descriptor are comparable to state of the art global image descriptors for this task, while it is shown to be more compact and presents higher rotation invariance (to rotation around the vertical axis of the camera). These additional advantages are important when working with a wearable system, due to the constant and heterogeneous movements done by a person. Besides, the proposed description method extracts and processes scene lines which are also required for following scene analysis steps, such as 3D layout reconstruction.

4.2 Related Work

Different types of contour based image features have been used in many computer vision applications since they provide very distinctive information.

For example, one of the applications where line cues have shown great potential for 3D scene understanding from a single image. Straight lines are highly present in man made environments, in particular, parallel lines aligned with the main directions of the scene (Manhattan World assumption). Based on these cues, authors of [Hedau et al., 2009] presents a method to extract the spatial layout of a room even with cluttered

boundaries. The approach from [Schwing et al., 2012] proposes an improvement of the performance of state-of-the-art methods for spatial layout computation by decomposing the potentials used in previous literature into more computationally tractable pair-wise potentials. We also find approaches specific for omnidirectional vision [Omedes et al., 2013], that extract the spatial layout of indoor scenes from a single image.

Lines and boundaries have been also used for shape and object recognition tasks. Authors in [Belongie et al., 2002] presented the shape context, which stores the relation between a contour point and the rest of the contour points of the shape. We also find works that use similarity measures defined for sets of connected contour segments for object recognition [Ferrari et al., 2008]. Line sketches were also shown to work well as models for object recognition in [Eitz et al., 2010]. Other applications of image lines include recovering the rotation between frames for visual based localization [Kosecká and Zhang, 2002] or their use for image retrieval [Russell et al., 2009].

Working with lines presents difficulties to obtain correspondences between images, usually because of the low accuracy or robustness of the line tip detection. However, lines present advantages for tasks that need to deal with extreme illumination changes or low textured environments [Wang et al., 2009a], outperforming local point feature based methods for these settings [Lowe, 2004]. We find approaches that propose to use straight line segments as local image features, describing them with different statistics around the edges. Many of these works make use of geometric constraints to obtain more robust matching results, e.g., homographies [Sagüés et al., 2006] or epipolar constraints [Bay et al., 2005]. Recent works have proposed more sophisticated line-based local descriptors, such as the Line Signature [Wang et al., 2009a] that outperforms point based local features matching low textured images. MDSL descriptor [Wang et al., 2009b] is another line-based local descriptor, which is built for each detected line segment. It is shown to be highly distinctive and robust to image rotation, illumination and viewpoint change.

Closer to our work, other approaches try to encode the image information with a line-based global descriptor [Kosecká and Zhang, 2002]. Here, the authors propose the Line Histogram, which represents angles and lengths of all the boundaries of an image in a histogram. Our approach also creates a line-based global descriptor, but it captures the distribution of the scene lines in the omnidirectional image.

We have chosen global descriptors because they have shown good compromise between precision and computational cost for general scene recognition problems. In [Oliva and Torralba, 2001] a global Gist descriptor is presented for scene recognition in real world scenes. Authors in [Dalal and Triggs, 2005] present the Histogram of Oriented Gradients (HOG) which encodes the gradient orientations present at different image regions.

As already mentioned, this work is focused on omnidirectional cameras. Due to the wide FOV of this kind of systems, more lines of the scene appear in the image and they are longer that in conventional images. However, in the catadioptric vision system used these lines appear as conics in the image. We find works that have faced the use of lines in omnidirectional images for rotation estimation [Bazin et al., 2012, Bermúdez-Cameo et al., 2012]. Both papers propose a method to detect scene lines in omnidirectional images and compute their vanishing points. Estimating the vanishing points in omni-

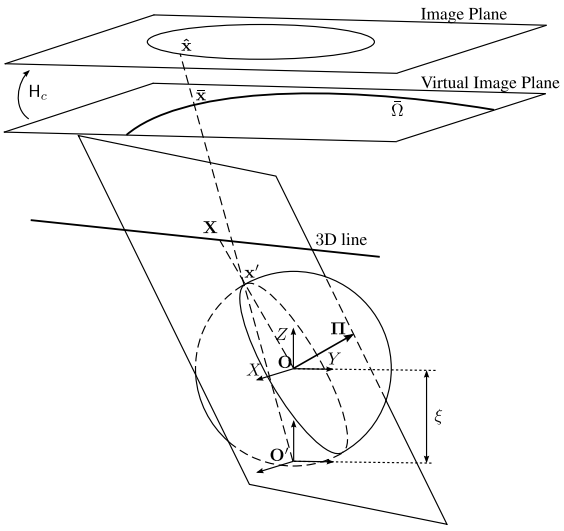


Figure 4.2: Projection of a 3D line and a point **X** in the line with the Spherical Camera Model.

directional images is typically robust and accurate, since these points are visible in the image. In this work we use the second work for line extraction and vanishing point computation.

4.3 Line-based Image Signature descriptor

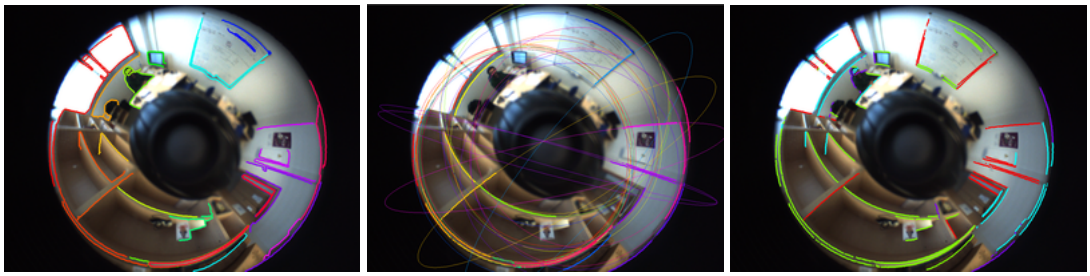
This section details the steps to obtain the proposed Line-based Image Signature (LIS) descriptor. First, the conics of the image, which actually correspond to straight lines of the scene, are extracted. Later, these conics are classified using the vanishing points information. Finally the descriptor is built as a set of histograms of the distribution of the classified contours in the image space.

4.3.1 Line extraction

This section details how straight lines of the scene are detected in the image and then classified according to their vanishing point. The difference between the different omnidirectional systems is how the scene lines are computed from the boundaries detected in the images.

Catadioptric images

The method used for the extraction of the scene lines in the catadioptric images was presented by Bermudez et al. in [Bermúdez-Cameo et al., 2012]. In this work, the authors describe a system to detect the conics projected in an omnidirectional image corresponding to straight lines of the scene. The method requires the calibration of the camera and



(a) (b) (c)

Figure 4.3: (a) shows the edges extracted by the Canny algorithm and how they are grouped. Each color represents a group. (b) shows the conics extracted for certain groups of edges and (c) represents the edges once all the conics have been classified according to the vanishing points. Vertical vanishing point (red), Horizontal vanishing points (blue and green), and Non aligned conics (purple).

uses just two points to adjust a conic in the image.

The Spherical Camera Model [Geyer and Daniilidis, 2000] is used to model the omnidirectional camera projection. The projection of a 3D point in the image through this model is performed in three steps. First, the 3D point, $\mathbf{X}$, in the reference system of the camera, is projected into a unitary sphere centered in the effective viewpoint $\mathbf{O}$. The resulting point, $\mathbf{x}'$, is reprojected into the virtual image plane through $\mathbf{O}'$, whose distance to $\mathbf{O}$ is ξ. The relation between the virtual image plane and the real image is a collineation: H_c. The process is shown in Fig. 4.2.

In central catadioptric cameras, the projection of a 3D straight line results in a conic in the image. A 3D line, l, defines a plane, $\mathbf{\Pi}$, together with the effective point of the omnidirectional camera, $\mathbf{O}$. Given this plane $\mathbf{\Pi}$, the equation of the conic projected in the virtual image plane is

$$\bar{\Omega} = \begin{pmatrix} n_x^2(1-\xi^2) - n_z^2\xi^2 & n_x n_y(1-\xi^2) & n_x n_z \\ n_x n_y(1-\xi^2) & n_y^2(1-\xi^2) - n_z^2\xi^2 & n_y n_z \\ n_x n_z & n_y n_z & n_z^2 \end{pmatrix}, \tag{4.1}$$

where n_x, n_y and n_z are the components of the normal of the plane, $\mathbf{\Pi} = (n_x, n_y, n_z)^T$.

A point, $\bar{\mathbf{x}}$, is part of the conic if $\bar{\mathbf{x}}^T\bar{\Omega}\bar{\mathbf{x}} = 0$. The relation between the point coordinates and the plane formed by the 3D line is

$$\alpha = -\frac{\bar{z}}{1-\xi^2} \pm \frac{\xi}{1+\xi^2}\sqrt{\bar{z}^2 + (\bar{x}^2+\bar{y}^2)(1-\xi^2)} \tag{4.2}$$

where $\alpha = \frac{n_x\bar{x}+n_y\bar{y}}{n_z}$.

Given two points in the virtual image plane, $\bar{\mathbf{x}}_1$ and $\bar{\mathbf{x}}_2$, the image conic including both being the projection of a 3D line, can be computed solving the next system

$$\begin{pmatrix} \bar{x}_1 & \bar{y}_1 & -\alpha_1 \\ \bar{x}_2 & \bar{y}_2 & -\alpha_2 \end{pmatrix} \begin{pmatrix} n_x \\ n_y \\ n_z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{4.3}$$

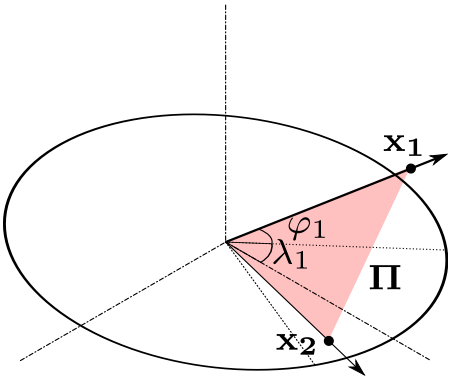


Figure 4.4: Line detection in an equirectangular panorama. $\mathbf{x_1}$ and $\mathbf{x_2}$ are two image points. For point $\mathbf{x_1}$ the angles φ_1 and λ_1 are shown. The plane Π, shown in red, defines the straight line joining both image points.

Using this relation, the process of the line extraction is the following. First, the Canny algorithm is used to detect the edges that appear in the image. Connected edges are grouped together in boundaries. For each boundary a two point RANSAC is run in order to detect the conics formed by the edges. Process repeats till the number of edges in the boundary falls below a threshold, and no more conics are adjusted. Fig. 4.3 shows plots of different steps of the detection process.

Equirectangular panoramas

In equirectangular images, coordinates in the image, relate linearly to pan and tilt angles in the real world

$$x = f_x\ \lambda; \qquad y = f_y\ \varphi \tag{4.4}$$

where x and y are the image coordinates and λ and φ are the pan and tilt angles related to the image coordinates trough f_x and f_y. To extract the lines from the boundaries, a RANSAC algorithm is used. Given two edges of a boundary and the camera center, a plane can be computed, Fig. 4.4. RANSAC algorithm looks for planes containing the most number of edges. These planes correspond to straight lines in the scene.

4.3.2 Classification according to vanishing points

Once all the image conics of the scene have been detected, we classify them according to the vanishing points. The vanishing points (VP) are the image points where parallel lines intersect. In man made environments, we find three main vanishing points, which correspond to the vertical lines and two sets of horizontal lines.

The vanishing points lay in the infinite, so they are defined by a direction, $\mathbf{v}_{VP}$. As showed in Fig. 4.5 all the normals of the planes created by parallel lines and $\mathbf{O}$ are coplanar. They are also perpendicular to the corresponding direction of the VP where they intersect, $\mathbf{v}_{VPk}$. Being l_i, l_j and l_m 3 parallel lines intersecting in the kth VP, and

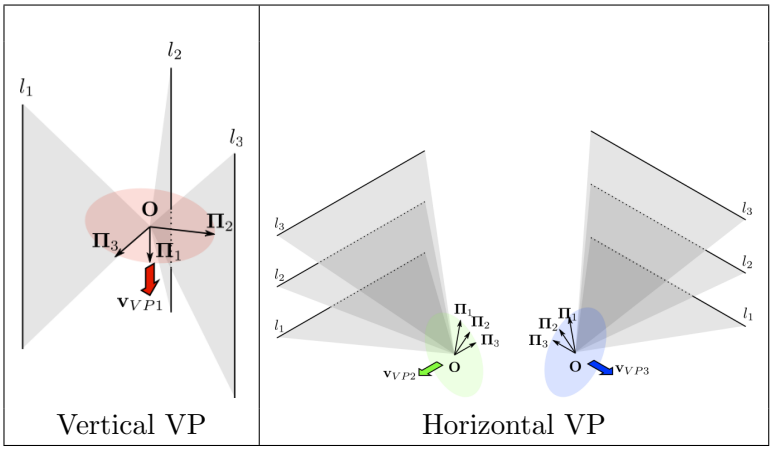


Figure 4.5: Sample sets of parallel lines and the corresponding VP directions. l_i denotes the line i and $\mathbf{\Pi}_i$ the normal of the plane created with $\mathbf{O}$ and represented by a gray surface. The normals of parallel lines are coplanar and perpendicular to the VP direction. The colored circles show the plane formed by the normals of parallel lines. These planes are perpendicular to the VP directions. Vertical VP, $\mathbf{v}_{VP1}$, (red), Horizontal VP, $\mathbf{v}_{VP2}$ and $\mathbf{v}_{VP3}$, (blue and green).

$\mathbf{\Pi}_i$, $\mathbf{\Pi}_j$ ad $\mathbf{\Pi}_m$ their corresponding planes, then

$$\begin{aligned}(\mathbf{\Pi}_i \times \mathbf{\Pi}_j) \cdot \mathbf{\Pi}_k &= 0 \\ (\mathbf{\Pi}_i \times \mathbf{\Pi}_j) \cdot \mathbf{v}_{VPk} &= 1\end{aligned} \tag{4.5}$$

These properties can be used to group the conics according to the vanishing points. We start with all the detected conics, each one represented by a $(\mathbf{\Omega}_i, \mathbf{\Pi}_i)$ pair. We assume an approximate vertical position of the camera, so we can predefine prior directions for the vanishing points: $\mathbf{v}_{VP1}$ is vertical and $\mathbf{v}_{VP2}$ and $\mathbf{v}_{VP3}$ are parallel to the image plane. We use a robust estimation algorithm (RANSAC) to adjust this estimation and obtain one group of conics corresponding to each of these main directions. In each RANSAC iteration two conics are randomly selected to create a group hypothesis. If these conics are parallel and aligned with the corresponding VP, $|(\mathbf{\Pi}_1 \times \mathbf{\Pi}_2) \cdot \mathbf{v}_{VPk}| \geq 1 - thP$, we check how many of the rest of the conics are parallel to them and vote for that hypothesis, $(\mathbf{\Pi}_1 \times \mathbf{\Pi}_2) \cdot \mathbf{\Pi}_j \leq thC$. The most voted hypothesis is chosen as one of the groups. Once the groups for the three VP have been obtained, the remaining conics are grouped as *nonAlignedConics*.

The result of this process is useful not only for globally describe the image. The information of VP aligned conics can be used to perform other scene understanding tasks such as 3D analysis of the scene under certain assumptions.

Once all the detected conics have been classified according to their VP, we can build the proposed LIS descriptor. To build the descriptor, the image space is discretized in a polar grid. The image is split in $4 \times n$ angular sections and each angular section is

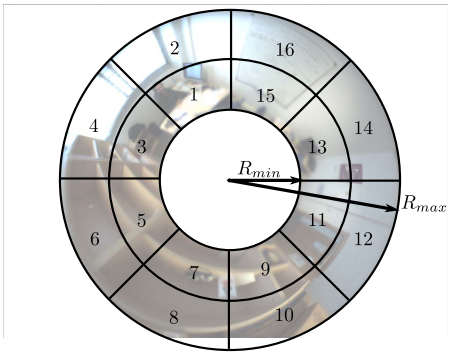


Figure 4.6: Image tessellation to incorporate in the descriptor the spatial distribution of scene lines (in this case $n = 2$).

split into n radial sections. Modifying n we can adjust the size and resolution of the descriptor.

Due to the self-reflection some parts of the camera and the mirror support appear in the image. In our system, these image parts are constrained by the minimum and maximum radius, R_{min} and R_{max} respectively. With this discretization, each histogram will be compound of $4 \times n \times n$ bins. Figure 4.6 shows the grid and the bins of the discretization for the simplest case $n = 2$.

We create histograms for the vertically aligned conics, h_{VVP}, for the horizontally aligned conics, h_{HVP} and for the non aligned conics, h_{nAl}. The value of bin i of each histogram is

$$\begin{aligned} h_{VVP\ i} &= 100\frac{\#\text{ Vertically aligned edges in bin } i}{\#\text{ Total edges}} \\ h_{HVP\ i} &= 100\frac{\#\text{ Horizontally aligned edges in bin } i}{\#\text{ Total edges}} \\ h_{nAl\ i} &= 100\frac{\#\text{ non aligned edges in bin } i}{\#\text{ Total edges}} \end{aligned} \tag{4.6}$$

where $i \in [1..4 \times n \times n]$. Histogram h_{VVP} is compound by the conics included in $VP1Conics$, h_{HVP} by the conics in $VP2Conics$ and $VP3Conics$, and the h_{nAl} histogram by $nonAlignedConics$. The total number of edges, # Total edges, correspond to the sum on all the edges of all the detected conics.

When grouping the conics of a scene, the two possible horizontal VP could be misclassified due to a different orientation of the camera in the same scene. In order to avoid this we join the conics aligned with the horizontal vanishing points, $VP2$ and $VP3$, in the same histogram, h_{HVP}. The final descriptor, h_{LIS}, is composed of the three histograms organized as follows:

$$h_{LIS} = [h_{VVP},\ h_{HVP},\ h_{nAl}]. \tag{4.7}$$

For omnidirectional cameras, rotation invariance is an important property to be able to recognize a scene when facing it with different direction of travel. To achieve rotation

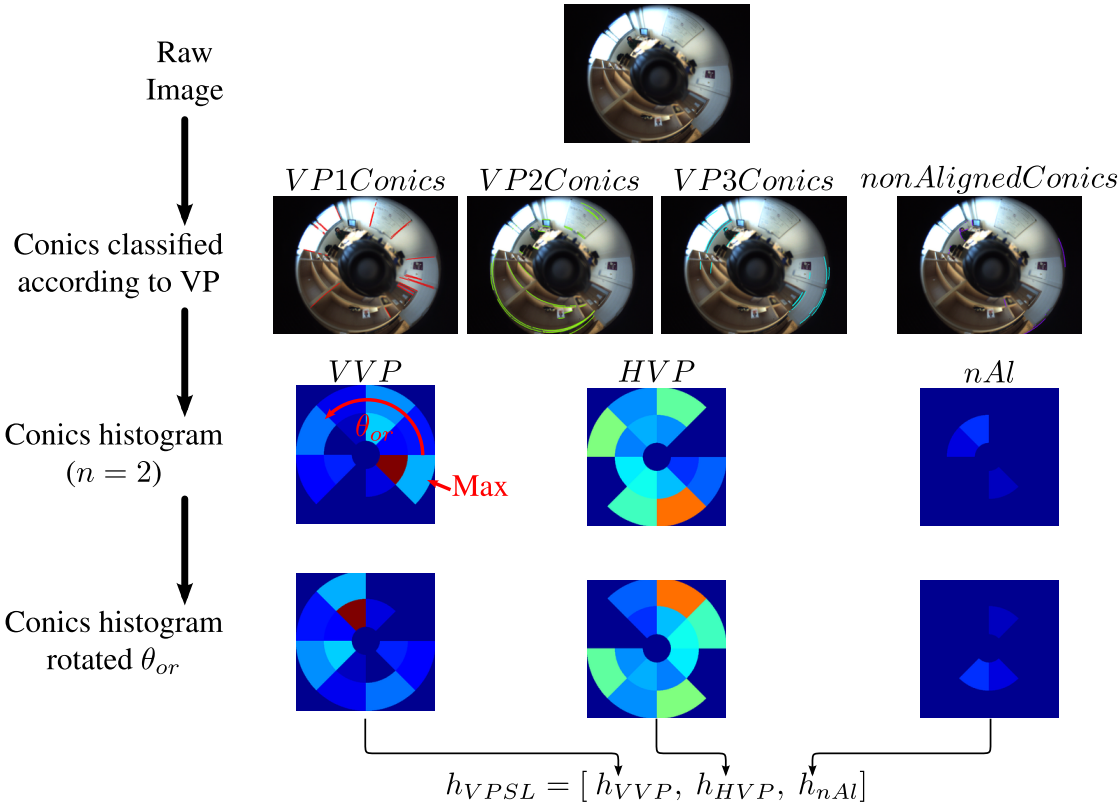


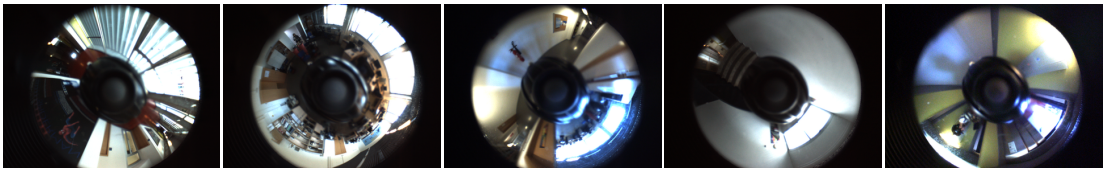
Figure 4.7: Steps to build the descriptor (from top to bottom). From the raw image we obtain the aligned conics, conics are then classified according to the VP of the scene and the histograms are built. Third row shows the sample histograms in polar coordinates (the color of each bin is related with the number of edges in that bin). Last row shows the final histogram, once the rotation invariance orientation has been obtained.

invariance, we have defined a common reference for all the images. We set the reference for each image to the angular segment where most of the vertical line edges lie. This segment gives us the orientation angle θ_{or}. Fig. 4.7 represents the described process from top to bottom.

Image similarity using the LIS descriptor

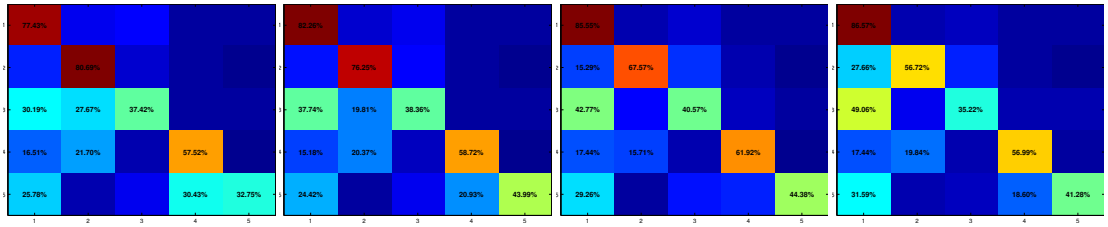
The distance between two of LIS descriptors can be simply computed as the absolute distance between histograms. Given the LIS descriptor of two images, h_1 and h_2, the distance d between them is

$$d = \sum_{i=1}^{3 \cdot 4 \cdot n \cdot n} \|h_{1i} - h_{2i}\|. \tag{4.8}$$



Class 1 Corridor | Class 2 Room | Class 3 Door/Jamb | Class 4 Stairs | Class 5 Elevator

Figure 4.8: Examples of the images included in the dataset for each considered category.



$n = 2$ $n = 4$ $n = 8$ $n = 16$

Figure 4.9: Confusion matrices for different image space discretization (n). Each row shows how many tests of that class were classified as any of the possible classes (1: Corridor, 2: Rooms, 3: Doors or Jambs, 4: Stairs, 5: Elevator). Only numeric values above 15% are shown. Color goes from Dark blue for 0% to Dark red for 100%.

4.4 Experiments

This section analyzes the properties of the proposed scene signature through experiments with real images, acquired with a calibrated wearable catadioptric vision system at 1024×768 resolution. The dataset is detailed in the following subsections.

4.4.1 LIS image tessellation analysis

This experiment evaluates the performance of our proposed descriptor for image categorization with different tessellation values.

Experimenta setting

We used the OmniCam dataset presented in [Rituerto et al., 2014c], and used for metric and topological indoor navigation. The dataset consists of two different sequences of images acquired in the same environment. To evaluate the robustness of the place categorization, the two sets were acquired at different times (several months between acquisitions) and covering different trajectories, i.e., some areas covered by the test sequence do not appear in the training set. The sequence used as training set in this work includes about 12200 catadioptric image frames, and the test set includes about 7300 frames. All the images were manually labeled with the corresponding type of indoor area

	Train	Test
Class 1	61%	70%
Class 2	19%	12%
Class 3	9%	4%
Class 4	9%	10%
Class 5	2%	4%

Table 4.1: Percentage of frames of each class in each sequence.

to set the labels ground truth. The 5 classes used to describe the areas of our indoor environment are shown in Fig. 4.8 together with some examples of images in the dataset. The percentage of frames of each class for the training set is shown in Table 4.1. It can be observed that some classes have many more examples than others, e.g., 61% corridor images and 2% of elevator images, due to typical configuration of indoor environments, since any user spends more time traversing corridors than in the elevators, and the whole sequences have been used.

The experiment consists on assigning a scene class label to each test image, according to the nearest neighbor found among the training images. We choose a nearest neighbor based approach due its simplicity and because other standard classification frameworks such as SVM or boosting based approaches typically require more training data. Probably the use of more complex learning techniques will raise the performance with any of the descriptors evaluated. Test images are compared with all the images of the training set, and the query is labeled with same class of the train image with the lowest descriptor distance, computed as described in (4.8). We run this experiment with different configurations of our approach.

Analysis of the results

Fig. 4.9 shows the confusion matrices of the classification using different image tessellation: $n = 2$, 4, 8 and 16. Higher values of n mean larger (descriptor size: $3 \times 4 \times n \times n$) and higher resolution descriptors. Looking at the results we can see how the performance is not homogeneous for all the classes. Corridors (Class 1) are recognized better in all the experiments, probably because they have well-defined vanishing points and lines. The worst performance corresponds to Doors/Jambs (Class 3). When traversing a door or a jamb, large part of the omnidirectional image contains areas of the environment before and after that door, so confusion using the description of the whole omnidirectional image seems reasonable.

Parameter n is directly related with the size and the resolution of the descriptor. In general, increasing n improves the performance, however, there is a point where higher n values does not improve the result, even performance is decreased. For our settings the best value is $n = 8$. We should note the behavior of the recognition for the Rooms (Class 2). For this class, an increment of n produces, in all the cases, worst results. This class groups many kind of spaces (halls, laboratories, offices) with different appearance but

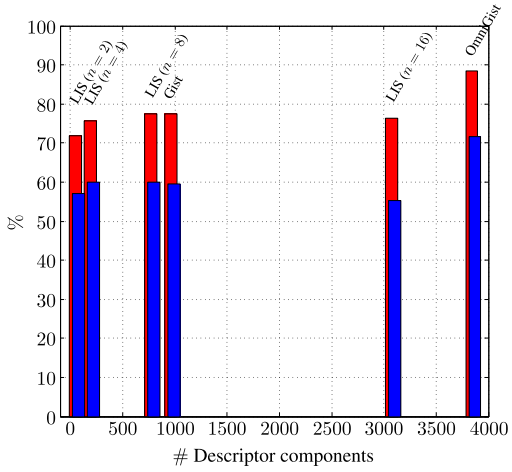


Figure 4.10: Average precision (red) and Average class precision (blue) for the descriptors tested as function of the descriptor size.

similar structure. In this case an increment of the resolution of the descriptor reduces the performance because it starts enclosing too subtle details that are not common to all elements in the class.

4.4.2 Comparison with state-of-the-art descriptors

This subsection shows the performance of our proposal compared to other global descriptors used for scene recognition in omnidirectional images. We compare three global descriptors: our proposed LIS approach, the Gist descriptor [Oliva and Torralba, 2001] using the code provided by the authors, and the adaptation of this descriptor to omnidirectional images [Murillo et al., 2010].

The average precision and average class precision for all the descriptors are shown in Fig. 4.10. The average precision (red) and the average class precision (blue) are plot as function of the number of components of each descriptor: $3 \times 4 \times n \times n$ for LIS, 920 for Gist and 3480 for OmniGist. Our proposed descriptor achieves a precisions between 72% and 78% depending on the discretization, Gist gets a precision of 83% and OmniGist 88%, and similarly for the average class precision. The precision of our approach is slightly lower for the used dataset, but the shorter length descriptor is an important advantage for navigation applications where memory and computational power can be a limitation. Fig. 4.11 shows some examples where LIS ($n = 8$) classify correctly the category of the query image while the OmniGist fails.

The dataset used was captured while traversing a typical indoor environment, therefore for most of the cases train and test images are related by multiples of 90^o rotation. Therefore, the more robust rotation invariance of our descriptor cannot be observed using only that dataset. Next experiment proves the higher rotation invariance of our approach.

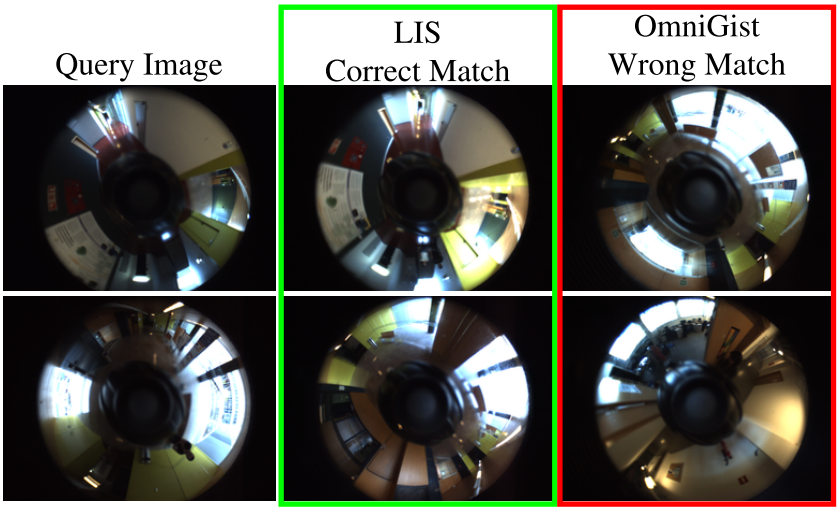


Figure 4.11: Examples of images where LIS descriptor performs better than the other descriptors. The first column shows the query images, second and third columns show the nearest train image selected by LIS and OmniGist respectively. Rotation between the query image and the correct match selected with LIS can be notice in the figure.

Rotation invariance

Rotation invariance is an interesting property to achieve robust place recognition when working with omnidirectional images. It allows to have less training data (we do not need to have examples of every possible acquisition angle at each location) but still a robust modeling of the different classes. This is one important advantage of our proposed descriptor.

Next experiment analyzes the rotation invariance of our approach compared to other global descriptors. We get 36 images equally distributed along a 360^o camera rotation movement, around the vertical camera axis. The camera used for this experiment is different than the one used in the rest of experiments. It has been installed a goniometer for angle measurement and the images resolution is 1024×768. Fig. 4.12 presents a plot of the descriptor distance versus the angular difference between all images in the sequence and the first one. To be able to compare the distances of different descriptors, the distance values have been normalized using Standard Score normalization. Descriptor distances are shown as the difference with the mean of the distances, μ, in units of the standard deviation, σ.

For the standard Gist descriptor, we observe it is not rotation invariant. The minimum distance appears in 0^o and 360^o, but grows continuously to the maximum value at rotations around 180^o. For the OmniGist descriptor, minimum values of the distance appear for angle differences multiple of 90^o, while maximum values appear for angles multiple of 45^o. Finally, for our approach the distance also varies with angle, the maximum distance occurs for rotations around 180^o, however we can observe local minimum points around rotations multiple of 45^o. This is due to the image space discretization

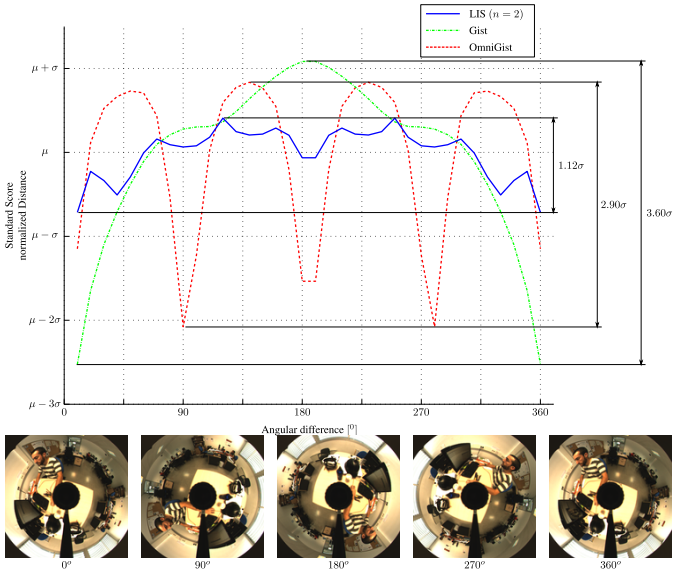


Figure 4.12: Rotation invariance analysis. Relation between the descriptor distance (vertical axis) and the angle difference (horizontal axis) among images of the same scene, acquired at the same location only with varying angles around the vertical axis. The proposed LIS descriptor presents much lower variation than other global descriptors.

used for the descriptor, $n = 2$, where each angular segment corresponds to an interval of 45^o. The variation of the distances, between images in the sequence and the reference image at 0^o, is much lower for our proposed method, showing higher invariance to rotations around the vertical axis.

4.4.3 Scene Categorization performance with panoramic images

This section shows the performance of the LIS descriptor for scene categorization using a panoramic dataset and for navigation assistance using images from a catadioptric vision system. We use the publicly available SUN360 dataset, presented in [Xiao et al., 2012]. It contains $67,583$ panoramas of 80 categories. The panoramas are labeled as indoor or outdoor scenes and further classified into more detailed categories. In this work just indoor images are used: $3,889$ images classified into 15 categories (cave, church, corridor, hotel room, living room, lobby atrium, museum, old building, restaurant, shop, showroom, subway station, theater, train interior and workshop).

The goal in scene categorization is to detect the type of environment where an image was acquired. Our experiment consists of classifying the test images into the correct class by comparing them to the reference set using our proposed descriptor. We randomly split the dataset into test and reference sets: 70% of the images are used for test, and the reference set consists of the remaining 30%. All the images in the test set are compared to the images in the reference set, and the class assigned is the class of the closest reference image (Nearest Neighbor classification).

	Accuracy		Descriptor
	Total (%)	Average (%)	size
$n = 2$	18.81	16.69	48
$n = 4$	18.59	16.50	192
$n = 8$	19.10	16.92	768
$n = 16$	18.81	17.45	3072

Chance: 6.67%

Table 4.2: Total and average accuracy of the scene categorization for different image tessellations.

Table 4.2 shows the Total and Average per class accuracy of the classification for different tessellations, and the descriptor size. Best total accuracy is achieved for $n = 8$, when the descriptor includes 768 components. It has to be noticed that the accuracy differences are small compared with the descriptor size, bigger descriptor mean higher resolutions but do not represent better performance. Best performance correspond to hotel room (46.79%) and subway station (38.42%), however, the descriptor accuracy falls for old building (8.13%) and shop (7.12%).

4.4.4 Integration with spatial layout extraction steps

As already described, our goal is to integrate the proposed place recognition in our navigation assistance system. This work shows how to use our line based descriptor for scene recognition, but the same line information can be used for additional tasks such as 3D layout recovery. Using LIS global descriptor for place recognition, our system is able to detect when the kind of area traversed has changed and the category of the new place. The 3D scene analysis, that is computationally expensive, can then be run just when necessary, i.e., after a change of the type of area visited or for certain types of places.

As initial example of further exploitation of the lines detected we have run the code presented in [Omedes et al., 2013] to detect the scene layout in omnidirectional images. This approach follows a heuristic to iteratively fit the wall-floor boundary, which follows the same steps and priors for any location. The same lines used to build the LIS descriptor are used to detect the scene layout. Fig. 4.13 shows the results of both tasks: the LIS histogram for scene categorization and the spatial layout of the image, where floor and walls are detected.

4.5 Conclusions

In this section we have presented a new line-based global descriptor for omnidirectional images that encloses the structure of the scene observed, by encoding the spatial distribution of the scene lines in the image. The descriptor is built as a histogram that captures how the different types of scene lines lay in the different parts of the image

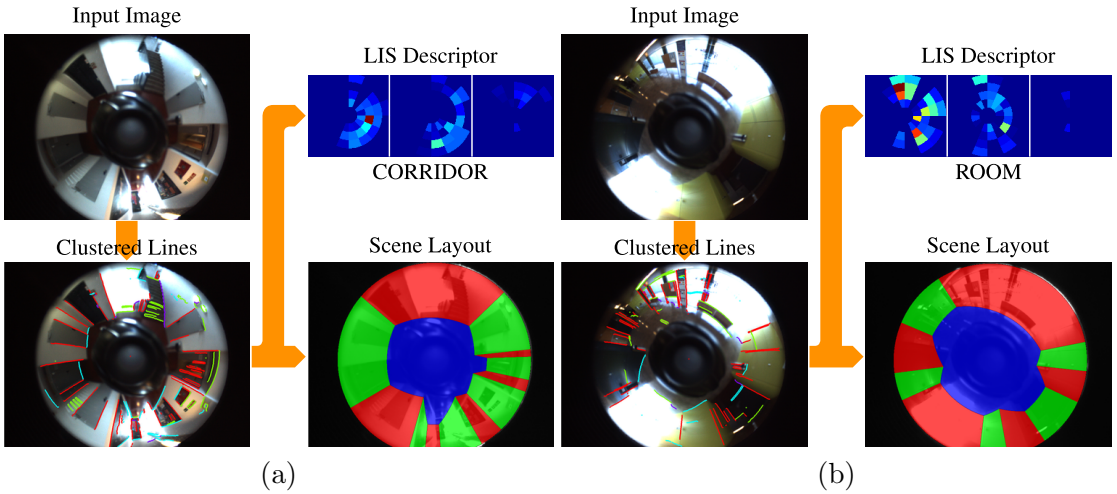


Figure 4.13: Two examples of the joint process of scene categorization and layout recovery for a corridor image (a) and a room image(b). First we extract and cluster the scene lines from the original image. With the line information we can compute the LIS descriptor ($n = 4$ shown) and get the kind of area being traversed. Besides, the layout recovery approach uses the same line information to get the scene planes of the scene, floor (blue) and walls (red and green) in this case.

space. We have shown how the descriptor can be easily adapted to work with different omnidirectional systems. We have used two different systems: catadioptric and panoramic cameras. Experiments for scene categorization show that the performance of our proposed descriptor is close to state-of-the-art global descriptors. Besides, we have shown our approach to have interesting advantages for such as small size, reducing the memory consumption and comparison time, and higher rotation invariance are interesting properties for person-mounted cameras. Additionally, preliminary results of integration with complementary line based scene understanding techniques have been shown.

Semantic models of the environment

The use of assumptions or constraints about the environment, can be used to facilitate the extraction of semantic information about a system surrounding areas. For example, by assuming that the system is working on an indoor environment where the different scene planes (floor, walls and ceiling) are aligned with 3 main directions, we can know how the scene structure will be projected in a camera. We can also assume where some objects will appear, i.e. signs and doors appear in the walls at different heights, light sources in the ceiling and obstacles in the floor. Next chapters describe different ways of including priors about the environment in the process of building a model. This allows to add semantic information to the resulting models.

Chapter 5 studies how to leverage the standard vocabulary construction process to obtain a more meaningful visual vocabulary for particular applications using image sequences. Vision based recognition approaches frequently use quantized feature spaces, commonly know as Bag of Words (BoW) or vocabulary representations. We take advantage of spatio-temporal constraints and prior-knowledge about the position of the camera to incorporate tracking information to the process of vocabulary construction, and to add geometric cues to the appearance descriptors.

Chapter 6, we study the particular case of indoor scene understanding from monocular images. We address the problem of scene layout propagation along a video sequence. The presented approach uses a Particle Filter framework to propagate the scene layout obtained using a state-of-the-art technique on the initial frame. We propose how to generate, evaluate and sample new layout hypotheses for the scene on each frame.

Chapter 5

Building a hierarchical visual vocabulary from an image sequence

Vision based recognition approaches frequently use quantized feature spaces, commonly know as Bag of Words (BoW) or vocabulary representations. A drawback using standard BoW approaches is that conceptual or semantic information is not considered as criteria to group the visual features into words. To solve this challenging task, this section studies how to leverage the standard vocabulary construction process to obtain a more meaningful visual vocabulary for particular applications using image sequences. We take advantage of spatio-temporal constraints and prior-knowledge about the position of the camera. The key contribution of our approach is to define a new method to incorporate tracking information to the process of vocabulary construction, and to add geometric cues to the appearance descriptors. Motivated by long-term indoor robotic applications, we focus in a robot camera pointing to the ceiling, which facilitates the capture of more stable regions of the environment, improving long term operation and the discovery of repetitive and representative elements. The experimental validation shows how our vocabulary models the environment in more detail than standard vocabulary approaches, while keeping comparable recognition performance. We show different robotic tasks that could benefit of the use of our visual vocabulary approach, such as place recognition or object discovery. For this validation we use a publicly available dataset[1] fitting the requirements of our approach.

5.1 Introduction

Quantized feature spaces have been broadly used in visual recognition problems, such as object or place recognition and image retrieval [Sivic and Zisserman, 2003, Tuytelaars et al., 2010, Cummins and Newman, 2011, Philbin et al., 2007, Jégou et al., 2010]. These

[1] Dataset: `http://aass.oru.se/Research/Learning/datasets.html`

techniques create a catalog of image features or words and describe images as a vector of occurrence counts of these words.

A typical drawback using standard BoW approaches is that semantic information is usually neglected when grouping the visual features into the clusters or visual words. In a general setting, including conceptual information is challenging since no assumption can be made about the type and meaning of the visual features that may appear. However, many applications could benefit from including semantic content of their working environment in a visual vocabulary. We study certain reasonable assumptions that are effective to achieve that goal for one of those applications. The general objective of our work is to create a vocabulary modeling the working environment of an autonomous system. Particularly we focus on mobile robotic platforms performing long-term operations indoors, in environments such as a warehouse [Everett et al., 1995] or a museum [Burgard et al., 1999].

The approach presented in this work is built after the following hypotheses. First, robotic platforms provide sequential information (a sequence of images), therefore, including tracking information while building the vocabulary helps to cluster the different appearances found along the way for the same scene element. Secondly, we can find spatial restrictions because cameras in robotic platforms have a fixed location respect to the robot. Reasonable assumptions can be made about the location of elements of the environment (e.g., the lamps are on the ceiling, the posters on the wall,...). We use simple geometric information together with the appearance to cluster the environment elements according to their appearance and scene location.

Following previous robotics applications [Fukuda et al., 1996, Wulf et al., 2006], we benefit of having a camera pointing to the ceiling. Upper parts of indoor scenes are typically less dynamic than the rest of the scene and provide a more robust visual model. In this setting, we expect the environment to include a small number of repeatable elements (e.g., different kinds of lamps, windows) and few elements which are rather unique (e.g., exit signs, posters, labels), most of them with fixed locations within the scene. We assume that these elements are viewpoint dependent and present a restricted set of appearances from a few points of view that are reachable by our acquisition platform.

Our proposed approach consists of a novel hierarchical process (summarized in Fig. 5.1) that obtains a visual vocabulary with richer information about the environment, compared to standard visual word clustering methods. Instead of computing the visual words directly on the acquired data, we propose to obtain an initial grouping into classes containing similar sets of tracked key-points. Then, standard vocabularies are computed in each of those classes. This ensures that the vocabulary will include words covering all the different types of key-points.

The main differences and contributions of our approach with regard to prior work are the following. We study how to leverage the visual vocabulary for a more meaningful representation of a given working environment, as opposed to try to build a generic visual model. This facilitates complex tasks in that particular environment; We propose a novel way to include spatio-temporal information in the vocabulary construction process:

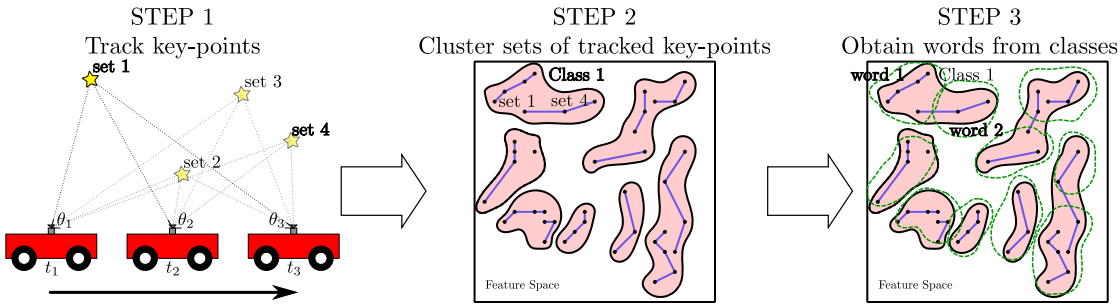


Figure 5.1: Diagram of our novel vocabulary construction method. During the acquisition, extracted image key-points are tracked. All the image appearances and azimuth angle values of these key-points are stored and grouped together under the same set number. These sets of tracked key-points are later clustered into classes, with the elements in each class representing the same environment element. Finally, words are extracted running a clustering for each of the obtained classes.

thanks to feature tracking, our approach automatically groups the different appearances of scene elements; additionally we propose to include each key-point azimuth value in the key-point descriptors to encode the different feasible viewpoints for the scene elements.

Our experimental section demonstrates how the presented approach builds a visual model that provides higher representativity of the environment. At the same time the created vocabulary mantains similar performance for place recognition than a standard vocabulary (e.g. built with the usual k-means algorithm). The resulting enhanced vocabulary can benefit plenty of robotic applications that require a visual model, such as the experiments on place recognition or object discovery that we have run.

5.2 Related Work

How to acquire a representative visual model of the environment is a problem that has been studied for long. In particular, acquiring models of the environment that could facilitate long term-operation is a subject of great interest, since it would provide intelligent systems with significantly higher autonomy [Konolige and Bowman, 2009, Valgren and Lilienthal, 2010].

Indoor vision based robotic tasks have taken advantage of using visual ceiling features [Fukuda et al., 1996, Wulf et al., 2006] because they typically correspond to less dynamic environments, most suitable for long term operations. Elements on ceilings are usually more stable over time than those in floors or walls, where dynamic scene elements appear often. Recent results revisited this idea for indoor and industrial oriented robotic applications [Xu et al., 2009, Hwang and Song, 2011], where long-term operation in a particular environment is required.

One of the most popular approaches to build visual models, is the Bag of Words (BoW) representation. It is based on a quantization of the image feature descriptor space into several groups (visual words) that compose the visual vocabulary. BoW based

approaches are very popular in various recognition tasks due to their good performance despite the simplified representation. For example, we find many good results on recognition at large scale, achieved partly thanks to BoW image representation, such as appearance based localization and mapping in large scenarios [Cummins and Newman, 2011] or efficient object retrieval [Philbin et al., 2007]. In the seminal work from Sivic et al. [Sivic and Zisserman, 2003], authors proposed to use tracked features to build a visual vocabulary from sequential data because they are more likely to be stable features. Inspired by these ideas, we use key-point tracking not only to find a set of stable features but to automatically discover and group the different appearances of the environment elements depending on the viewpoint.

In parallel with the growing popularity of vocabulary based recognition approaches, we also find research results analyzing the feature descriptor quantization drawbacks [Philbin et al., 2008, Boiman et al., 2008], such as the loss of rare but discriminative information or under-representation of descriptor space regions with low population. Trying to overcome some of these issues, we find several papers proposing enhanced methods to build visual vocabularies. Besides avoiding some of the drawbacks listed before, they augment the model with additional information. For example, in [Yang et al., 2008], the processes of vocabulary generation and training data classification are unified. Authors of [Jing-Yan Wang et al., 2013] present Joint-ViVo, a method where words and weights are learned jointly. In [Fernando et al., 2012] a supervised learning method is presented to add semantics to the words of the vocabulary. The criteria used are the likelihood of the training data and the purity of the clusters. The work in [Cao et al., 2010] proposes to include some geometrical information in the vocabulary obtained from features location in the scene.

As previously mentioned, one of our intuitions is that geometric and spatial restrictions on the scene are important cues to improve our visual model. In our work, spatial information in the form of the key-points azimuth is included as part of the key-points descriptor. Closely related to some of our hypothesis are those from the work presented in [Chum et al., 2009], describing the Geometric min-Hashing. This method uses semi-local geometric information to create and increase the discriminative power of the hash keys, and demonstrates advantages of using geometric information to improve their visual model.

We can also find several results using BoW representations together with spatial information. In [Yang and Newsam, 2011] spatial pyramids are used to characterize the spatial arrangement of words in the image. Following a different approach, papers [Bolovinou et al., 2013, Penatti et al., 2014] include the relative spatial configuration of the words to improve the vocabulary. The first work introduces a bag of spatio-visual words representation (BoSVW) obtained by clustering of visual words correlogram. The second, uses spatial arrangement of visual words (WSA) for image retrieval and classification outperforming spatial pyramids in the retrieval scenario.

Finally, related to our goal of including information in the model about interest elements discovered in the sequence, it is necessary to consider some interesting recent related papers on unsupervised learning for object or feature discovery. This kind of

approaches try to find image elements that appear frequently and are meaningful for humans. For example, unsupervised learning has been used to discover the distinctive architecture elements of determined areas [Doersch et al., 2012], representative views of an object [Berg and Berg, 2009], object models [Fergus et al., 2003] or the appearance of objects and its image segmentation [Russell et al., 2006]. Closer to our approach, the work presented in [Singh et al., 2012], finds a set of discriminative and representative image patches by using an iterative process of clustering and training.

5.3 Enhanced vocabulary construction

This section details our approach to build a semantic vocabulary of the environment traversed by a mobile camera. Main 3 steps are summarized in Fig. 5.1: 1) feature detection and tracking, grouping the features into sets of key-points, 2) clustering sets of similar appearance into classes, 3) obtain the visual words from each class, achieving a balanced vocabulary which has the words organized into classes (e.g., we have all the words that describe certain object organized into the same class).

5.3.1 Key-point detection and tracking.

First, we detect interest key-points in the scene and track them using Lucas-Kanade tracker, which provides the location of the tracked scene points in consecutive frames. For each frame we compute an image descriptor around each key-point. For both key-point detection and image descriptor SURF [Bay et al., 2008] is used. Note that the appearance of key-points is likely to change while they are tracked. We can exploit the knowledge of these appearance variations being generated by the same entity thanks to the tracking. Due to the camera configuration, pointing towards the ceiling, the azimuth angle encodes the change of point of view that produces the appearance change of the scene point. Therefore, the tracked image descriptors are stored, together with the azimuth angle θ of the corresponding location, as shown in equation 5.1

At this point, we have identified m sets of tracked key-points, where each set, *set* i, contains n_i image descriptors and azimuth angles from the same tracked scene point from different points of view:

$$\left[\begin{array}{c|cc} & SURF_{1_1} & \theta_{1_1} \\ set\ 1 & \vdots & \vdots \\ & SURF_{n_1} & \theta_{n_1} \\ \hline \vdots & \vdots & \vdots \\ \hline & SURF_{1_m} & \theta_{1_m} \\ set\ m & \vdots & \vdots \\ & SURF_{n_m} & \theta_{n_m} \end{array}\right] \tag{5.1}$$

Some of these sets could still contain features that actually belong to the same scene element, e.g., scene points tracked at different time while revisiting the same location or points corresponding to repeated objects in the environment. The following step of

our approach groups sets that are likely to have been produced by the same object or element.

5.3.2 Clustering of the sets of tracked key-points

The second step of the proposed method clusters sets of tracked key-points that have a high similarity. First we describe the similarity measure used to compare two sets of key-points. Then we detail the clustering methods evaluated in our experiments: Hierarchical Clustering and DBSCAN.

Similarity measure between sets of key-points

As mentioned before, our goal with this clustering step is to merge key-point sets that are likely to correspond to the same scene elements; we estimate this according to the following similarity measure.

The azimuth value encodes the relative position of the scene point with respect to the camera. We assume that the appearance of a scene point from the same viewpoint is the same, but different scene points can look similar from different positions. We use the azimuth difference to penalize these cases.

The distance between two features $\mathbf{X}_i = [desc_i,\ \theta_i]$ and $\mathbf{X}_j = [desc_j,\ \theta_j]$, is computed using:

$$d(\mathbf{X}_i, \mathbf{X}_j) = ||desc_i, desc_j|| f(||\theta_i, \theta_j||), \tag{5.2}$$

where $||desc_i, desc_j||$ is the euclidean distance between the appearance descriptors, $||\theta_i, \theta_j|| = |\theta_i - \theta_j|$, and $f(x)$ is the penalization due to the azimuth difference between both features, shown in Fig. 5.2 and obtained as follows:

$$f(x) = (1 + a \exp(b \exp(c\ x))), \tag{5.3}$$

where a, b and c are the parameters that define the shape of the penalization.

The penalization grows rapidly and continuously after a small difference of azimuths (0.1 rad), and the maximum penalization is used for values above 0.8 rad. Parameters a, b and c have been selected to accomplish these requirements. Other functions with the same properties are also valid.

Experiments section demonstrates that the inclusion of the azimuth in the descriptor and the similarity measurement improves drastically the object recognition rates, allowing us to distinguish among similar objects in appearance, but with particular and discriminative locations in the scene.

To compare two sets of tracked key-points we have to compare all the features of one set with all the features from the other set. Each feature of $set\ i$ is compared with the features of $set\ j$, the minimum distance value is selected, and the mean of all these minimum values is considered as the distance between sets D_{set}:

$$D_{set}(set\ i, set\ j) = \underset{\mathbf{X}_k \in set\ i}{mean} (\min_{\mathbf{X}_l \in set\ j} (d(\mathbf{X}_k, \mathbf{X}_l))), \tag{5.4}$$

where $D_{set}(set\ i, set\ j)$ is the distance between two different sets, $set\ i$ and $set\ j$, and $\mathbf{X}_k$ and $\mathbf{X}_l$ are features included in these sets respectively.

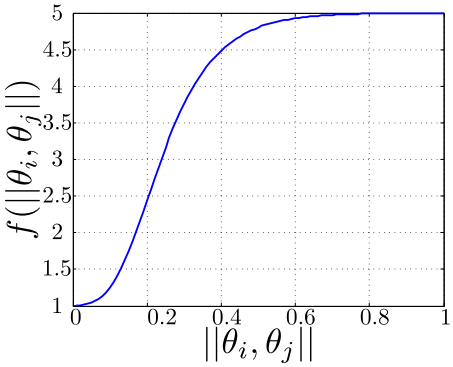


Figure 5.2: Penalization distance (5.3) between two azimuth values, with $a = 4$, $b = -7.5$ and $c = -10$.

Clustering approaches

We have considered the following two common clustering approaches that build on a similarity measure between the elements to be clustered.

- *Hierarchical Clustering.*
  We have implemented a clustering method based on the agglomerative Hierarchical Clustering [Ward and Joe, 1963], where each element starts as a cluster and, each cluster is paired with its most similar cluster as we move up in the hierarchy. Our hypothesis is that elements in the same cluster are probably observations originated from the same object or scene point. We define a similarity threshold, th_S, to avoid merging too dissimilar clusters, therefore clusters are not merged if the distance between their elements is over this threshold. As a result of this modification of the standard Hierarchical Clustering, key-points sets dissimilar to any other set are not paired and compose a singleton cluster. In this method, new clusters are created on every iteration, so new distances have to be computed. We adopt the Unweighted Pair Group Method with Arithmetic Mean [Sokal and Michener, 1958] approach, where the distance between two clusters is the mean of the distances between the sets of tracked key-points included in each cluster.

  Hierarchical Clustering is conceptually simple, that means it is easy to implement and modify, as required in this work. Additionally, it outputs a hierarchy of clusters, a structure more informative that flat clustering techniques results. The drawback is its complexity, $\mathcal{O}(n^3)$ in the general case, what makes it too slow for big datasets.

- *DBSCAN (Density-based Spatial Clustering of Applications with Noise).*
  DBSCAN [Ester et al., 1996] is a density-based clustering algorithm that uses a estimated density distribution of corresponding nodes to find clusters in the data. This algorithm is based in the notion of density reachability: Two elements, q and p, are directly density reachable if their distance is not bigger than ε. q is called density-reachable from p if there is a sequence of elements, $p_1 \dots p_n$ with

$p_1 = p$ and $p_n = q$ where each p_{i+1} is directly density-reachable from p_i. With this definitions, a cluster is a subset of elements mutually density-connected. To handle the noise, this method defines the parameter $minPts$, the minimum number of elements required to create a cluster. Subsets of density-connected elements with less than $minPts$ elements are considered as noise.

DBSCAN is a widely used clustering technique. We use the general DBSCAN implementation included in ELKI 0.6.0 [Achtert et al., 2013]. The complexity of this method is lower than the complexity of the Hierarchical Clustering, $\mathcal{O}(n^2)$ for the basic form of the algorithm, so it is faster and more appropriate for big datasets.

The results of this clustering step can be semantically understood as follows:

1. The obtained clusters represent common scene points and include their possible appearances according to the different viewpoints under which the scene elements were observed.

2. Non paired or noisy sets are unique scene points. These sets are dissimilar to the rest of sets but may be highly representative of the locations where they appear. These unique sets are clustered together after running the clustering step. The created cluster don't represent any common scene points, but include unique key-points representative of a location. Unique sets are clusters with just one set of tracked key-points when using Hierarchical Clustering. When running DBSCAN with $minPts = 2$, these unique sets are considered as noise.

5.3.3 Vocabulary construction

Our last step to obtain the visual model of the environment consists of building a visual vocabulary for each class resulting from previous steps. This vocabulary is composed by a fixed number of words K (as done in normal BoW generation). Differently from standard approaches, we consider how to distribute the visual words across all our feature space. We compute a number of words, k_i, for each class i proportionally to the number of features included in that class, with a minimum number of words of 1:

$$k_i = \left\lceil K \, \frac{\#\ features\ \ \in class\ i}{\#\ total\ \ features} \right\rceil \tag{5.5}$$

A usual k-means clustering algorithm is run with the elements within each class. Note that, differently from previous steps, now we use only the appearance descriptors of all the features in that class. This step provides different number of visual words for each class. Each word has a representative appearance descriptor, and we add an approximate azimuth value computed as the average azimuth of all features assigned to that word.

In the resulting vocabulary larger classes receive more words than small ones. The class including non paired sets, which is usually big, will receive a large amount of words, which guarantee that we actually account for these marginal and unique scene elements.

Assigning words to a new feature using the created BoW

To classify a new feature, $\mathbf{X}_{query}$ into the discovered classes, it is compared with the words included in the vocabulary using the distance described in (5.2). In a similar manner to standard vocabularies, we assign the corresponding word i according to the nearest neighbor, but only if that distance is below a threshold, th_M.

$$i = \arg\min_{i \in [1,k]} (d(\mathbf{X}_{query}, \mathbf{X}_i) | d(\mathbf{X}_{query}, \mathbf{X}_i) < th_M), \tag{5.6}$$

By using this threshold we model the fact that a new feature could not belong to any of the modeled classes and therefore fall into the non paired sets class.

5.4 Analysis of the performance of the hierarchical vocabulary

This section evaluates all steps of our proposed approach and evaluates the performance and properties of the obtained visual vocabulary.

5.4.1 Experimental Settings

Datasets

We validate our method with a dataset acquired from a robotic platform at the AASS laboratories and offices in Örebro University, Sweden. It includes two image sequences of two different trajectories around the same environment: 158.5 m (1879 frames) and 164 m (2142 frames). They were acquired at different days and follow different trajectories but they present certain overlap to allow place recognition experimental validation. The acquisition was performed at 30 frames per second, and the images have a resolution of 768x768 pixels. As explained previously about our assumptions, the camera has been set pointing to the ceiling. Therefore the objects that appear are mostly light sources, windows and signs. Fig. 5.13 shows some sample images of this dataset. This dataset is available on-line [2].

A second dataset is used for additional qualitative evaluation of the method. This small sequence has been acquired in a trajectory of about 10 m on a different indoor environment, traversing a corridor. The purpose of this sequence is to further analyze the correspondence between classes and real objects in a different scenario than the main dataset used.

Performance measurements

The proposed method is evaluated following three different criteria:

[2]Dataset: `http://aass.oru.se/Research/Learning/datasets.html`

- *Accuracy:* it evaluates the accuracy of the vocabulary to classify new features into the discovered classes. Total and average class accuracy of the classification, A_{Total} and $A_{Average}$ are respectively computed as:

$$A_{Total} = 100\frac{\#\ correct\ classification}{\#\ test\ features} \tag{5.7}$$

$$A_{Average} = \underset{\forall i}{mean}\left(100\frac{\#\ correct\ classification_i}{\#\ test\ features_i}\right) \tag{5.8}$$

  the i index represents the classes. $A_{Average}$ averages the accuracy achieved in the classification of each class separately.

- *Normalized inverse pixel deviation:* it evaluates the similarity of the key-points patches included in each class. This quality measurement is based on the standard deviation of the image patches of features that have been clustered together. Given class i, we define the class pixel deviation, S_i, as the mean of the standard deviation of the gray level of every pixel of the features patches included in class i:

$$S_i = \underset{\forall(x,y)}{mean}(\underset{\forall j\in i}{std}(I_j(x,y))), \tag{5.9}$$

  where $j \in i$ represents all the patches of the features included in the class i, $I_j(x,y)$ is the gray level of pixel (x,y) from the patch of feature j, (x,y) values are limited to the size of the patches (32x32 pixels in our case) and $std()$ is the standard deviation.

  We define the normalized inverse pixel deviation for each class, S'_i:

$$S'_i = 1 - \frac{S_i}{S_{max}} \tag{5.10}$$

  where S_{max} is the maximum pixel deviation. According to (5.9) and (5.10), more meaningful classes would have higher S'_i values. Next section analyzes the evolution of S' that is the mean value normalized inverse pixel deviation of all the classes.

- *Intra-class distance:* it evaluates the similarity between all the sets of key-points included in each class. Distance between all the sets of key-points is computed using equation 5.4. The intra-class distance is the mean of these distances. Lower values of this distance mean more compact clusters, where the sets grouped are more similar.

Normalized inverse pixel deviation and Intra-class distance both model how similar are the elements grouped under the same class label. However, the first one computes distances between key-point patches, just the key-points appearance; while the second computes distances between sets of key-points using tracking and viewpoint information together with the SURF descriptors.

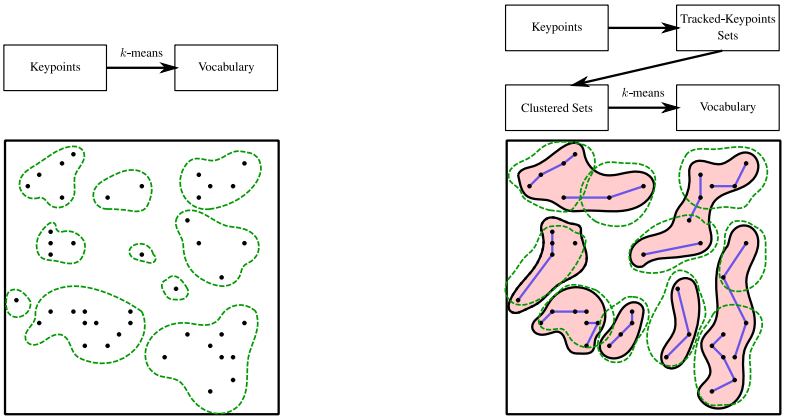


Figure 5.3: Differences between usual k-means vocabulary and our hierarchical vocabulary building process. k-means clusters directly in the feature descriptor space. Our enhanced vocabulary, first groups by tracking, then clusters tracked key-point sets into classes, and finally clusters with a k-means within each of the found classes to create the final visual words.

Comparison with k-means vocabulary

Along this experimental section, we compare the properties of our proposed vocabulary with those of the usual k-means vocabulary. We have choose this technique as baseline for the visual vocabularies. The usual k-means vocabulary is built by clustering all the features in the training data, using the k-means algorithm. It has shown good performance in plenty of computer vision applications, but one of its drawbacks is that neither semantic nor conceptual information is considered when building in the words. In contrast, the hierarchical vocabulary presented in this work merges in the same class key-points that are likely to come from the same scene element. Fig. 5.3 shows the basic differences between both approaches. As a result, the main difference between these two types of vocabularies is that our vocabulary construction approach discovers semantic relationships between the visual words, grouping them in the resulting classes. Our output model is not only a set of visual words, but also a set of classes that represent elements of the environment.

5.4.2 Analysis of the clustering of tracked key-points sets

Figure 5.4 shows the evolution of the main parameters of the clustering results for different configuration of the two evaluated clustering algorithms. Fig. 5.4(a) shows the mean value of S' (equation 5.10), Fig. 5.4(b) shows the mean intra-class distance of the clusters, Fig. 5.4(c) represents the number of clusters created and Fig. 5.4(d) shows the number of non paired sets grouped together in the last step of the clustering. We can observe how S', intra-class distance and the number of non paired sets have similar behavior for both methods and values of th_S and ε in the interval 0.05 to 0.35. The main

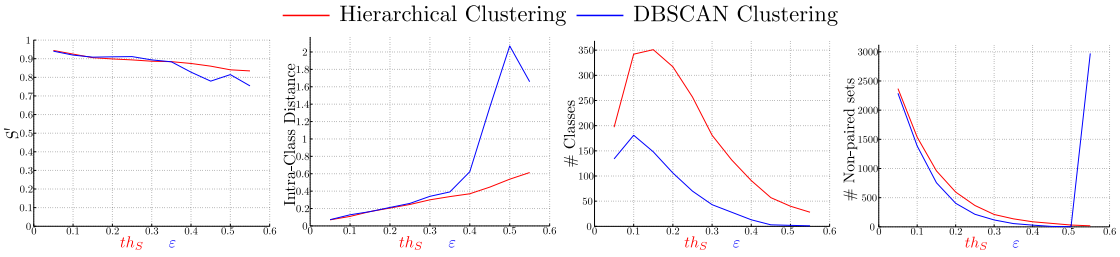


Figure 5.4: Comparison of the normalized inverse pixel deviation, intra-class distance, number of created classes and number of non paired classes for both clustering methods: Hierarchical CLustering (red) and DBSCAN (blue). th_S and ε are equivalent parameters for both clustering methods respectively. (Best seen in color).

difference in this interval is the number of clusters created: DBSCAN creates less clusters than the Hierarchical Clustering. In DBSCAN two elements can be clustered together if they are density reachable, even if their distance is high. In contrast, in our modified version of the Hierarchical Clustering, to pair two elements the distance between these elements must be lower than th_S. The requirement to cluster elements is more relaxed for DBSCAN than for Hierarchical Clustering, so more sets are clustered together and less clusters are created. For ε higher that 0.35, very few clusters are created with DBSCAN, below 50 clusters. Then dissimilar sets are clustered together affecting the quality of the result as can be seen in the evolution of S' and the intra-class distance.

5.4.3 Influence of th_S, ε and K parameters in the performance of the resulting vocabulary

These experiments analyze the influence of the clustering method parameters (th_S and ε) and the number of words in the vocabulary (K), in the performance of our approach. The effect of varying th_S and ε in the clustering quality is shown in previous Fig. 5.4.

For this analysis we randomly split one of the sequences and use 70% as training, and 30% as test data. The clustering of sets of tracked key-points into classes is performed on all the sets included in the whole dataset to set a ground truth class assignment for all the key-points (training and test ones). The vocabulary is built using only the training data. The use of just one sequence allows us to have the class ground truth for the test images. Total and Average per class accuracy of the results of classifying test data using this vocabulary are shown in Fig. 5.5. Total accuracy is the percentage of correct classifications respect the total number of key-points classified. The average per class accuracy is the mean of the accuracy computed for each class.

Figures 5.5(a) and 5.5(b) correspond to the modified Hierarchical Clustering. In this case for higher values of th_S A_{Total} increases, while $A_{Average}$ remains constant. The reason of this behavior is that when th_S increases large classes get even larger by clustering dissimilar sets of tracked key-points. A_{Total} increases due to the good performance of

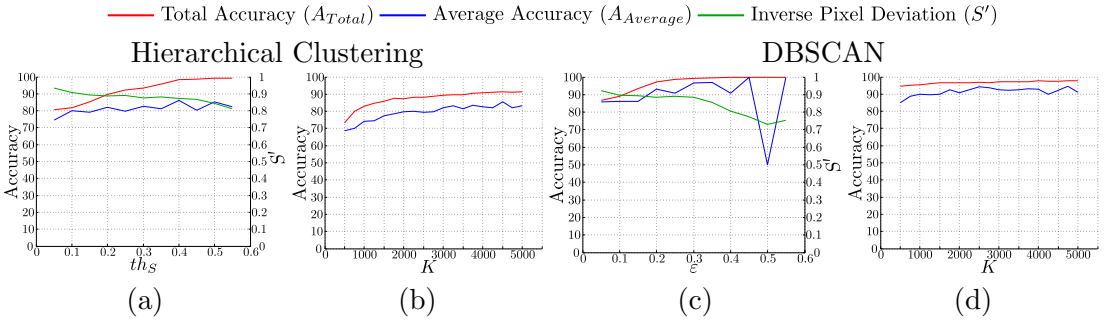


Figure 5.5: Influence of parameters, th_S, ε and K in the performance and quality of the vocabulary. Plots show the evolution of total (red) and average (blue) accuracies and normalized inverse pixel deviation (green). Figs. (a) and (c) show the results obtained for values of th_S or ε between 0.05 and 0.55, $k = 3000$. (b) and (d) shows the accuracy results for values of k in $[500, 5000]$, with the parameters of each clustering (th_S and ε) equal to 0.2.

large classes that are more common, but $A_{Average}$ remains constant due to the poor performance of the small classes. In the same figure we can observe the evolution of S'. As expected, when th_S grows, the quality of the classes, represented by S', decreases.

Figures 5.5(c) and 5.5(d) correspond to DBSCAN clustering. We can observe how the A_{Total} and $A_{Average}$ grow with ε while the S' decreases. The high values observed for the total accuracy, 100% when ε is greater than 0.4, are effect of the low number of clusters created. The chance of classifying correctly a key-point when most of the features are part of the same cluster is very high. Values of A_{Total} and $A_{Average}$ remain almost constant when k grows, those values are in all cases higher than the ones for the modified Hierarchical Clustering. The reason of the better accuracy is that the first class created by DBSCAN clustering is too big, as we show later in subsection 5.4.5. Most of the training key-points fall in this first class what makes easy to classify test key-points under that class. However this class is compound by dissimilar sets of tracked key-points. High values of K produce large vocabularies that fit better the feature space. However, one of the goals of using vocabularies is to reduce the number of elements included in the model, so small K values are preferred.

5.4.4 Influence of the robot motion in the detection of key-point classes

This experiment analyzes the suitability and correctness of our vocabulary to model the environment. We classify key-points from a test sequence, not used to build the vocabulary, into the classes discovered by our approach. We can observe how key-points that were generated by the same scene element but seen from different viewpoints are correctly identified in most of the cases.

Figure 5.6 shows examples of key-points classified along different test sub-sequences. First row of the figure shows the trajectory followed by the robot in each sub-sequence, which include rotation and translation. Each column of the figure shows key-points

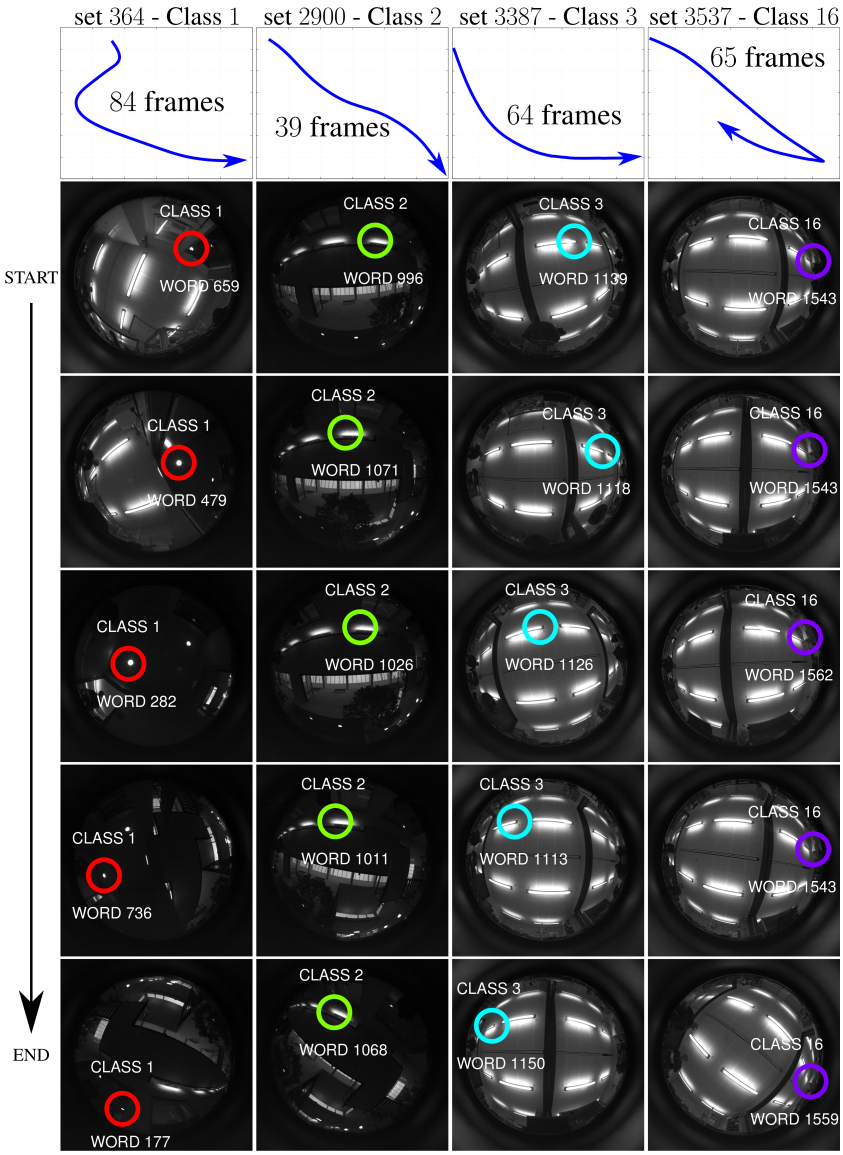


Figure 5.6: Key-point classification into words and classes as the robot moves. The plots in the first row represent the trajectories followed by the robot while acquiring the frames. The trajectories include rotation, translation and combinations of both. Each column shows how the key-points of a set of key-points are individually classified as being of the same class. All the key-points on a set correspond to the same element of the scene that has been tracked. (Best seen in color).

detected in those frames, together with the word assigned to that key-point and the corresponding class. Note that as the key-point appearance and image position varies, it is assigned to different words but is still classified under the same class. This validation

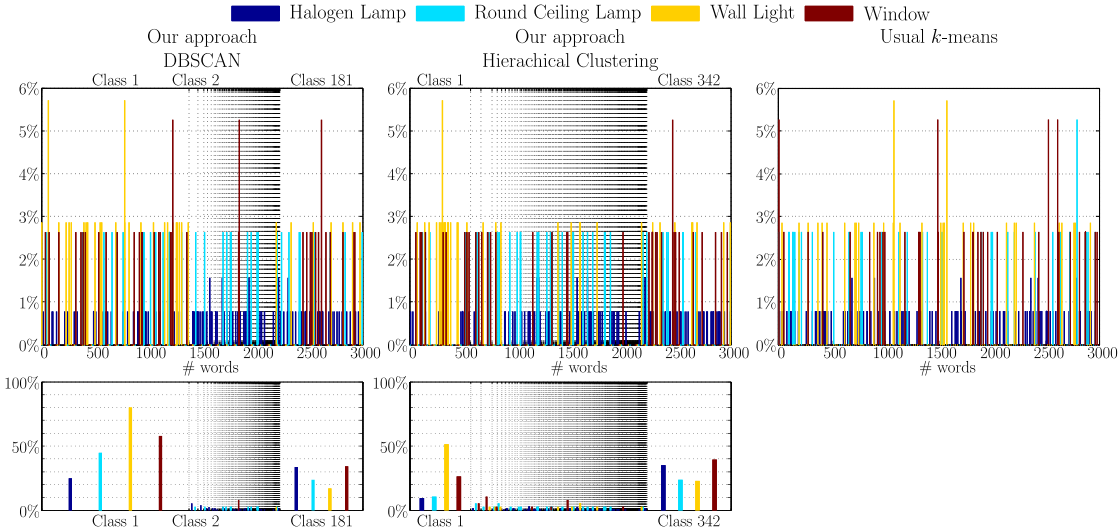


Figure 5.7: Histogram of the percentage of key-points of each object that have been quantized within each word (top) and class (bottom), using our approach with DBSCAN and with the modified Hierarchical Clustering. The baseline compared is a usual k-means vocabulary, where we can only analyze objects per word (the additional organization of words into classes is part of our approach). For each word (or class), we see 4 bars, one per considered object class. x-axis shows the words and the classes width for our approach and both clustering methods. Vertical dashed lines represent the number of words assigned to each class. (Best seen in color).

demonstrates for a given scene element, i.e., one of our classes, how the different words encode the element appearances and viewpoints.

5.4.5 Analysis of the object information included in the vocabulary

One interesting property of our method is that the classes created by clustering sets of tracked key-points are related to scene elements of the environment. Large classes represent highly repeated sets, and small classes represent less common but more discriminative scene elements. We want to analyze if these discovered classes actually correspond to objects or parts of the environment (note that every step of our method is unsupervised, so we do not have any concept name associated with any of the classes).

For this experiment we use both sequences, one for obtaining the vocabulary and the second one for testing. In both sequences we have labeled manually 10% of the images with bounding boxes around the four most repeated objects in the environment. Since the camera is pointing to the ceiling, the objects that appear are mainly different kinds of lamps and windows. The labels used are: Halogen Lamp, Round Ceiling Lamp, Wall light and Window (see Fig. 5.13).

Figure 5.7 shows histograms that count the percentage of key-points of each object that have been quantized with each word and each class. The best results are obtained

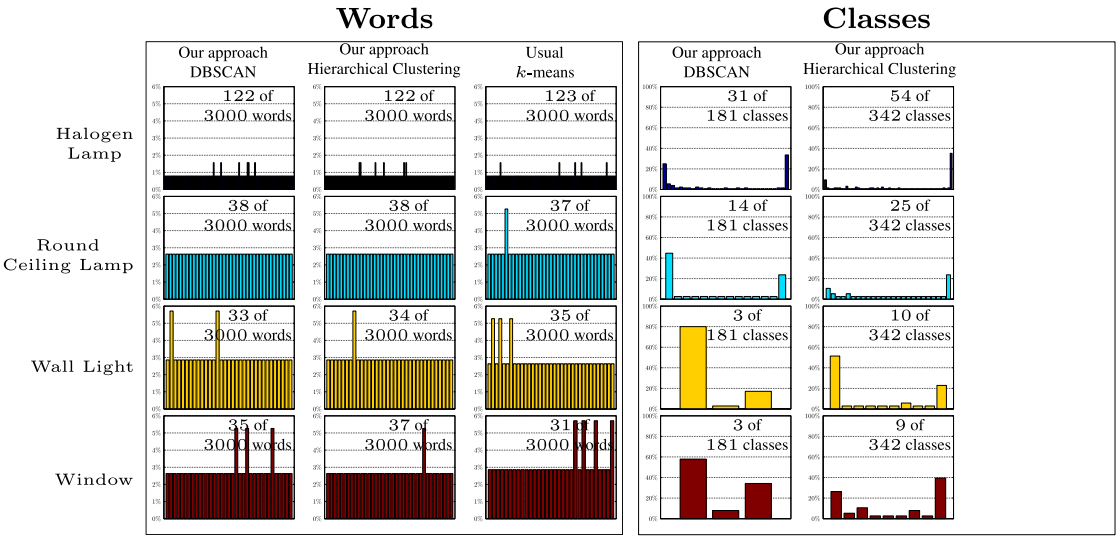


Figure 5.8: Histogram of the percentage of key-points of each object that have been quantized within each word and class. Just words and classes where the percentage is bigger than zero are considered. Results for each object and each clustering method, approach with DBSCAN and with the modified Hierarchical Clustering and usual k-means vocabulary, are shown. (Best seen in color).

for our method with modified Hierarchical Clustering, showing a cleaner grouping of words from the same object for some of the discovered classes.

Vertical dashed lines show the number of words created for each class. As explained in subsection 5.3.3, the number of visual words assigned to each class depends on the number of key-points included in that class. When DBSCAN is used in our approach the first class is too big, so most of the object key-points are classified under those words (about 80% of Wall Light features, 60% of Window, 45% of Round Ceiling Lamp and 25% of Halogen Lamp). In the case of Hierarchical Clustering first class is smaller, and is more representative of the Wall Light features (50%). Using a usual k-means vocabulary (c), there is no relation between words and scene objects and words generated from the same objects are not related in any way.

Figure 5.8 shows similar information to Fig. 5.7. However in this case each object histograms are shown in different plots, and just words or classes where the count of key-point objects is bigger than zero are shown. We can observe how the vocabularies behave in a similar way at the words level. Objects are spread in the same number of words. However at class level, we find two main representative classes for the key-points of each object.

Table 5.1 shows numeric values that support the analysis of the relation between scene objects and the classes created by our vocabulary. It shows the entropy values for the classification of the objects into classes and words. Entropy, measures the unpredictability of a random variable. In our case the random variable is the classification of

Table 5.1: Normalized entropy of the object classification into classes or words.

Words

Object	Our approach DBSCAN	Our approach Hierarchical Clustering	Usual k-means
Halogen Lamp	0.598	0.598	0.599
Round Ceiling Lamp	0.454	0.454	0.450
Wall Light	0.434	0.439	0.424
Window	0.441	0.450	0.441
Average	**0.482**	**0.485**	**0.479**

Classes

Object	Our approach DBSCAN	Our approach Hierarchical Clustering	Usual k-means
Halogen Lamp	0.460	0.524	–
Round Ceiling Lamp	0.356	0.497	–
Wall Light	0.840	0.266	–
Window	0.112	0.290	–
Average	**0.275**	**0.394**	–

a key-point generated by a scene element into words or classes. If the key-points of an object are always classified in the same word, the entropy will be zero, so the object key-points and that word are highly related. However, if an object is classified in a lot of words the entropy value will be high. To compare the entropy of different vocabularies, we use the Normalized Entropy:

$$Normalized\ Entropy = \left(-\sum_{i=1}^{n} f_i \log_2(f_i) \right) \log_2(n) \tag{5.11}$$

where, for each object, f_i is the proportion of occurrences of that object in the class or word i and n is the number of classes or words.

Analyzing the classification at word level, the three vocabularies behaves similarly, the normalized entropy values are similar. However, looking at the values for the classification into classes the values are lower. This tendency is clear analyzing the mean of the normalized entropy for the four objects. We obtain values about 0.48 at words level and values of 0.275 for our approach with DBSCAN and 0.394 for our approach with Hierarchical Clustering. Again, for the vocabulary built using usual k-means there are no classes. Note that the normalized entropy does not inform about which classes are related with each object, it just informs if an object is related with more or less words or classes of the model.

Object-class correspondence

Fig. 5.9 shows some examples of the correspondence between the classes created by our vocabulary and the elements of the environment. For this experiment we have run our

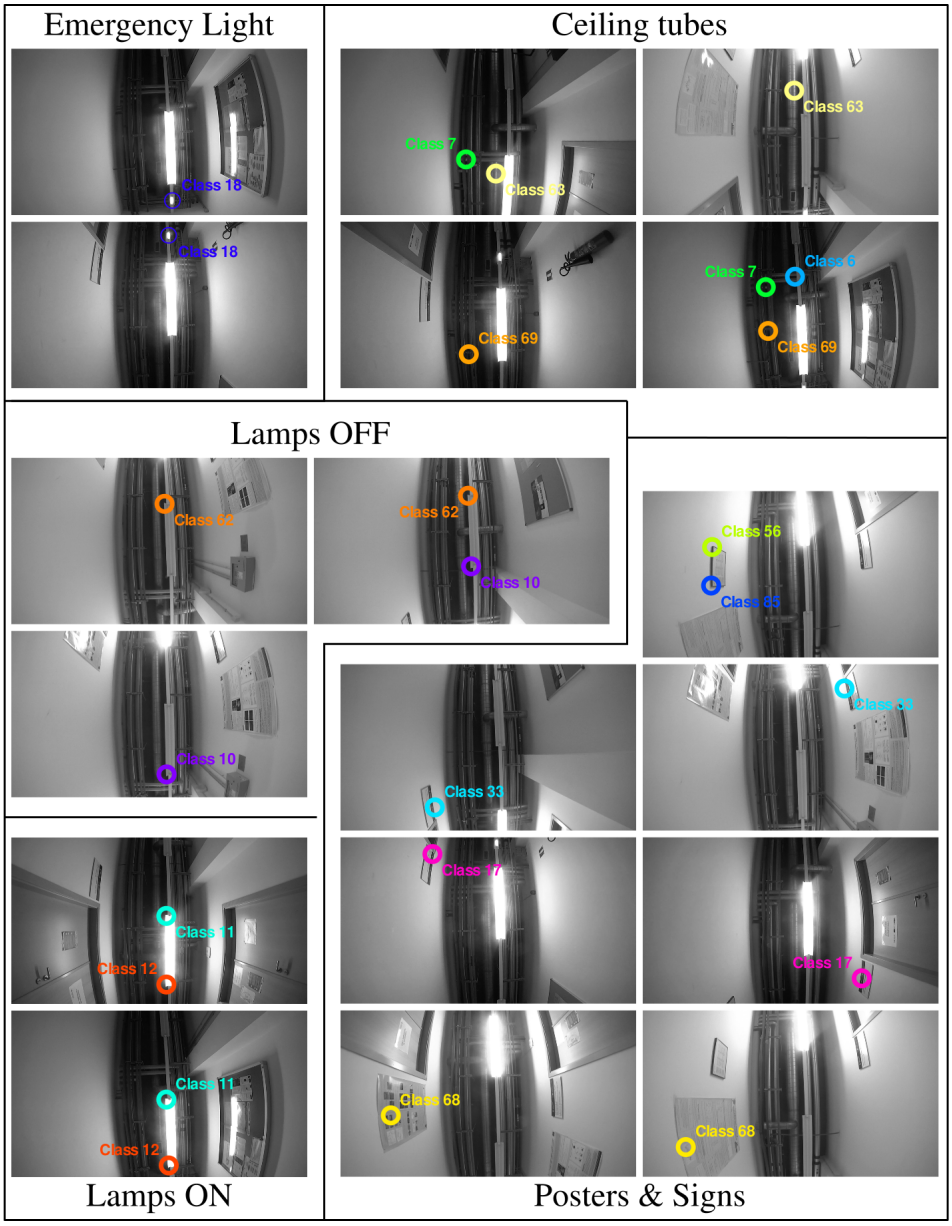


Figure 5.9: Examples of classes created by our method that correspond to real scene elements. Most of the classes are detected on various frames, and on different sides of the images. (Best seen in color).

vocabulary creation method in the small sequence described in the previous section. Figure shows examples of classes correspondence with objects in different frames of the sequence. Again, most of the objects detected are lamps, halogen lamps (on and off) and emergency lights. Ceiling tubes, posters and labels are also detected. For most of

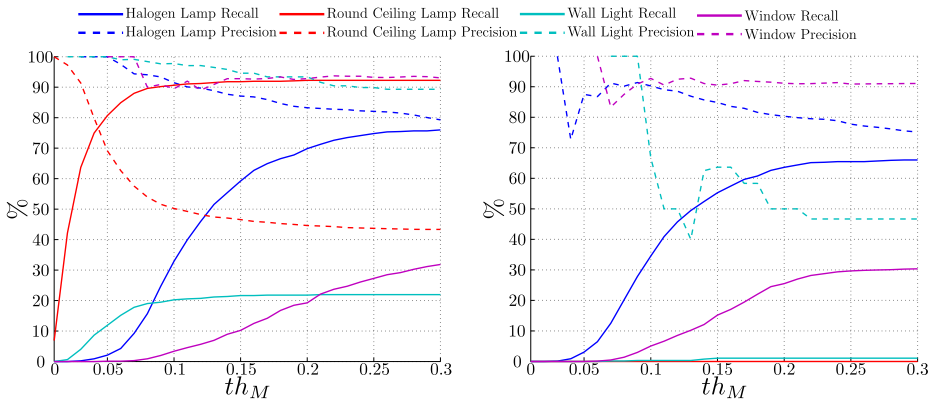


Figure 5.10: Precision (dashed line) and Recall (solid line) for the detection of real objects as the matching threshold th_M varies. We can appreciate the much better performance including the azimuth in the descriptor together with SURF (a) that using only SURF (b).

the classes occurrences in different frames are shown. Note that classes 33 and 17, that correspond to wall sign, are detected in both sides of the corridor.

Scene objects key-point detection

We want to quantify how our method classifies new occurrences of objects. In the training data, we select which classes correspond to real objects (we determine that a class correspond with an object label if $> 50\%$ of the class occurrences fall inside the bounding boxes of that object). The test key-points detected as being of a class related to an object are labeled as being generated by that object. Fig. 5.10(a) shows precision and recall when classifying features into object labels for different values of the similarity threshold, th_M. Precision and recall are computed for each object as follows

$$Precision = 100\frac{TP}{TP+FP} \qquad Recall = 100\frac{TP}{TP+FN} \tag{5.12}$$

where, for a determined object, TP (True Positives) counts the number of key-points of that object correctly classified, FP (False Positives) counts the number of object key-points not classified as being of that object, and FN (False Negatives) counts the number of key-points not corresponding to that object but classified as being of that object.

The highest recall, above 90%, was achieved for the detection of Round Ceiling Lamps which appear in almost all the rooms traversed during the sequence. However, the precision is low for this class, below 50% for higher values of th_M. For the rest of objects, precision is above 90%. For Halogen Lamps recall reaches almost 80%, while for Wall Lights and Windows recall values are lower, about 20% and 30% respectively. The low

recall values for the detection of key-points generated by Wall Lights and Windows can be explained looking Fig. 5.7(b). There we can see how both objects key-points appear in the same classes. Those objects appear in the same parts of the environment, so the key-points appear in the same part of the image and share similar azimuth values. Also, they have similar appearance. This causes that some Window key-points appear in the representative class of Wall Lights (class 1) and vice-versa.

Inclusion of the Azimuth values

Fig. 5.10(b) shows the same plot that Fig. 5.10(a), but in this case the azimuth has not been used at all in the process. While the results are similar for Halogen Lamps and Windows, the accuracy for Wall Lights decreases from around 20% to 2% when we do not use the azimuth. Even more dramatic is the change to detect Round Ceiling Lamps: without the azimuth value, no Round Lamps are detected. In our case, the image descriptors of features around the Round Ceiling Lamps are similar to the descriptors created by other entities of the environment, so the appearance descriptor is not enough to distinguish this object.

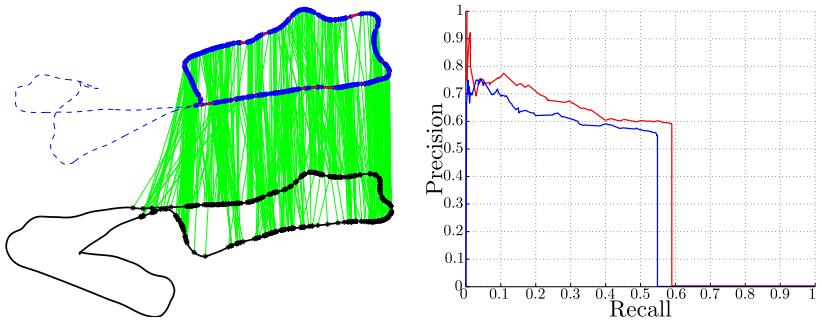
5.5 Applications using the proposed vocabulary

This section describes two possible applications of the vocabulary presented in this work. Place recognition and object detection are studied here, however other applications can benefit of enhanced vocabularies.

5.5.1 Place Recognition

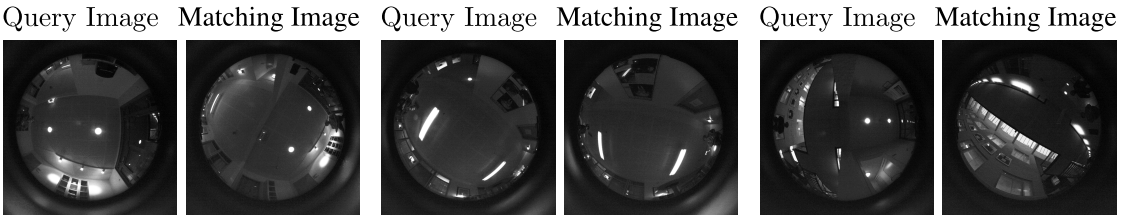
One of the applications where BoW representations have been widely used is place recognition. We show the performance of our vocabulary in an example of this application and compare it with the performance of a usual k-means vocabulary. The trajectories of the two sequences included in our dataset are shown in Fig. 5.11. The available ground truth is raw odometry, not very accurate, therefore not all the test images can be automatically evaluated for the place recognition tests. We only use as test the part of the test sequence where correspondences can be established according to the ground truth.

According to BoW representations, images of both sequences are represented by a histogram of the number of occurrences of each word in the image. These histograms are normalized with the total number of words in the image, and the image similarity is obtained according to these histograms distance. The localization of a test image is assigned as the localization of the most similar reference image found. If the training nearest neighbor is within a similarity distance larger than a threshold ($th_{HistDist} = 0.15$ in our case), we consider that the location is unknown or uncertain, therefore no answer is given. The localization will be considered correct, if the match and the test odometry positions are within 7.5 m (due to the accuracy of the ground truth, lower acceptance thresholds give wrong evaluation results). Fig. 5.11 shows the results of the



(a) (b)

Figure 5.11: Comparison our approach and k-means vocabulary for Place Recognition. (a) shows the odometry of the two sequences used: Test (blue) and Train (black). The part of the test trajectory not considered due to the noisy ground truth is shown as dashed line. Correct localization results are shown as green lines, and errors are shows with red points in the test trajectory. (b) shows the Precision-Recall curves of both vocabularies, our enhanced vocabulary (red) and k-means vocabulary (blue). These curves have been obtained varying $th_{HistDist}$. (Best seen in color).



(a) (b) (c)

Figure 5.12: Examples of correct place recognition where the robot location is rotated significantly between test and training time. Left images show the query test image, and right images show the corresponding image in the training data.

experiment, including the visualization of the tests and trajectories and a Precision-Recall curve for the results obtained where we can appreciate that our approach shows comparable results than usual k-means vocabulary (our method slightly outperforms the k-means vocabulary for the threshold range analyzed). Therefore, the new proposed process clearly gains important semantic information without discriminative power loss for place recognition.

Finally, we show some examples of correct place localization when the robot orientation is different between train and test, Fig. 5.12. In these cases, our model was robust enough to evaluate those images as the same location.

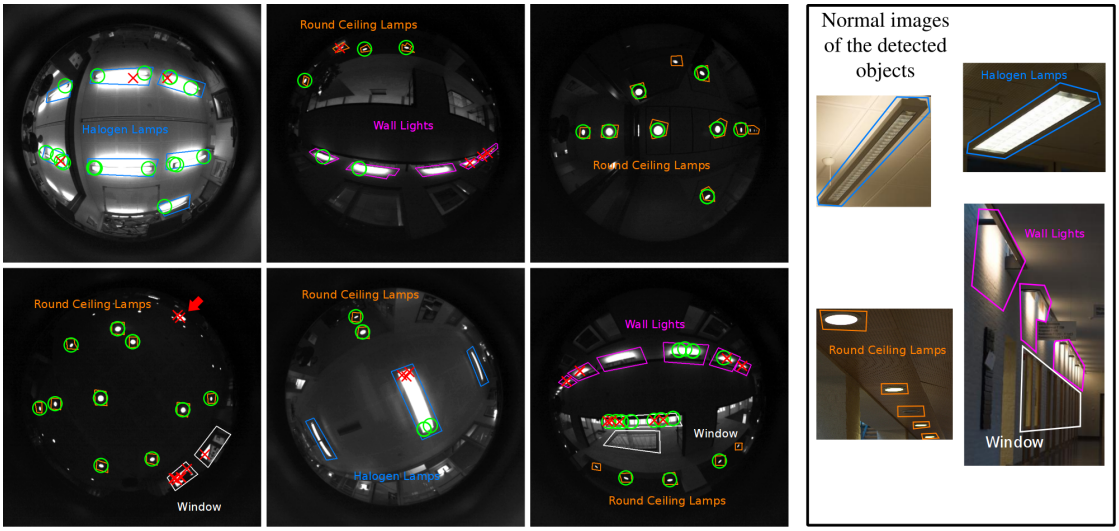


Figure 5.13: Examples of objects found in the dataset. Green circles mark correct assignments and red crosses show incorrect ones. The colored shapes show the labeled areas: Round Ceiling Lamps (orange); Halogen Lamps (blue); Wall Lights (magenta); Windows (white). Matching threshold is set to 0.15. For clarity, some conventional images of the objects in the environment are shown on the right of the figure. (Best seen in color).

5.5.2 Object Detection

In the previous section we show how the vocabulary represents some of the objects found in the environment. Fig. 5.13 shows the detection of objects in different frames. Correct matches are shown as green circles and errors as red crosses, the labels of the images are also plotted. In those examples we can see how most of the Round Ceiling Lamps are correctly detected. In the first image of bottom row the red arrow shows points classified as Round Ceiling Lamp that correspond to a different kind of lamp that newly appears in this room. We can also see how the areas labeled as windows are very dissimilar due to the objects seen through these windows and their different shapes.

To perform this experiment we need to relate objects to classes and words. This step, as explained in section 5.4.5, requires the sequence labeling so the process is not completely unsupervised.

5.6 Conclusions

We have presented a new method to create an enhanced visual vocabulary from an image sequence that learns semantic relationships between visual words of the working environment. The key elements of the method are using tracked scene points and including information of the key-points azimuth angles when building the visual words. Our approach is focused on long term robotics applications. With this purpose, we consider systems where the camera points to the ceiling, which facilitates the acquisition of more

stable and repetitive scene elements. Comparisons with the usual k-means vocabulary present our method as a richer alternative for the usual BoW approach to build a visual vocabulary. Our model provides more representative semantic information of the environment, including relationships between visual words and environment elements.

Chapter 6

3D Spatial Layout on video sequences

Intelligent autonomous systems need complex and detailed models of their environment to achieve sophisticated tasks. Vision sensors provide rich information and are broadly used to obtain or improve these models. The particular case of indoor scene understanding from monocular images has been widely studied, and a common initial step to solve this problem is the estimation of the 3D layout of the scene. Many previous approaches obtain the layout of a single image, however, this work addresses the problem of scene layout propagation along a video sequence. Our approach uses a Particle Filter framework to propagate the scene layout obtained using a state-of-the-art technique on the initial frame. We propose how to generate, evaluate and sample new layout hypotheses for the scene on each frame. The intuition we follow is that we can obtain better layout estimation at each frame through propagation than running separately at each image. The experimental validation is run on a publicly available indoor dataset and shows promising results for the layout computed using our approach, without the need of estimating accurate 3D maps. Additionally, our experiments demonstrate how this layout information can be used to improve detection tasks, in particular sign detection, by easily rejecting false positives.

6.1 Introduction

This section investigates the construction of simple indoor scene models given an image sequence. The models contain essential information about the environment structure that may allow us to better understand the image: detect the kind of area traversed (e.g. a corridor or a room); provide a rough 3D model of the scene for navigation assistance; or help to the detection and recognition of objects. Prior approaches demonstrate this applications and the fact that obtaining information about the 3D structure of the scene is a powerful tool to improve the accuracy of other tasks, such as object recognition [Hoiem et al., 2008]. Recent approaches [Hedau et al., 2010, Lee et al., 2010, Bao et al., 2011] propose to solve together the problem of estimating the layout of the scene

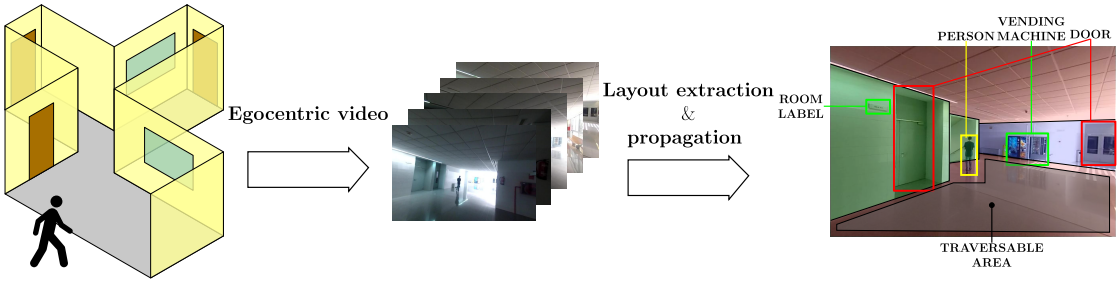


Figure 6.1: Our goal is to process video acquired from a wearable camera to obtain the 3D layout of the scene in each frame (Red: floor and ceiling; Green and Blue: walls from different orientations). This layout information is a strong prior to facilitate the detection of other scene details: persons, signs, doors or the traversable area.

and detecting the objects that appear there.

Even a basic 3D model of the scene provides useful information about the environment structure that facilitates automatic scene understanding. For example, it can help us identify the type of area traversed (e.g., a corridor or a room) or provide strong priors for detection and recognition of objects [Hoiem et al., 2008]. The goal of this work is to provide a basic 3D model of the environment traversed while recording a video (Fig. 6.1) to enhance the performance of more complex tasks.

Most of the layout estimation prior work assumes certain constrains, like the Manhattan World assumption, that allow to infer 3D scene models from 2D images. These approaches try to solve this problem for single images [Hoiem et al., 2005, Delage et al., 2006, Lee et al., 2009, Hedau et al., 2010]. In this work we aim to use the sequential constraints of a video sequence to provide a scene model at every step of the camera trajectory.

Other papers use sequential information to model the environment from a mobile camera. Most of these approaches rely on SLAM or Structure-from-Motion techniques, and enhance the 3D maps created by those techniques with semantic labels defining the environment structure [Flint et al., 2011, Tsai and Kuipers, 2012, Furlan et al., 2013].

The goal of our work is to provide semantic information about the layout of the scene traversed during the sequence. Our proposed approach propagates the estimated scene components by taking advantage of restrictions in the sequential data and prior knowledge on the projection of 3D man made environments in 2D images. We show how to achieve this task without the need to compute accurate camera motion or 3D maps of the environment.

6.2 Related Work

Recognizing the 3D scene structure of an environment captured in an image is a widely studied problem. Earlier studies, like the one presented by Hoiem et al. [Hoiem et al., 2007], proposed to learn appearance-based models of the scene parts (sky, floor, vertical

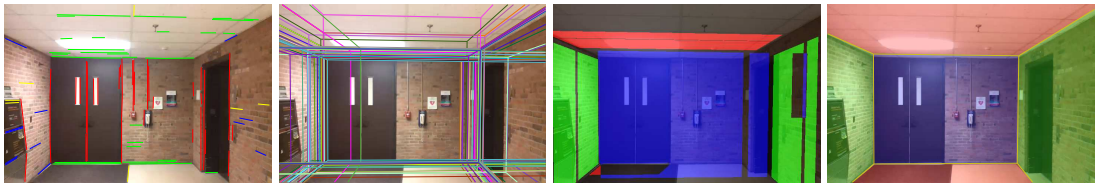
objects) and describe the scene geometry using these coarse labels. Later, Saxena et al. [Saxena et al., 2009] used Markov Random Fields to infer plane parameters, such as 3D location and orientation, for homogeneous patches extracted from the image. Both works are intended to solve the scene structure for general scenes.

We find specific approaches for indoor environments, where certain additional assumptions can be made. Delage et al. [Delage et al., 2006] proposed a dynamic Bayesian network model to find the "floor-wall" boundary in the images. They create 3D models of the scene assuming a Manhattan World [Coughlan and Yuille, 1999]. More recently, Lee et al. [Lee et al., 2009] presented a method to generate interpretations of a scene from a set of line segments extracted from an indoor image. Similarly, Hedau et al. [Hedau et al., 2009] model the scene as a parametric 3D box based on the detected lines. López et al. [López-Nicolás et al., 2014] solved the spatial layout in omnidirectional images, where the whole the scene can be modeled with just one image. Extending similar ideas to outdoor scenes, Gupta et al. [Gupta et al., 2010] propose to create physical representations of outdoor scenes where objects have volume and mass, and their relationships describe the 3D structure and mechanical configurations.

The papers described so far analyze the structure of a single image. However, if we consider images of a video sequence we can propagate the information already obtained about the scene and get better and more robust results. This is the idea exploited in this work. Acquiring sequential information is the usual scenario when working with mobile platforms, and the implicit spatio-temporal restrictions between consecutive frames can provide both efficiency and accuracy advantages by accumulating the information obtained across consecutive frames.

Most of the works which also take advantage sequential data to obtain better scene models, are based on SLAM or structure-from-motion techniques. Flint et al. [Flint et al., 2011] combined geometric and photometric cues to obtain their scene model from a moving camera. They integrate ideas from semantic reasoning in monocular images and 3D information obtained with structure-from-motion. Similarly, Furlan et al. [Furlan et al., 2013] proposed a method to estimate the 3D indoor scene layout from a moving camera. They pre-process the sequence to obtain the camera motion and 3D map of the environment. Tsai et al. [Tsai and Kuipers, 2012] described as well a method to create a model of the environment using images acquired from a mobile robot. Since they focus on a robot moving indoors they can adopt constraints about the camera motion and the environment. Also working with multiple images, but not in a sequence, Furukawa et al. [Furukawa et al., 2009] proposed how to reconstruct indoor environments from multiple photographs.

Attending how to propagate semantic information in video sequences using probabilistic frameworks. Badriranayanan et al. [Badrinarayanan et al., 2010] used a probabilistic graphical model. They are able to use pixel-wise correspondences from motion estimation, image patch similarities or semantical consistent hierarchical regions to propagate the labels. Vazquez et al. [Vazquez-Reina et al., 2010] presented the Multiple Hypothesis Video Segmentation method for unsupervised segmentation of video sequences. Rituerto et al. [Rituerto et al., 2011] focused on label propagation indoors using images acquired



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

Figure 6.2: Steps of the method presented in [Lee et al., 2009]: First, straight lines are detected in the image and grouped according to the scene vanishing directions (a). Then structure hypotheses are proposed (b), and the orientation map is computed (c). Finally, the hypotheses are compared against the orientation map. The best hypothesis is shown in (d).

from a mobile robot. They learn the appearance of the different regions of interest from some training examples and propagate them trough the sequence using a non-parametric model. Similarly, Hussain et al. [Raza et al., 2013] estimate the 3D structure of outdoor video scenes by computing different appearance, location and motion features.

Our work is also proposing a probabilistic framework to propagate semantic information in sequences, in particular we aim to propagate the 3D layout of the environment traversed by the camera. The initial frame layout is obtained using the state-of-the-art single image technique presented in [Lee et al., 2009], and we then propagate and update this information by making use of spatio-temporal restrictions and the new lines detected in each consecutive frame. As previously mentioned, SLAM or structure-from-motion steps are not needed.

6.3 3D Spatial Layout propagation framework

The objective of this work is to compute the 3D layout at every frame of a video sequence. We exploit the fact that consecutive frames in a video have certain spatio-temporal restrictions that constrain the possible variations in the scene acquired. Image lines are very informative, but their detection is noisy. By propagating the possible layouts, we improve the results and obtain a more robust estimation of the layout on each frame. As detailed next, we adopt a particle filter based strategy to track the posterior probability of the layout given all the observations up to the current frame.

Algorithm 2 presents the steps of our algorithm. I_t is the frame at time t and X_t is the layout state, compound by n layout hypotheses, $\mathbf{X}_t = \{x_1, x_2, \ldots x_n\}$. For the first frame, hypotheses are created using a single image algorithm. These hypotheses are evaluated and ranked as detailed in the following subsections, and the best one is selected as the solution for that frame. For next frames, new hypotheses (particles) are randomly generated depending on previous hypotheses and their evaluation score. Again, these hypotheses are evaluated in a similar manner and the best one is selected as the solution in each of the following frames.

Algorithm 2 Particle Filter algorithm.

Input: Video sequence: $\mathsf{I}_t | t = 0 \ldots \#$ frames
Output: 3D Scene Structure Layout: $bestHyp_t$
  $\mathbf{X}_0$ = generateHypothesisFromImage(I_0)
  $\mathbf{p}_0$ = evalHypotheses($\mathbf{X}_0$, I_0)
  for $i = 1 \ldots \#$ frames **do**
    $\mathbf{X}_t$ = sampleNewHypotheses($\mathbf{X}_{t-1}$, $\mathbf{p}_{t-1}$)
    $\mathbf{p}_t$ = evalHypotheses($\mathbf{X}_t$, I_t)
  end for

This propagation method can be used with different single-image techniques. We study the use of two different approaches.

First, we have integrated the method proposed by Lee et al. [Lee et al., 2009]. This approach generates interpretations of the scene structure given a set of straight lines detected in a single image. Their approach proposes several physically valid scene structures that are then validated against an orientation map to find the best fitting model. The code of this approach is available on-line [1], including a basic model for the environment with three walls: left, middle and right.

To model more complex environments, we adapt and integrate the single-image technique proposed by López et al. [López-Nicolás et al., 2014]. This proposal computes the scene layout from an omnidirectional image. Authors propose to start the process by looking for a basic scene layout: a rectangular room around the camera. Once this layout has been detected they proceed to expand it by looking for plausible walls and corners. The method uses floor points as base to build and expand the hypothesis.

A common point of both single-image techniques is that they are based on line detection. They need to compute lines and vanishing directions of the observed scene in each image. In each frame, the image lines are detected using a Canny edge detector (Kovesi [Kovesi, 2000] Matlab toolbox). The vanishing points are detected following the method presented by Rother [Rother, 2002].

Next sections describe the particularities of both methods. From this point we are going to consider the method in [Lee et al., 2009] as the Baseline model method, and the adaptation of the method in [López-Nicolás et al., 2014] as the Complex model method.

6.3.1 Creating and propagating Baseline model hypotheses

The single image algorithm proposed by Lee et al. [Lee et al., 2009] generates interpretations of the scene structure given a set of straight lines detected in a single image. This approach proposes several physically valid scene structures that are then validated against an orientation map to find the best fitting model.

[1] Code: `http://www.cs.cmu.edu/~dclee/code/index.html`

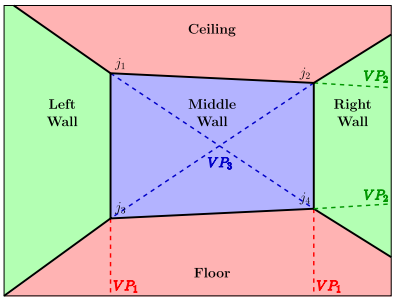


Figure 6.3: Scene model used to create the scene structure hypotheses. It includes three walls, floor and ceiling, and it is represented by four junctions: j_1, j_2, j_3 and j_4. Despite of the simplicity of the model, many indoor environments fit this configuration.

Building first frame room hypotheses

Our method uses the single image algorithm proposed by Lee et al. [Lee et al., 2009] to create layout hypotheses on the first frame. It generates interpretations of the scene structure given a set of straight lines detected in a single image. The single image technique creates several physically valid scene structure hypotheses that are then validated against an orientation map to find the best fitting model.

The authors adopt the Indoor World model that combines the Manhattan World assumption [Coughlan and Yuille, 1999] and a single-floor-single-ceiling model. This model applies to most indoor environments and introduces symmetry between ceiling and floor shapes, something useful when the floor-walls boundaries are occluded.

The process to find the building structure includes three steps: line segments and vanishing points are found, model hypotheses are proposed, and each hypothesis is tested against an orientation map (Fig. 6.2). After line detection and vanishing point estimation, the generation of hypotheses is made based on the model showed in Figure 6.3. This model includes three walls: left, middle and right, floor and ceiling, and it is defined by four line junctions.

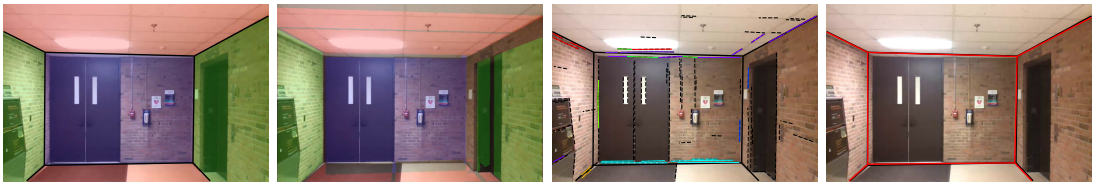
Layout parametrization

The model used to define the 3D scene layouts is composed by three walls (left, middle and right), and symmetric floor and ceiling (Fig. 6.3). This model can be parametrized by the image coordinates of the four junctions defining the middle wall (j_1, j_2, j_3 and j_4) and the directions of the scene vanishing points (VP_1, VP_2 and VP_3).

$$x_i = \{j_1, j_2, j_3, j_4, VP_1, VP_2, VP_3\} \tag{6.1}$$

Hypotheses evaluation

The evaluation of the hypotheses is performed on every frame. For all the images, lines and vanishing points are computed and used to evaluate the compatibility of the layout



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

Figure 6.4: Evaluation of the hypotheses. (a) shows the layout and the orientation map computed from it and (b) shows the orientation map computed from the observed lines. Both orientations maps are compared to compute S_{omap}. (c) shows all the lines supporting the model, S_{lines}, and (d) the layout hypothesis (black) and the closest observed layout (red) computed using the lines observed, S_{model}. Best seen in color.

hypotheses. We define three measurements for this evaluation computed for each layout hypothesis x_i:

- *Orientation map:* The orientation map is presented in [Lee et al., 2009]. It expresses the local belief of region orientations computed from line segments (Fig. 6.4(b)). This orientation map, $omap(\mathsf{l}_i)$, is compared with the orientation map defined by the hypothesis being evaluated, $omap(x_i)$ (Fig. 6.4(a)). The evaluation score is computed as the number of pixels where the orientation of both maps is the same divided by the total number of image pixels, $nPix = width \times height$

$$S_{omap\ i} = \frac{\sum_{k=0}^{nPix} omap(\mathsf{l}_i)_k = omap(x_i)_k}{nPix} \tag{6.2}$$

  where k is the pixel index. This score is the only evaluation used in [Lee et al., 2009] where the highest S_{omap} gives the chosen solution.

- *Number of lines supporting the model:* This evaluation measures how many of the observed lines support the hypothesis being evaluated. The layout parametrization used defines model lines delimiting the layout areas (Fig. 6.4(c)). To evaluate how a hypothesis fits the new observation, we look for lines parallel and close to these model lines. The score of this evaluation is computed as the number of lines supporting the model divided by the total number of lines detected.

$$S_{lines\ i} = \left(\frac{\#\ supporting\ lines}{\#\ total\ lines} \right)_i \tag{6.3}$$

- *Distance to the closest layout obtained with new observed lines:* This last evaluation scores a propagated layout hypothesis, x_i, by first computing the closest lines to this layout in the current image, determining a layout based on these lines, x_{obs}. Then both layouts are compared. If the propagated hypothesis is correct, it should be supported by a layout generated by the current frame (Fig. 6.4(d)). The

distance between layouts, $d(x_i, x_{obs})$, is computed as the mean distance between the junctions, j_k, of both layouts. The score is computed as follows:

$$S_{model\ i} = \frac{1}{1 + d(x_i, x_{obs})} \quad \text{where} \quad d(x_i, x_{obs}) = \underset{k=1...4}{mean}(||j_k, j_{k\ obs}||) \tag{6.4}$$

The three scores are used together as evaluation: $S_{total\ i} = mean(S_{omap\ i},\ S_{lines\ i},\ S_{model\ i})$.

Sampling new hypotheses

A new set of hypotheses is created by sampling from the hypotheses of the previous frame and their evaluation score. For each hypothesis, a score has been computed, $S_{total\ i}$. The number of new hypothesis sampled from each previous hypothesis depends on this score. The probability of generating a new hypothesis, x'_i, from previous hypothesis x_i is $p_i = S_{total\ i}$. New hypotheses are created randomly, high scores will generate more new hypotheses since they are more probable, and low scores hypotheses will receive few samples or even disappear.

The model used parametrizes the layout as four coplanar junction points and the vanishing points of the scene. Given the camera motion, a homography relates the projection of the coplanar junctions between frames and the vanishing points are related by the rotation matrix. To create a new hypothesis from a previous one, we assume a random motion of the camera, with zero velocity and random noise in camera translation and rotation. Random rotation and translation are created, R and $\mathbf{t}$, from 3 random angles ($\mathsf{R} = f(roll, pitch, yaw)$ and 3 random translations ($\mathbf{t} = [t_x, t_y, t_z]^T$). From hypothesis x_i, sampled hypothesis x'_i will be related by the random R and $\mathbf{t}$. The junctions are related by a homography, H:

$$j'_k = \mathsf{H} \cdot j_k = (\mathsf{R} - \frac{\mathbf{t}\ \mathbf{n}^T}{d}) j_k \quad (k = 1 \dots 4) \tag{6.5}$$

where $\mathbf{n}$ is the normal of the plane where the junction points lie and d the distance between the camera and the plane. We assume d distance as unitary so the scale of the random translation t is defined by the real distance to the plane. The vanishing points are related by the rotation matrix:

$$VP'_k = \mathsf{R} \cdot VP_k \quad (k = 1 \dots 3) \tag{6.6}$$

Every time that a hypothesis is sampled, random R and $\mathbf{t}$ are created.

6.3.2 Creating and propagating Complex model hypotheses

We adapt now the hierarchical method proposed in [López-Nicolás et al., 2014] to compute the scene layout from an omnidirectional image. This method starts the building process by looking for a basic scene layout: a rectangular room around the camera. Once this layout has been detected it is expanded by looking for plausible walls and corners. The method uses floor points as base to build and expand the hypothesis.

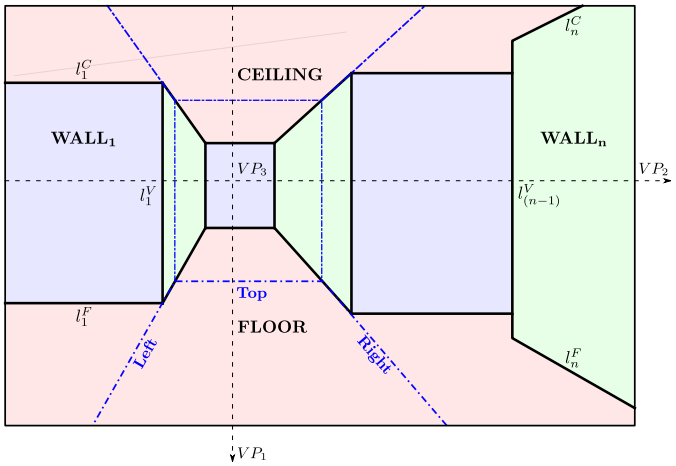


Figure 6.5: Scene layout model. The *colored areas* in the image encode the different planes of the scene, according to the surface orientation (red for horizontal planes and green and blue for vertical planes). The *black lines* define the scene structure and are grouped as floor (l_i^F), vertical (l_i^V) and ceiling lines (l_i^C). A scene with n planes contains n ceiling and floor lines and $n-1$ vertical lines. *Blue dashed lines* denote the basic scene layout that originated the complete layout. *Black dashed lines* are the horizon and vertical lines defined by the vanishing points of the scene. (Best seen in color).

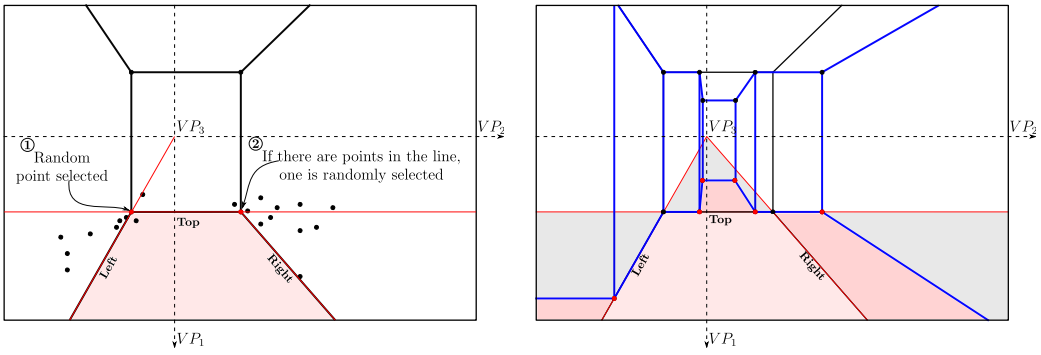
Building first frame room hypotheses

Fig. 6.5 shows the scene model that we adopt. Since we are using conventional cameras, with smaller field of view than omnidirectional cameras, the basic scene layout is formed by just three walls, Left, Top and Right. As the original work, we expand the basic layout in a hierarchical process.

1. **Building a basic room layout.** Room hypotheses are randomly generated from the intersections computed. The process is shown in Fig. 6.6. To build a basic room, a floor intersection is randomly chosen. The vanishing lines crossing in that point are computed. To finish the hypothesis, another floor intersection is chosen between those aligned with the computed vanishing lines. The lines crossing in those points compound the basic room hypothesis.

2. **Expanding a basic room layout.** Given a basic room hypothesis, it can be expanded to fit more complex environments. Fig. 6.6 shows this process. For each boundary of the basic room model, we look for intersections that could enlarge the floor area. The gray areas show where these intersections appear. The expansion process depends on the kind of boundary that we try to expand:

    - *Top boundary:* we start with a random floor intersection in the Top boundary. From this point, a vanishing line is computed and another point aligned with this new line is chosen. This process is repeated until we close the area.



Basic layout hypothesis Expanded layout hypothesis

Figure 6.6: Basic and expanded scene layout hypothesis. In both figures, red lines represent the vanishing lines used to build the hypotheses and red points are the points used to define one hypothesis. To build a basic layout hypothesis (black lines), a floor intersection point is chosen randomly and vanishing lines in the directions intersecting on that point are computed. To expand a basic layout hypothesis (blue lines), for each of the basic room boundaries (left, top and right) we look for intersections that enlarge the room area. Gray areas show the expansion area where these intersections appear. (Best seen in color).

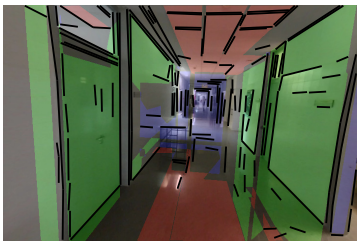
- *Left or Right boundaries:* in the case of Left or Right boundaries we define two ways of expanding the floor. We can choose a point aligned with the Top boundary as show for the Right boundary in Fig. 6.6, or choose a point in the correspondent boundary as done for the Left boundary in the same figure.

3. **Ceiling detection.** We adopt the Indoor World model that combines the Manhattan World assumption [Coughlan and Yuille, 1999] and a single-floor-single-ceiling model. This model applies to most indoor environments and introduces symmetry between ceiling and floor shapes, something useful when the floor-walls boundaries are occluded. Once the floor boundaries have been defined, we look for the ceiling boundaries. We assume floor-ceiling symmetry, so we just have to detect the height of the room. We compute the first vertical line of our model, and look for ceiling intersections aligned with that line: one is randomly chosen. When we have computed the height of one vertical line, the rest of the ceiling boundaries can be computed drawing parallel lines to the floor boundaries and computing the intersections with the rest of vertical lines.

Layout parametrization

A room hypothesis x_i is compound of n walls and can be parametrized as sets of floor, vertical and ceiling lines, and the vanishing points of the scene:

$$x_i = \{(l_1^F \dots l_n^F),\ (l_1^V \dots l_{(n-1)}^V),\ (l_1^C \dots l_n^C),\ (VP_1,\ VP_2,\ VP_3)\} \tag{6.7}$$



Observed orientation map

Hypothesis orientation map

Overlapping lines

Figure 6.7: Evaluation of the hypotheses. The observed orientation map, computed from the detected lines, and the hypothesis orientation map are compared to compute S_{omap}. $S_{overlap}$ is computed as the length of overlapping (red) divided by the total length of the hypothesis lines (black). Blue lines are the detected lines that are parallel and close to the model lines. (Best seen in color).

where l_i^F is the i-th floor line, l_i^V the i-th vertical line and l_i^C the i-th ceiling line, that is aligned with the same vanishing point than l_i^F (they are parallel in the scene). The model also includes the vanishing points: VP_1, VP_2 and VP_3.

Hypotheses evaluation

The evaluation of the hypotheses is performed on every frame. For all the images, lines and vanishing points are computed and used to evaluate the compatibility of the layout hypotheses. We define two measurements for this evaluation computed for each layout hypothesis x_i:

- *Orientation map:* as for the previous method presented, the orientation map is used as evaluation, S_{omap}.

- *Observed lines overlap:* this evaluation measures the length of the overlapping between the observed lines and the lines of the hypothesis being evaluated. The layout parametrization used defines model lines delimiting the layout areas (Fig. 6.7(c)). We look for lines parallel and close to these model lines and compute their overlapping length with the model lines. The score of this evaluation is computed as the total overlapping length divided by the total length of the model lines:

$$S_{overlap\ i} = \left(\frac{\sum\ overlap\ length}{\sum\ model\ lines\ length} \right)_i \tag{6.8}$$

Both scores are used together to evaluate the hypotheses:

$$S_{total\ i} = mean(S_{omap\ i},\ S_{overlap\ i}) \tag{6.9}$$

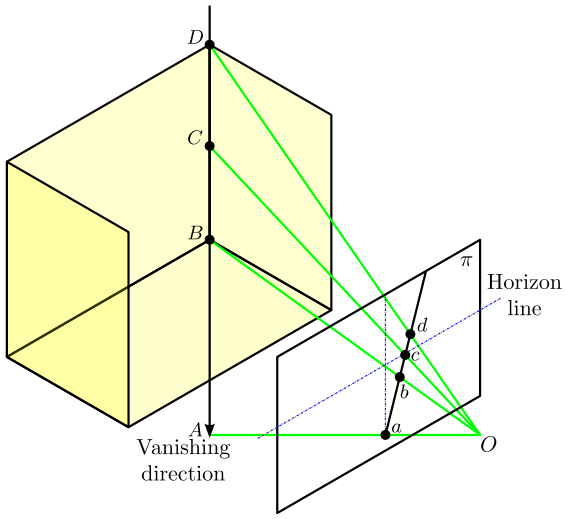


Figure 6.8: The cross-ratio of the four points showed remains constant between consecutive views. This relation is used to locate the ceiling points when sampling new hypotheses. (Best seen in color).

Sampling new hypotheses

A new set of hypotheses is created by sampling from the hypotheses of the previous frame and their evaluation score. For each hypothesis, a score has been computed, $S_{total\ i}$. The number of new hypothesis sampled from each previous hypothesis depends on this score. The probability of generating a new hypothesis, x_i', from previous hypothesis x_i is $p_i = S_{total\ i}$. New hypotheses are created randomly, high scores will generate more new hypotheses since they are more probable, and low scores hypotheses will receive few samples or even disappear.

The model used parametrizes the layout as sets of lines describing planes. Given the camera motion, a homography relates the projection of the coplanar points between frames and the vanishing points are related by the rotation matrix. We work with a moving camera where rotation and translation are unknown. To create a new hypothesis from a previous one, we assume a random motion of the camera, with zero velocity and random noise in camera translation and rotation. Random rotation and translation are created, R and $\mathbf{t}$, from 3 random angles ($\mathsf{R} = f(roll, pitch, yaw)$ and 3 random translations ($\mathbf{t} = [t_x, t_y, t_z]^T$). The homography H relating coplanar points can be computed as

$$\mathsf{H} = \mathsf{R} - \frac{\mathbf{t}\ \mathbf{n}^T}{d} \tag{6.10}$$

where $\mathbf{n}$ is the normal of the plane where the junction points lie and d the distance between the camera and the plane. The plane used is the floor plane, where we have computed the room hypothesis. We assume d distance as unitary so the scale of the random translation t is defined by the real distance to the plane.

From hypothesis x_i, sampled hypothesis x'_i will be related by the random R and $\mathbf{t}$. Points, pt, of the floor lines are related by a homography:

$$pt' = H \cdot pt = (\mathsf{R} - \frac{\mathbf{t}\ \mathbf{n}^T}{d})pt \quad (pt \in l_i^F | i = 1 \dots n) \tag{6.11}$$

and the vanishing points are related by the rotation matrix:

$$VP'_k = \mathsf{R} \cdot VP_k \quad (k = 1 \dots 3) \tag{6.12}$$

Ceiling points relation. Through the computed homography, we are able to relate the points on the floor, however we cannot relate the points in the ceiling of the scene, since they are part of a different plane. To relate the ceiling points, we assume that the distance between camera and floor remains the same between consecutive frames. We use the cross ratio to relate the height of the scene between images, Fig. 6.8. Given 4 collinear points, A, B, C and D, the cross ratio, C_R remains invariant for any perspective. The collinear points in our case are the two intersections defining the height of the room in the image, b and d, the vertical vanishing point, a, and the intersection between the vertical line and the line of the horizon, c. Since we consider the camera height to be the same between consecutive frames, the horizon is the same and the cross ratio remains constant. So, the cross ratio, C_R, is computed in the current image as:

$$C_R = \frac{|ac|}{|ad|}\frac{|bd|}{|bc|} \tag{6.13}$$

where $|ac|$ is the signed distance between a and c. Therefore, we obtain the ceiling point from the the floor point in next image as:

$$|b'd'| = C_R \frac{|a'd'||b'c'|}{|a'c'|} \tag{6.14}$$

6.4 Experiments

6.4.1 Analysis of the proposed method parameters

This section shows how the accuracy of our method varies with two important choices: a) the evaluation measurements used and b) the number of particles used. The results shown in this section correspond to the Entrance 1 sequence.

Experimental settings

We have tested our method on the 10 sequences included in the dataset presented in [Furlan et al., 2013]. These sequences have been acquired indoors with two different mobile cameras (Fig. 6.9) and include between 203 and 965 images. For all the sequences, the ground-truth has been manually annotated in one out of ten images.

Figure 6.9 shows example frames of all the dataset sequences. Not all of the environments captured in these sequences fit the layout model adopted in this work, some

Figure 6.9: First image of all the sequences included in the dataset [Furlan et al., 2013].

sequences consist of more than three walls (Corridor and Room 4) or the Manhattan World assumption cannot be applied (walls are not orthogonal in Room 1).

The accuracy of the solution is computed as the number of pixels where the orientation defined by the ground-truth and the orientation computed from the layout hypothesis is the same divided by the total number of pixels of the image

$$Accuracy = 100 \frac{\sum_{k=0}^{nPix} omap(GT)_k = omap(x_i)_k}{nPix} \tag{6.15}$$

where k is the pixel index, GT denotes the ground-truth layout, x_i is the layout hypothesis being analyzed and the number of pixels in the image is $nPix = width \times height$.

Influence of the evaluation measurement

Tables 6.1a and 6.1b show the mean accuracy of the solution hypothesis on all the frames of the sequence. It analyzes the contribution of the different evaluation measurements in the accuracy of our method. For the Baseline model, results show clearly that combining the different evaluation measurements we get to choose always a better solution. However, for the Complex model the combination of both measurements give worse results than using just the orientation map evaluation, so for the next sections just the orientation map evaluation will be used for this model.

Influence of the number of particles

Tables 6.1c and 6.1d show the accuracy of the tested methods depending on the number of hypotheses. The worst results correspond to the minimum number of hypotheses studied for both models, 25 hypotheses, from that point the resulting values improve. However, augmenting the number of hypotheses do not represent a big change in the method accuracy. This allows to reduce the computation time of the algorithm by keeping the number of hypothese low.

Table 6.1: Accuracy of the proposal depending on the method paramenters.

(a) Baseline model - Evaluation methods

	μ
S_{omap}	70.58
S_{lines}	75.47
S_{model}	59.38
$mean(S_{omap}, S_{lines})$	78.70
$mean(S_{omap}, S_{model})$	84.05
$mean(S_{lines}, S_{model})$	75.14
S_{total}	86.86

(b) Complex model - Evaluation methods

	μ
S_{omap}	79.50
$S_{overlap}$	53.40
S_{total}	75.88

(c) Baseline model - Number of Hypotheses

	μ
25	$73,24$
50	$86,80$
100	$81,02$
250	$85,09$
500	$83,63$
1000	$86,48$

(d) Complex model - Number of Hypotheses

	μ
25	$72,44$
50	$75,88$
100	$81,35$
250	$77,01$
500	$75,54$
1000	$78,47$

6.4.2 Analysis of the proposed framework with different single-image techniques

This section compares the performance of the proposed framework to propagate the scene layout with different models. It also compares this performance with the performance obtained running the single image methods studied, the Baseline and the Complex model, on all the images of a sequence. The experimental settings are the same that for the previous analysis. We analyze the performance propagating 50 and 100 hypotheses, all the evaluation measurements have been used for the Baseline model and just the orientation map evaluation for the Complex model. These results are shown in table 6.2.

We should note the performance differences depending on the model used. We see how the Complex model performs on average about a 10% better than the Baseline model. At the same time, the Complex Model represents better environments that do not fit the 3 walls scene modeled by the Baseline modelm such as Corridor or Room 4. However, when the environment fits that simple layout, the difference between both models is not so clear and the Baseline model performs good, even better solutions that the Complex model in some cases, e.g. see results for Entrance 1 and Room 1 in table 6.2.

We also analyze the differences between running a single-image algorithm in all the frames and using the propagation approach. We see how the best accuracies correspond to the propagation of 50 hypotheses for the Baseline model and 100 hypotheses for the Complex model, however, the differences are small. The great advantage of using the

Table 6.2: Accuracy of the proposed framework and different models.

	Baseline Model			Complex Model		
	Single Image	Propagation		Single Image	Propagation	
		50 Hyp.	100 Hyp.		50 Hyp.	100 Hyp.
Corridor	56.84	42.93	49.66	**81.31**	67.38	71.07
Entrance 1	80.13	**86.80**	81.02	81.50	79.50	75.53
Entrance 2	**74.27**	71.34	65.91	68.98	59.89	66.92
Lounge 1	47.40	**56.52**	26.42	56.51	50.42	52.58
Lounge 2	36.38	31.78	24.85	62.15	**68.02**	64.75
Room 1	50.73	**60.69**	57.30	59.84	54.01	55.30
Room 2	66.79	75.91	67.52	73.15	**84.46**	**75.83**
Room 3	36.82	63.42	60.60	59.62	63.35	**65.58**
Room 4	25.93	20.64	22.95	44.08	54.10	**68.36**
Room 5	64.70	69.27	85.44	81.99	83.90	**88.08**
Average	54.00	57.93	54.17	66.91	66.50	**68.40**

propagation framework is the reduction in the number of hypotheses, reducing the time needed to process each frame. When the approaches are run in a single image, around 850 hypotheses are generated. However, when the propagation framework is used, the number of hypotheses can be reduced to 100 or even 50, obtaining better or similar performances while reducing the running time, that is proportional to the number of hypotheses.

Finally, Figure 6.10 shows examples of the layout solution obtained for some frames of the dataset propagating the Complex Model. If we look at the layout for the Corridor sequence, we can see how this model can represent all the planes of the scene, obtaining a really good estimation. When lines are occluded, the method performs worse because the evaluation of the hypotheses is less accurate. For example, in sequences Lounge 1, Lounge 2 and Rooms 1 and 3 the layout fits the environment, but the method fails in adjusting hypotheses lines to the structure.

6.4.3 Improving object recognition tasks

This subsection shows results on object recognition tasks, poster detection in this case, using an egocentric vision dataset.

Experimental settings

The images used in this second experiment are part of the Wearable Computer Vision Systems dataset, further described in appendix A. It consists of several indoor sequences acquired with wearable vision sensors. We have selected certain frames along those sequences that contain poster or signs, our objects of interest, to be able to demonstrate how the context information provided by the layout helps to automatically discard wrong detections.

We analyze how the performance of a sign detector can be improved by using the

Figure 6.10: Examples of the resulting layout in some frames of all the dataset sequences. They show how the layout parametrization fits most of the environments of the dataset. Best seen in color.

layout information as prior information. We run the (rectangular) sign detector presented in [Cambra and Murillo, 2011]. We use the layout information to filter the rectangles and detect which ones are actual signs. We compute the Precision and Recall of the correctly detected signs given the rectangles provided by this detector, i.e., among those hypothesis given by the detector, which ones are correct or wrong after the filtering achieved thanks to the layout information.

Poster detection evaluation

Fig. 6.11 shows how the layout information improves the detection of posters in the images. We run the sign detector presented in [Cambra and Murillo, 2011] on a selection of images of the dataset. The detector creates detection hypotheses all over the image, but just some of them are correct. The rectangle hypothesis detected can be easily filtered using the scene layout and prior knowledge about man made environments to decide which hypothesis actually correspond to posters/signs or not:

- Scene objects are aligned with the scene vanishing points and with the vanishing points of the scene plane where they lay.
- Interesting objects, posters in our case, appear in walls, nor in floor or ceiling.
- Posters height is smaller than the wall height, they appear close to the eyes height (camera height in our case) and do not appear on top or bottom parts of the wall.

Sign detector detects 25 sign candidates per frame on average, and about 18 of these candidates are rejected (not aligned with the vanishing directions or are part of more than one layout region). The sign candidates remaining after the filtering are classified into poster or no poster. The precision of this classification is 95.24% and the recall is a bit lower 88.19%. The main reason for these high values is the filter step, where

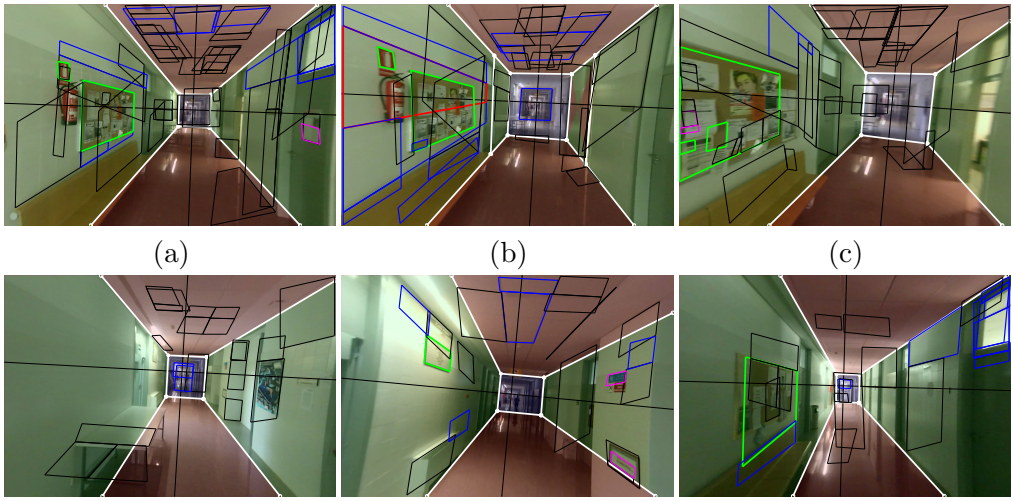


Figure 6.11: Black rectangles show rectangles that have been correctly discarded with our filtering: they are not aligned with the scene vanishing directions or are part of more than one layout region. Blue rectangles are aligned with the layout and the vanishing directions, but they are not classified as posters by our filtering because of the relative location in the scene. Green and red show detections accepted by our filtering (correct or incorrectly respectively) and magenta rectangles are signs that have been incorrectly rejected. (Best seen in color).

rectangles no fitting the structure are rejected. Fig. 6.11 shows examples of the poster detection.

6.5 Conclusions

In this section we have presented a new framework to obtain the 3D spatial layout of all the frames of a video sequence. This framework uses a single-image spatial layout technique to compute a set of hypotheses of the 3D scene structure in the first frame of the sequence. These hypotheses are then propagated to the rest of the images using a Particle Filter. We have used two different single image methods. First we have integrated the method presented in [Lee et al., 2009] in our framework. Then we have adapted the work presented in [López-Nicolás et al., 2014] for omnidirectional images to work with conventional images. This method can model more complex scenes than the work by Lee et al. We have tested our framework with both methods in real images and with different parameters. These experiments showed how by propagating the layout hypotheses we obtain lightly better results and a reduction of the number of hypotheses used, what reduces the computation time. Additionally, we have shown how a basic 3D spatial layout can improve the results in tasks such as object recognition, particularly for poster detection.

Chapter 7

Conclusions

This thesis has studied different vision based tools to create models and representations of the environment. Particular interest has been shown to the use of egocentric systems and omnidirectional cameras and interesting results have been achieved in the creation of environment representations at different semantic levels. The main contributions comprise from methods to create detailed 3D metric models of the environment using omnidirectional vision to techniques to build rough topological models with enhanced information about the kind of place where images were acquired. An important part of this work has been focused on augmenting the semantic information of the environment models to create applications where human-computer interaction is fundamental.

Creating models of the environment with computer vision

Intelligent systems require a proper representation or model of the environment to be able to understand his surroundings and perform complex tasks in it. This thesis has presented different techniques to create these representations.

In the case of metric models of the environment, we have focused on the use of omnidirectional cameras for SLAM. We have adapted a SLAM application to work with omnidirectional vision systems. The adaptation consist in the integration of the Spherical Camera model in a Extended Kalman Filter based visual SLAM. This adaptation allows to work with all kind of central vision systems (e.g. conventional cameras, catadioptric or dioptric systems) since all of them can be modeled with the Spherical Camera projection model. Experiments with real data have shown the superiority of omnidirectional systems over conventional cameras for this application. Trajectory and orientation estimations are better with the omnidirectional camera than with the conventional one, as well as the 3D reconstruction of the scene.

In the case of topological models, we have presented a new method to classify indoor omnidirectional images according to the type of area where they were acquired. We have designed our approach to classify images as *Places* or *Transitions*, further classified into different types of *Places* (Big, Medium or Small room, Corridors) and *Transitions* (Door, Jamb, Stairs, Elevator). The classifier is based on the Gist global descriptor, that we have adapted to work with omnidirectional images. The general idea to create enhanced topological models is to simultaneously run a topological map building approach and the *Place-Transition* classifier to label the different types of indoor scenes. We have included a Hidden Markov Model to add spatio-temporal consistency. A detailed semantic analysis of the types of transitions is not common although it provides important information for later uses of the map. As a result of this approach we get a topological map of the environment where the nodes correspond to *Places* and the links to *Transitions*.

We have also presented a new line-based global descriptor for omnidirectional images. The descriptor encloses the structure of the scene observed, by encoding the distribution of lines, and aims to categorize images according to this structure. We compute the boundaries in the image that correspond to lines of the environment and then we classify them according to the vanishing points. The descriptor is built as a histogram that captures how the different types of lines lay in the image space. Experiments for scene categorization show that the performance of our proposed descriptor is close to state-of-the-art global descriptors. Besides, our approach has shown interesting advantages: small size, reducing the memory consumption and comparison time, and higher rotation invariance are interesting properties for person-mounted and robotic cameras. We have tested our descriptor with two sets acquired with two different omnidirectional systems, catadioptric and panoramic images. This shows how our proposal can be easily adapted to any kind omnidirectional image.

We have addressed also the creation of visual BoW representations. We have presented a new method to build an enhanced visual vocabulary from an image sequence that learns semantic relationships between visual words of the working environment. The key elements of the method are the use of tracked scene points and the inclusion of geometric information of the key-points azimuth angles when building the visual words. Our approach is focused on long term robotics applications. With his purpose, we have considered systems where the camera points to the ceiling, which facilitates the acquisition of more stable and repetitive scene elements. Exhaustive experimental validation with indoor sequences, has shown the influence of the different method parameters in its performance and enhanced semantic properties. A comparison with the usual k-means vocabulary presented our method as a richer alternative for the usual BoW approach to build a visual vocabulary. Our method provides more representative semantic information of the environment, including relationships between visual words and environment objects or parts.

Finally, we have presented a framework to estimate the 3D spatial layout of all the frames of a video sequence. The framework is designed to work indoors and makes use of a single-image technique to compute the 3D spatial layout in the first frame and propagate this layout along the video sequence. The single-image technique is integrated with a Particle Filter to take advantage of the sequential information of video sequences. We have used two single-image techniques and showed how to integrate them in the proposed framework. First we have integrated the method proposed in [Lee et al., 2009]. To model more complex environments, we have adapted the method proposed in [López-Nicolás et al., 2014] for omnidirectional images and integrated it with our framework. Our experiments have analyzed the different parameters of the method and compared the performance of the two single-image methods. The integration of single-image methods with our propagation framework produces lightly better results than running the single-image techniques on each image of the sequence. Additionally, the use of the propagation framework allows to reduce the number of hypotheses computed on each frame, reducing the computation time. Our experiments have also demonstrates how the 3D layout we obtain provides useful priors for recognition tasks. In particular we have shown how poster detection can be improved by easily rejecting the numerous false positive detections.

Future work

A point discussed in this work are the benefits of integrating different models of the environment working together, however . Different models with different precision for the localization can complement each other weaknesses and improve the modeling. Also, the use of different models providing different semantic information about the environment can improve the performance of other tasks such as object recognition. Our objective for

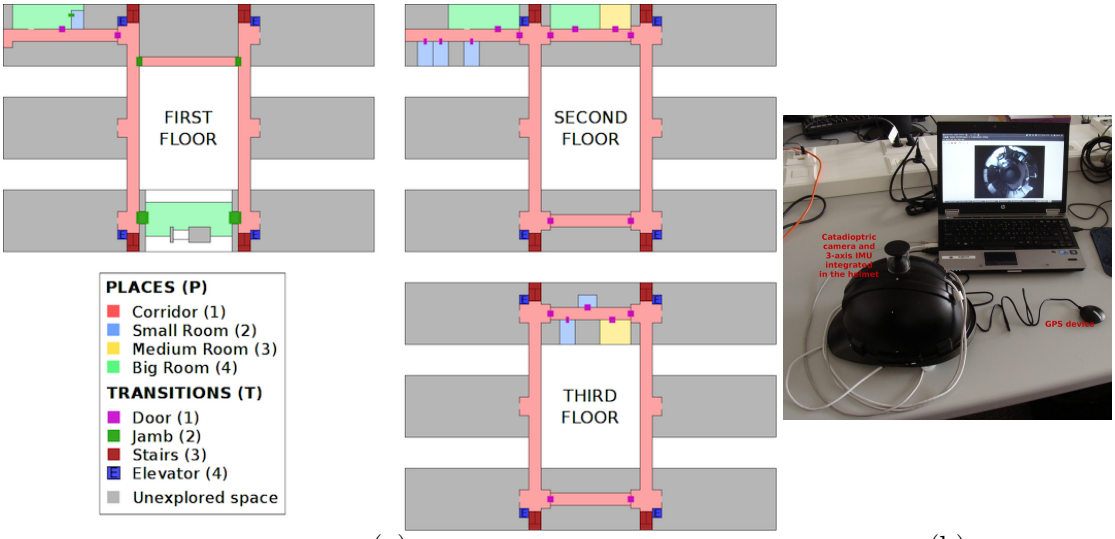
the close future is to develop a method integrating the different modeling levels studied: metric, topological and semantic, in order to create an base application to assist a person in the navigation on a new environment.

Great part of this thesis focuses on the use of wearable vision systems, usually with an omnidirectional camera. These systems are getting more and more attention lately with the appearance of affordable wearable cameras providing top video quality [GoPro, 2014, SONY, 2014, Contour, 2014]. More recently, are appearing new cameras that use omnidirectional vision [Giroptic, 2014, Geonaute, 2015, CENTRCamera, 2015]. As future work, I plan to keep researching in the use of wearable camera systems, both conventional and omnidirectional, to create assistance applications. Wearable cameras can be used as part of navigation assistance systems, to detect the position of the user and guide him to its destination or to the different elements of the environment. Additionally, these cameras have a huge potential as safety devices for cyclists or motorcyclist for example, to be aware of the surroundings all the time.

Appendix A

Public Datasets used

Different datasets and vision systems have been used along the different chapters of this thesis. Some of them are well known and widely used public datasets, such as the Rawseeds project, while others have been created for this work and released on-line for public use. All the sequences used have been acquired from an egocentric point of view, where the camera moves with the acquisition platform, and it points to the area of interest for the system. An important point of the work here described is the use of wearable sensor platforms, and most of the datasets used on this thesis have been created with these kind of platforms. Wearable systems are becoming very popular lately, and they are appropiate platforms to develop new human assistance applications.



(a) (b)

Figure A.1: *Wearable Omnidirectional System Dataset.* (a) Map of the building where the dataset has been acquired, different colors mean different type of area traversed. (b) Acquisition system: an omnidirectional catadioptric camera mounted on a helmet.

A.1 The Wearable Omnidirectional Vision System Dataset

The work in section 3 presents a method to create an enhanced topological model of an indoor environment. The method is designed for omnidirectional vision, particularly for a wearable omnidirectional system so it has been tested with such platform. The data used for this test is part of the Wearable OmniDirectional Vision System Dataset [1] acquired for this work and presented in [Rituerto et al., 2014c].

A.1.1 Environment

The dataset acquisition has been performed inside one building at our Campus at the University of Zaragoza, Spain. The building has three floors and includes areas of different types: corridors, research laboratories, offices, classes, etc. The acquisition has been performed by a person wearing the helmet, so the dataset suffers the typical motion of a person walking. A long trajectory covering as much areas as possible was performed (many areas are locked or with restricted access so it was not possible to cover all regions in the building). Figure A.1(a) shows the map of the three floors of the building highlighted with different colors, depending on the type of area traversed during the acquisition. The gray areas are parts not included in the dataset.

[1] `http://robots.unizar.es/omnicam/`

Table A.1: Camera calibration.

Im. size	1024x768
x_c	528.13
y_c	407.40
f_x	379.08
f_y	378.25
ξ	1.25

A.1.2 Acquisition platform

The dataset was acquired with our Wearable OmniCam acquisition system. This system includes a small hyper-catadioptric camera mounted on the top of a helmet (Fig. A.1(b)), a 3-axis IMU (compass, gyroscope and accelerometer) and a GPS device. The three sensors are synchronized and the camera has been calibrated using the approach described in [Puig et al., 2011]. However, the presented data has been acquired indoors, so GPS is deactivated. IMU data is also not used in this work, that is a purely vision based approach, but could be used for future works and included in the published data. Up to our our knowledge this is the first dataset published form indoor environments with a wearable omnidirectional camera.

Camera system

The visual data of the dataset has been acquired with a catadioptric system. It consists of 20905 omnidirectional images at 1024x768 pixels resolution acquired at a frame rate of 10 FPS. This catadioptric system has been made by vStone [REF] (Model: VS-C14U), it is compound by an USB camera and a small hyper-catadioptric mirror, it can be seen inFig. A.1(b).

Camera calibration

A.1.3 Ground truth

The ground truth labeling of the building areas has been made separating *Places* and *Transitions*. We consider the main spaces of a building, like corridors or rooms, as *Places*. *Transitions* label comprises all the areas joining different *Places*: doors, stairs, elevators, etc. The more detailed classification in type of *Places* or type of *Transitions* has been chosen to adequately describe the environment of acquisition. *Places* are classified as Big, Medium and Small Rooms and Corridors. Typically small rooms correspond to offices, medium to classes and big to halls or laboratories, for simplicity we classify them according to their size despite their different uses. *Transitions* are classified as Doors, Jambs, Stairs and Elevators. The areas labeled as *Transitions* starts about 0.5 meters before and ends about 0.5 meters after the *Transition* has been crossed.

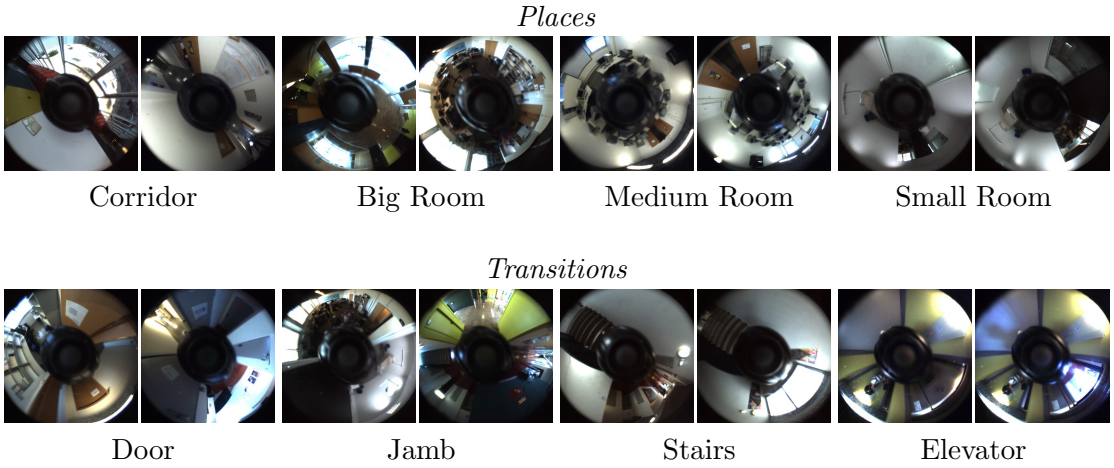
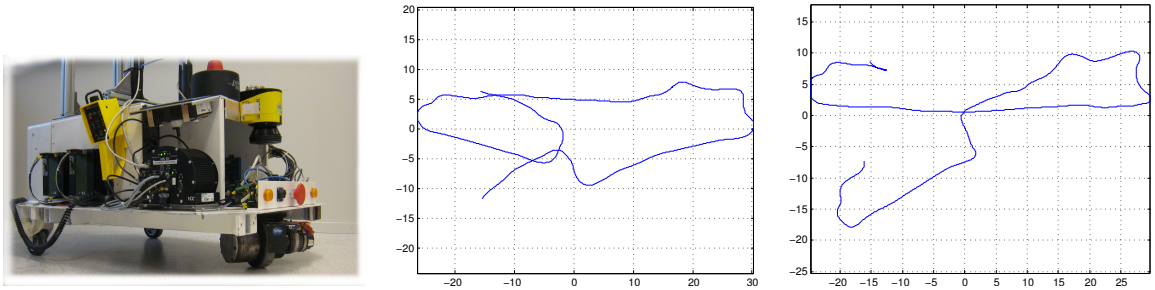


Figure A.2: Examples of images labeled in the ground truth as elements of the different classes and subclasses.

Table A.2: Number of clusters of each class in the dataset. Values between parentheses are the number of images of that class/subclass.

Places	TOTAL 56 (16522)	Corridor 38 (12577)	Big Room 7 (1559)	Medium Room 3 (1021)	Small Room 8 (1365)
Transitions	TOTAL 55 (4382)	Door 40 (1268)	Jamb 9 (514)	Stairs 4 (1933)	Elevator 2 (667)

All images have been manually labeled with the type of area where acquired and its position. Consecutive images labeled with the same type of area have been grouped into clusters. Table A.2 shows the number of clusters and the number of images, in parentheses, of each type.



(a) (b) (c)

Figure A.3: Robotic platform, (a), and trajectories followed during the acquisition of both sequences: (b) for sequence 1 and (c) for sequence 2.

A.2 Snowwhite robot Dataset

For the work in section 5 a new dataset is created. This work presents a method to enhance visual bag of words including information about important elements of the environment in the words. A particular point of the work is that it focus on long term operation. Pursuing this objective, the camera is set pointing to the ceiling so more stable regions and elements of the environment are captured. Since this configuration is not usual, we have acquired a new dataset that has been released for public use [2].

A.2.1 Environment

The dataset was acquired from a robotic platform at the AASS laboratories and offices in Örebro University, Sweden. The platform includes many sensors but we just used the wide angle camera. It includes two image sequences of two different trajectories around the same environment acquired at different days and following different trajectories. Both trajectories present certain overlapping in some parts of the sequence.

A.2.2 Camera system

The camera used is a uEye camera with a fish-eye lens. It has been set pointing to the ceiling to capture elements of the environment more stable over time with the objective of create applications for long term operation.

Camera calibration. To calibrate the camera and as projection system we used The Omnidirectional camera model presented in [Scaramuzza et al., 2006a].

A.2.3 Ground truth

The dataset includes the ground truth for the robot location and for important objects of the environment.

[2] `http://aass.oru.se/Research/Learning/datasets.html`

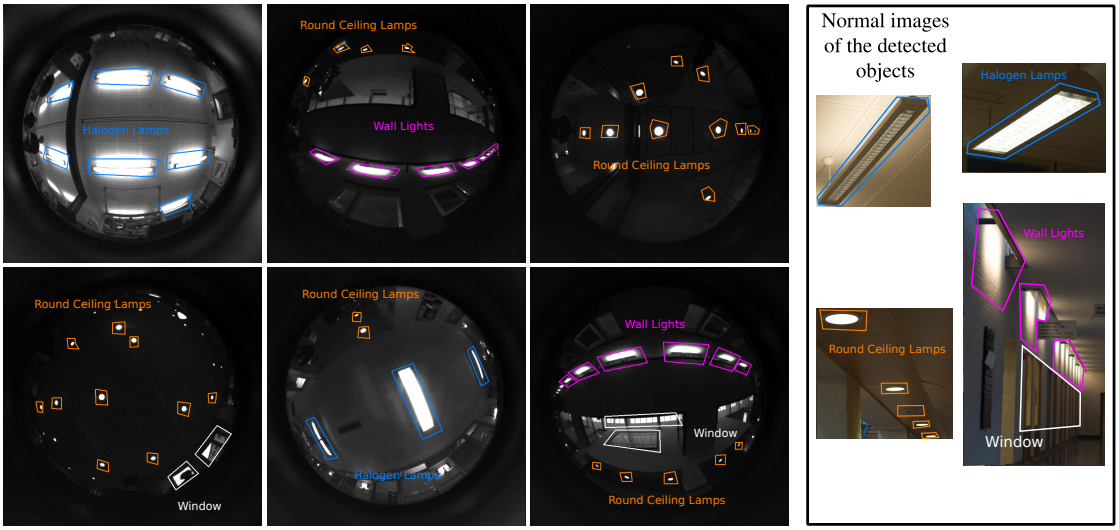


Figure A.4: Examples of objects found in the dataset.

Table A.3: Camera calibration.

Im. size	768x768
x_c	382.21
y_c	377.11
a_0	$-2.80 \exp 2$
a_1	0.00
a_2	$2.57 \exp -3$
a_3	$-9.66 \exp -6$
a_4	$2.89 \exp -8$

Localization ground truth

The trajectory ground truth corresponds to the odometry data of the robot (Fig. **??**(b) and (c)).

Object detection ground truth

Fig. A.4 shows some sample images of this dataset with the labeled objects and detailed images of the objects.

A.3 The Wearable Computer Vision Systems dataset

Section 6 makes use also of a different dataset: The Wearable Computer Vision Systems dataset [3]. This dataset is designed to evaluate different wearable computer vision systems, cameras that are worn by a person, mounted or attached to a human body, but not necessarily with the same point of view than the user. This dataset was released in the context of the ICCV 2013 Workshop on Wearable Computer Vision Systems [4]. During the acquisition of this dataset, several camera systems are worn simultaneously by 5 different users in 2 different scenarios. The goal is to acquire different perspectives of typical actions and activities performed indoors by an individual.

A.3.1 Environment

The acquisitions have been performed in two different buildings on Campus Río Ebro, in Zaragoza, Spain. And different users have wear the system. Users differ in height and sex, and the trajectories followed are also different

A.3.2 Acquisition platform

Each trajectory includes recordings from different vision sensors:

- an omnidirectional camera mounted on a helmet (helmet including IMU measurements);
- a RGB-d camera mounted on the head front;
- a mobile phone camera with;
- a wide angle lens mounted on the chest;
- a wide angle camera (GoPro Hero 2) mounted on the head front.

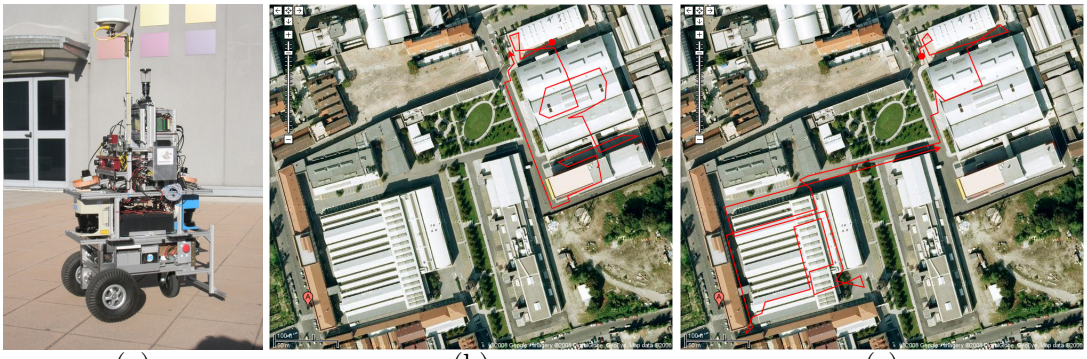
Camera system

For the las experiment performed in section 6, images from this dataset were used. The objective of this experiment is to show how the information of the scene structure can improve further tasks such as object recognition. Particularly, it demonstrate how poster detection can benefit from the inclusion of information about the structure of the environment traversed. Images from the head mounted camera are used (GoPro Hero 2) where different posters and signs appear in the images.

A.3.3 Ground truth

The ground truth of this dataset is to be released.

[3] https://i3a.unizar.es/es/content/wearable-computer-vision-systems-dataset
[4] https://sites.google.com/site/wwcv2013/home



(a) (b) (c)

Figure A.5: *Bovisa (outdoor + mixed)* dataset. (a) shows the Robocom platform. (b) and (c) show the aerial view of the environment and the trajectory followed by the robot split in two consecutive sections: first and second section respectively.

A.4 The Rawseeds Dataset

The Rawseeds Project [5] aims to build benchmarking tools for robotic systems. This is done through the publication of a comprehensive, high-quality Benchmarking Toolkit composed of: high-quality multisensor datasets, with associated ground truth; Benchmark Problems based on the datasets and Benchmark Solutions for the problems. The datasets are composed of raw sensor data, and they have been acquired acquired at different locations.

A.4.1 Environment

This thesis makes use of the data included in the *Bovisa (outdoor + mixed)* dataset acquired with the Robocom platform in in the Politecnico di Milano campus located in via Durando (Milan, Italy) (Fig.A.5). The explored environment include outdoor spaces among low buildings, narrow outdoor passages between buildings, a ramp connecting street level with a low terrace and the inside of a University buildings.

A.4.2 Acquisition platform

The sensors mounted on the Robocom platform are the following:

- Odometric system fitted to the Robocom robotic base.
- Binocular vision system, composed of a two-camera Videre Design STH-DCSG-VAR system.
- Trinocular vision system is realized combining the binocular STH-DCSG-VAR with an additional Videre Design DCSG camera.

[5] `http://www.rawseeds.org`

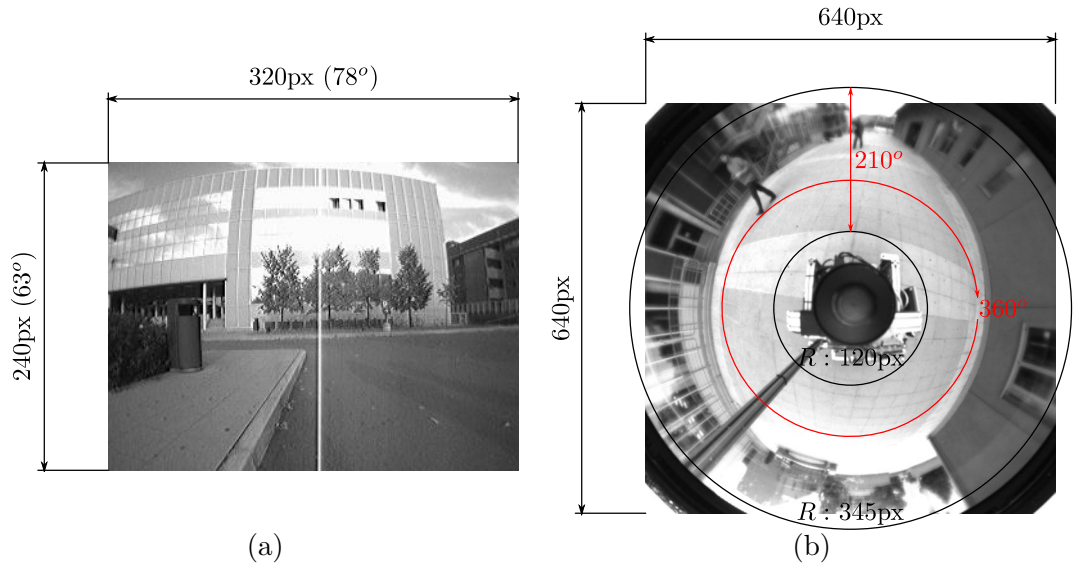


Figure A.6: Example images of the dataset used and the view angles of each system. (a) for the conventional images and (b) for the omnidirectional images.

- Color monocular vision is covered by an Unibrain Fire-i 400 camera.
- Omnidirectional color vision is obtained by using a Prosilica GC1020C color GigE Vision camera fitted with a hyperbolic mirror built by Vstone.
- Two Hokuyo URG-04LX laser range finders, mounted respectively on the front and the back of the robot.
- Sick LMS291 and LMS200 laser range finders, mounted on the front and the back of the robot.
- Xsense MTi inertial measurement unit.
- Trimble 5700 GPS receiver, with Zephyr GPS antenna.

We have chosen this dataset because it includes images from both systems, omnidirectional and conventional cameras. Chapter 2 analyzes the use of omnidirectional vision for SLAM and compares the results of using omnidirectional or conventional vision in for metric modeling. To perform this comparison, images acquired with both cameras simultaneously are needed. Besides, the dataset includes a ground truth trajectory to evaluate the performances of the SLAM trajectory results.

Camera systems

The data used for the work in chapter 2, images from the monocular conventional and omnidirectional cameras is used. Fig. **??** shows examples of images included in the dataset and corresponding to both systems.

Table A.4: Camera systems calibration

(a) Conventional camera

Im. size	320x240
x_c	171.91
y_c	127.41
f_x	195.72
f_y	195.72
k_c	-0.34
	0.15
	0.0004
	0
	-0.034

(b) Omnidirectional camera

Im. size	640x640	
	Provided	New
x_c	326.13	325.56
y_c	313.86	313.88
f_x	274.95	266.23
f_y	274.95	266.94
ξ	1	0.93

Angular resolution of the vision systems. Since we are using this data to perform a comparison between two different vision systems, we need them to be comparable. The angular resolution of a camera measures how many image pixels represent each degree captured. For the conventional camera, whose images size is 320x240 pixels, the view angles are 78^o in the horizontal and 63^o in the vertical direction. That gives angular resolutions of $4.10\frac{pixel}{o}$ and $3.80\frac{pixel}{o}$ in horizontal and vertical directions respectively, Fig. **??**(a). The omnidirectional camera captures the whole scene, 360^o, around the camera axis and 210^o in the radial direction of the image. The angular resolutions depend on where it is computed. It is $3.28\frac{pixel}{o}$ in the radial direction. In the tangential direction, the angular resolution goes from $2.09\frac{pixel}{o}$ for the inner radius to$6.02\frac{pixel}{o}$ for the outer radius, Fig. **??**(b). The average tangential angular resolution is $4.05\frac{pixel}{o}$.

The angular resolutions for both systems are similar, $3.80\frac{pixel}{o}$ and $4.10\frac{pixel}{o}$ for the conventional camera, and $3.28\frac{pixel}{o}$ and $4.05\frac{pixel}{o}$ for the omnidirectional camera, so the systems can be compared.

Camera calibration. The systems calibrations are included in the dataset. However, for the omnidirectional system we computed a new calibration. The reason is that the given calibration estimates $\xi = 1$ what corresponds to an para-catadioptric system, while the camera is hyper-catadioptric ($0 < \xi < 1$). The new calibration estimates $\xi = 0.93$.

A.4.3 Image sequences used

The whole sequence used [6] is compound by about 32200 omnidirectional images and 64400 conventional images. The frame rate for the omnidirectional images is 15 FPS while for the conventional camera ims set to 30 FPS. That causes the difference in the number of images of each type. For the experiments performed in section 2, the *Bovisa (outdoor + mixed)* dataset has been split in small challenging trajectories, Fig. A.7. All the selected

[6] `http://www.rawseeds.org/rs/capture_sessions/view/11`

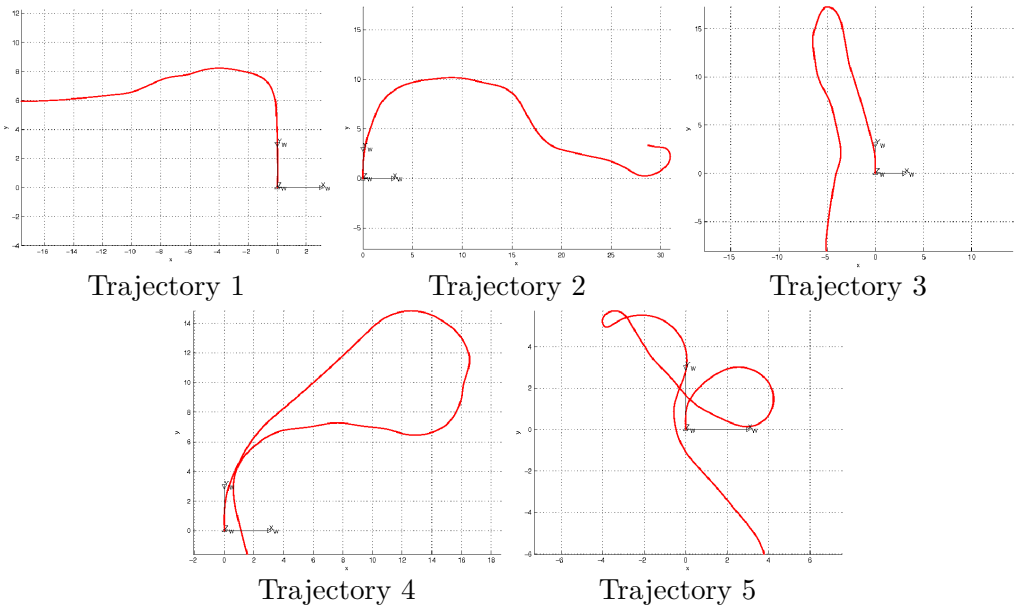


Figure A.7: Selected trajectories.

trajectories correspond to outdoor sections, the reason is the poor illumination of most of the indoor images.

- *Trajectory* 1*:* The simplest trajectory studied. The robot performs a 90^o turn.
- *Trajectory* 2*:* Robot perform different turns to both sides.
- *Trajectory* 3*:* The robot moves along a corridor and performs a 180^o turn to come back trough the same area.
- *Trajectory* 4*:* The robot performs a wide turn of and comes back to the same position.
- *Trajectory* 5*:* The most complex movement studied. The robot performs many turns, two of them of 270^o.

A.4.4 Ground truth

The dataset includes the ground truth for the robot trajectories. This ground truth is provided in two forms: GPS location and corrected odometry. The GPS location, despite being more precise than the odometry (maximum error of 2cm), is not available in all the sequence due to the outdoor indoor changes. Even when we are using just the outdoor data, recovering the GPS signal takes time and produces the lost of some position data. That is the reason to use the odometry data as ground truth for the experiments. Fig. A.7 shows the odometry trajectories of the selected sequences.

Figure A.8: First image of all the sequences included in the dataset.

A.5 Michigan Dataset

For section 6we have used the dataset presented in [Furlan et al., 2013]. In that work, authors introduce a new dataset [7][8] to evaluate the full capabilities of their approach. Their approach address the problem of estimating the 3D structural layout of complex and cluttered indoor scenes from monocular video sequences, where the observer can freely move in the surrounding space. So, contrary to previous similar datasets, the observer can move freely around to observe the scene.

A.5.1 Environment

The dataset includes 10 sequences in a variety of environments, spanning offices, corridors and large rooms. Most of the sequences frame ground-walls boundaries for short periods or do not frame them at all, making difficult thier detection. Other sequences present scenes that cannot be represented by a simple box layout model or relying on the Manhattan world assumption.

A.5.2 Camera systems

All the sequences were collected with common smartphones, to test the proposed method in real life scenarios with low-cost sensors. These smartphones correspond to a Samsung Galaxy S2 used for sequences Corridor, Entrance 1 and 2, Lounge 1 and 2 and Rooms 1 to 3, and a Nokia Lumia 920 used to acquire sequences Room 4 and 5. First image of all the sequences are shown in Fig. A.8.

Camera calibration

[7] http://vision.stanford.edu/3Dlayout/
[8] http://www.ira.disco.unimib.it/free_your_camera

Im. size	720x480
x_c	246.26
y_c	364.27
f_x	588.51
f_y	586.22

(a) Samsung Galaxy S2

Im. size	1280x720
x_c	424.14
y_c	706.87
f_x	1023.02
f_y	1040.83

(b) Nokia Lumia 920

A.5.3 Ground truth

Authors designed the dataset to evaluate their 3D structural layout method. For this purpose they annotated the first frame of all the sequences with the different surfaces that define the structure of the scene. To evaluate their approach, they processed all the sequence, but always evaluate comparing with this first frame projecting the solution to the first frame of the sequence. They can do this because they have run a Structure from Motion algorithm, so they have the camera translation and rotation of the camera between frames.

In our case, the work presented in section 6, estimates the 3D structure of the scene without computing the motion between cameras, so this kind of evaluation cannot be made. We annotate one of every ten frames of the rest of the sequence, to be able to evaluate our approach.

Bibliography

[Achtert et al., 2013] Achtert, E., Kriegel, H.-P., Schubert, E., and Zimek, A. (2013). Interactive data mining with 3D-parallel-coordinate-trees. In *ACM Conference on Special Interest Group on Management of Data (SIGMOD)*, pages 1009–1012.

[Angeli et al., 2008] Angeli, A., Filliat, D., Doncieux, S., and Meyer, J.-A. (2008). A fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions On Robotics, Special Issue on Visual SLAM*, 24(5):1027–1037.

[Ascending Technologies, 2013] Ascending Technologies (2013). Asctec falcon 8. `http://www.asctec.de/uav-applications/aerial-imaging/asctec-falcon-8/asctec-falcon-8/`.

[Autographer, 2013] Autographer (2013). Autographer. `http://www.autographer.com/#home`.

[Badrinarayanan et al., 2010] Badrinarayanan, V., Galasso, F., and Cipolla, R. (2010). Label propagation in video sequences. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3265–3272.

[Bao et al., 2011] Bao, S. Y., Sun, M., and Savarese, S. (2011). Toward coherent object detection and scene layout understanding. *Image and Vision Computing*, 29(9):569–579.

[Barreto and Araujo, 2001] Barreto, J. P. and Araujo, H. (2001). Issues on the geometry of central catadioptric image formation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 422–427.

[Bastanlar et al., 2008] Bastanlar, Y., Puig, L., Sturm, P., Guerrero, J. J., and Barreto, J. (2008). Dlt-like calibration of central catadoptric cameras. In *Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS)*.

[Bay et al., 2008] Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359.

[Bay et al., 2005] Bay, H., Ferrari, V., and Van Gool, L. (2005). Wide-baseline stereo matching with line segments. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Bazin et al., 2012] Bazin, J.-C., Demonceaux, C., Vasseur, P., and Kweon, I. (2012). Rotation estimation and vanishing point extraction by omnidirectional vision in urban environment. *The International Journal of Robotics Research*, 31(1):63–81.

[Belongie et al., 2002] Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522.

[Berg and Berg, 2009] Berg, T. L. and Berg, A. C. (2009). Finding iconic images. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1–8.

[Bermúdez-Cameo et al., 2012] Bermúdez-Cameo, J., Puig, L., and Guerrero, J. J. (2012). Hypercatadioptric line images for 3D orientation and image rectification. *Robotics and Autonomous Systems*, 60:755–768.

[Boiman et al., 2008] Boiman, O., Shechtman, E., and Irani, M. (2008). In defense of nearest-neighbor based image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

[Bolovinou et al., 2013] Bolovinou, A., Pratikakis, I., and Perantonis, S. (2013). Bag of spatio-visual words for context inference in scene classification. *Pattern Recognition*, 46(3):1039–1053.

[Burgard et al., 1999] Burgard, W., Cremers, A. B., Fox, D., Hähnel, D., Lakemeyer, G., Schulz, D., Steiner, W., and Thrun, S. (1999). Experiences with an interactive museum tour-guide robot. *Artificial intelligence*, 114(1):3–55.

[Cambra and Murillo, 2011] Cambra, A. B. and Murillo, A. (2011). Towards robust and efficient text sign reading from a mobile phone. In *International Conference on Computer Vision Workshops (ICCVW)*, pages 64–71.

[Cao et al., 2010] Cao, Y., Wang, C., Li, Z., Zhang, L., and Zhang, L. (2010). Spatial-bag-of-features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3352–3359.

[CENTRCamera, 2015] CENTRCamera (2015). CENTRCamera. `http://www.centrcam.com/`.

[Chahl and Srinivasan, 2000] Chahl, J. S. and Srinivasan, M. V. (2000). A complete panoramic vision system, incorporating imaging, ranging, and three dimensional navigation. In *Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS)*, pages 104–111.

[Chum et al., 2009] Chum, O., Perdoch, M., and Matas, J. (2009). Geometric min-hashing: Finding a (thick) needle in a haystack. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17–24.

[Civera et al., 2008] Civera, J., Davison, A. J., and Montiel, J. M. M. (2008). Inverse depth parametrization for monocular SLAM. *IEEE Transactions on Robotics*, 24(5):932–945.

[Contour, 2014] Contour (2014). ContourROAM3. `http://contour.com/`.

[Coughlan and Yuille, 1999] Coughlan, J. M. and Yuille, A. L. (1999). Manhattan world: Compass direction from a single image by bayesian inference. In *IEEE International Conference on Computer Vision (ICCV)*, pages 941–947.

[Cummins and Newman, 2009] Cummins, M. and Newman, P. (2009). Highly scalable appearance-only SLAM - fab-map 2.0. In *Robotics: Science and Systems (RSS)*.

[Cummins and Newman, 2011] Cummins, M. and Newman, P. (2011). Appearance-only SLAM at large scale with FAB-MAP 2.0. *The International Journal of Robotics Research*, 30(9):1100–1123.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Davison, 2003] Davison, A. J. (2003). Real-time simultaneous localisation and mapping with a single camera. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1403–1410.

[Davison et al., 2007] Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067.

[Delage et al., 2006] Delage, E., Lee, H., and Ng, A. Y. (2006). A dynamic bayesian network model for autonomous 3D reconstruction from a single indoor image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2418–2428.

[Deselaers et al., 2008] Deselaers, T., Keysers, D., and Ney, H. (2008). Features for image retrieval: an experimental comparison. *Information Retrieval*, 11(2):77–107.

[Doersch et al., 2012] Doersch, C., Singh, S., Gupta, A., Sivic, J., and Efros, A. A. (2012). What makes paris look like paris? *ACM Transactions on Graphics*, 31(4):101.

[Double Robotics, 2011] Double Robotics (2011). Double telepresence robot. `http://www.doublerobotics.com/`.

[Douillard et al., 2011] Douillard, B., Fox, D., Ramos, F., and Durrant-Whyte, H. (2011). Classification and semantic mapping of urban environments. *International Journal of Robotics Research*, 30(1):5–32.

[Eitz et al., 2010] Eitz, M., Hildebrand, K., Boubekeur, T., and Alexa, M. (2010). Sketch-based 3D shape retrieval. In *ACM SIGGRAPH*.

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 226–231.

[Everett et al., 1995] Everett, H. R., Gage, D. W., Gilbreath, G. A., Laird, R. T., and Smurlo, R. P. (1995). Real-world issues in warehouse navigation. In *Photonics for Industrial Applications*, pages 249–259.

[Fergus et al., 2003] Fergus, R., Perona, P., and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–264.

[Fernando et al., 2012] Fernando, B., Fromont, E., Muselet, D., and Sebban, M. (2012). Supervised learning of gaussian mixture models for visual vocabulary generation. *Pattern Recognition*, 45(2):897–907.

[Ferrari et al., 2008] Ferrari, V., Fevrier, L., Jurie, F., and Schmid, C. (2008). Groups of adjacent contour segments for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(1):36–51.

[Flint et al., 2011] Flint, A., Murray, D., and Reid, I. (2011). Manhattan scene understanding using monocular, stereo, and 3D features. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2228–2235.

[Friedman et al., 2007] Friedman, S., Pasula, H., and Fox, D. (2007). Voronoi random fields: extracting the topological structure of indoor environments via place labeling. In *International Joint Conference on Artifical intelligence (IJCAI)*, pages 2109–2114.

[Fukuda et al., 1996] Fukuda, T., Yokoyama, Y., Arai, F., Shimojima, K., Ito, S., Abe, Y., Tanaka, K., and Tanaka, Y. (1996). Navigation system based on ceiling landmark recognition for autonomous mobile robot -position/orientation control by landmark recognition with plus and minus primitives. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1720–1725.

[Furlan et al., 2013] Furlan, A., Miller, S., Sorrenti, D. G., Fei-Fei, L., and Savarese, S. (2013). Free your camera: 3D indoor scene understanding from arbitrary camera motion. In *British Machine Vision Conference (BMVC)*.

[Furukawa et al., 2009] Furukawa, Y., Curless, B., Seitz, S. M., and Szeliski, R. (2009). Reconstructing building interiors from images. In *IEEE International Conference on Computer Vision (ICCV)*, pages 80–87.

[Galindo et al., 2008] Galindo, C., Fernández-Madrigal, J., González, J., and Saffiotti, A. (2008). Robot task planning using semantic maps. *Robotics and Autonomous Systems*, 56(11):955–966.

[Garmin, 2014] Garmin (2014). vívofit. `http://sites.garmin.com/vivo/`.

[Gaspar et al., 2000] Gaspar, J., Winters, N., and Santos-Victor, J. (2000). Vision-based navigation and environmental representations with an omnidirectional camera. *IEEE Transactions on Robotics and Automation*, 16:890–898.

[Geonaute, 2015] Geonaute (2015). Geonaute 360. `http://www.geonaute.com/camera-geonaute-360.html`.

[Geyer and Daniilidis, 2000] Geyer, C. and Daniilidis, K. (2000). A unifying theory for central panoramic systems and practical applications. In *European Conference on Computer Vision (ECCV)*, pages 445–461.

[Giroptic, 2014] Giroptic (2014). 360Cam. `http://store.360.tv/en/`.

[Goedemé et al., 2007] Goedemé, T., Nuttin, M., Tuytelaars, T., and Van Gool, L. (2007). Omnidirectional vision based topological navigation. *International Journal of Computer Vision*, 74(3):219–236.

[Google, 2014a] Google (2014a). Google glass. `http://www.google.com/glass/start/`.

[Google, 2014b] Google (2014b). Project tango. `https://www.google.com/atap/projecttango/#project`.

[GoPro, 2014] GoPro (2014). GoPro HERO4. `http://es.gopro.com/`.

[Guerrero et al., 2008] Guerrero, J. J., Murillo, A. C., and Sagues, C. (2008). Localization and matching using the planar trifocal tensor with bearing-only data. *IEEE Transactions on Robotics*, 24(2):494–501.

[Gupta et al., 2010] Gupta, A., Efros, A. A., and Hebert, M. (2010). Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *European Conference on Computer Vision (ECCV)*, pages 482–496.

[Gutiérrez et al., 2011] Gutiérrez, D., Rituerto, A., Montiel, J., and Guerrero, J. (2011). Adapting a real-time monocular visual SLAM from conventional to omnidirectional cameras. In *International Conference on Computer Vision Workshops (ICCVW)*, pages 343–350.

[Hartley and Zisserman, 2003] Hartley, R. and Zisserman, A. (2003). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

[Hedau et al., 2009] Hedau, V., Hoiem, D., and Forsyth, D. (2009). Recovering the spatial layout of cluttered rooms. In *IEEE International Conference on Computer Vision (ICCV)*.

[Hedau et al., 2010] Hedau, V., Hoiem, D., and Forsyth, D. (2010). Thinking inside the box: Using appearance models and context based on room geometry. In *European Conference on Computer Vision (ECCV)*, pages 224–237.

[Hochdorfer and Schlegel, 2009] Hochdorfer, S. and Schlegel, C. (2009). Towards a robust visual SLAM approach: Addressing the challenge of life-long operation. In *International Conference on Advanced Robotics (ICAR)*, pages 1–6.

[Hoiem et al., 2005] Hoiem, D., Efros, A. A., and Hebert, M. (2005). Geometric context from a single image. In *IEEE International Conference onComputer Vision (ICCV)*, pages 654–661.

[Hoiem et al., 2007] Hoiem, D., Efros, A. A., and Hebert, M. (2007). Recovering surface layout from an image. *International Journal of Computer Vision*, 75(1):151–172.

[Hoiem et al., 2008] Hoiem, D., Efros, A. A., and Hebert, M. (2008). Putting objects in perspective. *International Journal of Computer Vision*, 80(1):3–15.

[Huang and Song, 2008] Huang, F.-S. and Song, K.-T. (2008). Vision SLAM using omnidirectional visual scan matching. In *International Conference on Intelligent Robots and Systems, 2008*, pages 1588–1593.

[Hwang and Song, 2011] Hwang, S.-Y. and Song, J.-B. (2011). Monocular vision-based SLAM in indoor environment using corner, lamp, and door features from upward-looking camera. *IEEE Transactions on Industrial Electronics*, 58(10):4804–4812.

[iRobot, 2002] iRobot (2002). irobot roomba vacuum cleaning robot. `http://www.irobot.com/us/learn/home/roomba.aspx`.

[Jégou et al., 2010] Jégou, H., Douze, M., and Schmid, C. (2010). Improving bag-of-features for large scale image search. *International Journal of Computer Vision*, 87(3):316–336.

[Jing-Yan Wang et al., 2013] Jing-Yan Wang, J., Bensmail, H., and Gao, X. (2013). Joint learning and weighting of visual vocabulary for bag-of-feature based tissue classification. *Pattern Recognition*, 46(12):3249–3255.

[Kang, 2000] Kang, S. B. (2000). Catadioptric self-calibration. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 201–207.

[Kim and Chung, 2003] Kim, J.-H. and Chung, M. J. (2003). SLAM with omnidirectional stereo vision sensor. In *International Conference on Intelligent Robots and Systems, 2003*, volume 1, pages 442–447.

[Konolige and Bowman, 2009] Konolige, K. and Bowman, J. (2009). Towards lifelong visual maps. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1156–1163.

[Kosecká and Zhang, 2002] Kosecká, J. and Zhang, W. (2002). Video compass. In *European Conference on Computer Vision (ECCV)*.

[Kovesi, 2000] Kovesi, P. D. (2000). MATLAB and Octave functions for computer vision and image processing. Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia.

[Kuipers, 2000] Kuipers, B. (2000). The Spatial Semantic Hierarchy. *Artificial Intelligence*, 119(1-2):191–233.

[Lee et al., 2010] Lee, D. C., Gupta, A., Hebert, M., and Kanade, T. (2010). Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *Advances in Neural Information Processing Systems (NIPS)*.

[Lee et al., 2009] Lee, D. C., Hebert, M., and Kanade, T. (2009). Geometric reasoning for single image structure recovery. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2136–2143.

[Lhuillier, 2007] Lhuillier, M. (2007). Toward flexible 3D modeling using a catadioptric camera. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

[López-Nicolás et al., 2014] López-Nicolás, G., Omedes, J., and Guerrero, J. (2014). Spatial layout recovery from a single omnidirectional image and its matching-free sequential propagation. *Robotics and Autonomous Systems*.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[Martínez Mozos and Burgard, 2006] Martínez Mozos, O. and Burgard, W. (2006). Supervised learning of topological maps using semantic information extracted from range data. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2772–2777.

[Microsoft Research, 2004] Microsoft Research (2004). Sensecam. `http://research.microsoft.com/en-us/um/cambridge/projects/sensecam/introduction.htm`.

[Mičušík and Pajdla, 2006] Mičušík, B. and Pajdla, T. (2006). Structure from motion with wide circular field of view cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7):1135–1149.

[Murillo et al., 2010] Murillo, A. C., Campos, P., Kosecka, J., and Guerrero, J. J. (2010). Gist vocabularies in omnidirectional images for appearance based mapping and localization. In *Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS)*.

[Murillo et al., 2012] Murillo, A. C., Gutiérrez-Gómez, D., Rituerto, A., Puig, L., and Guerrero, J. (2012). Wearable omnidirectional vision system for personal localization and guidance. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 8–14.

[Murillo and Kosecka, 2009] Murillo, A. C. and Kosecka, J. (2009). Experiments in place recognition using gist panoramas. In *Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS)*, pages 2196–2203.

[Murillo et al., 2007] Murillo, A. C., Sagüés, C., Guerrero, J. J., Goedemé, T., Tuytelaars, T., and Van Gool, L. (2007). From omnidirectional images to hierarchical localization. *Robotics and Autonomous Systems*, 55(5):372–382.

[Nagahara et al., 2006] Nagahara, H., Yagi, Y., and Yachida, M. (2006). Super wide field of view head mounted display using catadioptrical optics. *Presence*, 15(5):588–598.

[Nieto-Granda et al., 2010] Nieto-Granda, C., Rogers, J., Trevor, A., and Christensen, H. (2010). Semantic map partitioning in indoor environments using regional analysis. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1451–1456.

[Nieuwenhuisen et al., 2010] Nieuwenhuisen, M., Stückler, J., and Behnke, S. (2010). Improving indoor navigation of autonomous robots by an explicit representation of doors. In *International Conference on Robotics and Automation (ICRA)*, pages 4895–4901.

[Nüchter and Hertzberg, 2008] Nüchter, A. and Hertzberg, J. (2008). Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, 56(11):915–926.

[Oliva and Torralba, 2001] Oliva, A. and Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175.

[Oliva and Torralba, 2006] Oliva, A. and Torralba, A. (2006). Building the gist of a scene: The role of global image features in recognition. *Progress in brain research*, 155:23–36.

[Omedes et al., 2013] Omedes, J., López-Nicolás, G., and Guerrero, J. (2013). *Omnidirectional Vision for Indoor Spatial Layout Recovery*, volume Frontiers of Intelligent Autonomous Systems, pages 95–104. Springer.

[OrCam Technologies, 2010] OrCam Technologies (2010). Orcam. `http://www.orcam.com/`.

[pebble, 2012] pebble (2012). pebble. `https://getpebble.com/`.

[Penatti et al., 2014] Penatti, O., Silva, F. B., Valle, E., Gouet-Brunet, V., and Torres, R. d. S. (2014). Visual word spatial arrangement for image retrieval and classification. *Pattern Recognition*, 47(2):705–720.

[Philbin et al., 2007] Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

[Philbin et al., 2008] Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2008). Lost in quantization: Improving particular object retrieval in large scale image databases. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

[Pronobis et al., 2010a] Pronobis, A., Caputo, B., Jensfelt, P., and Christensen, H. I. (2010a). A realistic benchmark for visual indoor place recognition. *Robotics and Autonomous Systems*, 58(1):81–96.

[Pronobis et al., 2010b] Pronobis, A., Mozos, O. M., Caputo, B., and Jensfelt, P. (2010b). Multi-modal semantic place classification. *International Journal of Robotics Research, Special Issue on Robotic Vision*, 29(2–3):298–320.

[Puig et al., 2011] Puig, L., Bastanlar, Y., Sturm, P., Guerrero, J., and Barreto, J. (2011). Calibration of central catadioptric cameras using a dlt-like approach. *International Journal of Computer Vision*, 93(1):101–114.

[Puig et al., 2010] Puig, L., Guerrero, J. J., and Daniilidis, K. (2010). Topological map from only visual orientation information using omnidirectional cameras. In *IEEE International Conference on Robotics and Automation (ICRA), Workshop on Omnidirectional Robot Vision*.

[Pupilli and Calway, 2006] Pupilli, M. and Calway, A. (2006). Real-time visual SLAM with resilience to erratic motion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 1244–1249.

[Quattoni and Torralba, 2009] Quattoni, A. and Torralba, A. (2009). Recognizing indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 413–420.

[Ramisa et al., 2009] Ramisa, A., Tapus, A., Aldavert, D., Toledo, R., and Lopez De Mantaras, R. (2009). Robust vision-based robot localization using combinations of local feature region detectors. *Autonomous Robots*, 27(4):373–385.

[Ranganathan, 2010] Ranganathan, A. (2010). Pliss: Detecting and labeling places using online change-point detection. In *Robotics: Science and Systems (RSS)*.

[Raza et al., 2013] Raza, S. H., Grundmann, M., and Essa, I. (2013). Geometric context from video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Recon Instruments, 2014] Recon Instruments (2014). Recon jet. `http://reconinstruments.com/products/jet/`.

[Rituerto et al., 2014a] Rituerto, A., Andreasson, H., Murillo, A., Lilienthal, A., and Guerrero, J. (2014a). Building a hierarchical vocabulary from an image sequence. *Pattern Recognition (Under review)*.

[Rituerto et al., 2014b] Rituerto, A., Manduchi, R., Murillo, A. C., and Guerrero, J. (2014b). 3D spatial layout propagation in a video sequence. In *International Conference on Image Analysis and Recognition (ICIAR)*.

[Rituerto et al., 2014c] Rituerto, A., Murillo, A., and Guerrero, J. (2014c). Semantic labeling for indoor topological mapping using a wearable catadioptric system. *Robotics and Autonomous Systems, Special Issue Semantic Perception, Mapping and Exploration*, 62(5):685–695.

[Rituerto et al., 2013] Rituerto, A., Murillo, A. C., and Guerrero, J. (2013). Line image signature for scene understanding with a wearable vision system. In *International SenseCam & Pervasive Imaging Conference*, pages 16–23.

[Rituerto et al., 2014d] Rituerto, A., Murillo, A. C., and Guerrero, J. (2014d). 3D layout propagation to improve object recognition in egocentric videos. In *Assistive Computer Vision and Robotics (ACVR)*.

[Rituerto et al., 2014e] Rituerto, A., Murillo, A. C., and Guerrero, J. (2014e). Line-based global descriptor for omnidirectional vision. In *IEEE International Conference on Image Processing (ICIP)*.

[Rituerto et al., 2010a] Rituerto, A., Puig, L., and Guerrero, J. (2010a). Comparison of omnidirectional and conventional monocular systems for visual SLAM. In *Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS) (Best paper award)*.

[Rituerto et al., 2010b] Rituerto, A., Puig, L., and Guerrero, J. (2010b). Visual SLAM with an omnidirectional camera. In *International Conference on Pattern Recognition (ICPR)*, pages 348–351.

[Rituerto et al., 2011] Rituerto, J., Murillo, A., and Kosecka, J. (2011). Label propagation in videos indoors with an incremental non-parametric model update. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2383–2389.

[Rother, 2002] Rother, C. (2002). A new approach to vanishing point detection in architectural environments. *Image and Vision Computing*, 20(9):647–655.

[Rottmann et al., 2005] Rottmann, A., Mozos, O., Stachniss, C., and Burgard, W. (2005). Semantic place classification of indoor environments with mobile robots using boosting. In *National Conference on Artificial intelligence*, pages 1306–1311.

[Russell et al., 2009] Russell, B., Efros, A. A., Sivic, J., Freeman, B., and Zisserman, A. (2009). Segmenting scenes by matching image composites. In *Advances in Neural Information Processing Systems (NIPS)*.

[Russell et al., 2006] Russell, B. C., Freeman, W. T., Efros, A. A., Sivic, J., and Zisserman, A. (2006). Using multiple segmentations to discover objects and their extent in image collections. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1605–1614.

[Rybski et al., 2003] Rybski, P. E., Zacharias, F., Lett, J.-F., Masoud, O., Gini, M., and Papanikolopoulos, N. (2003). Using visual features to build topological maps of indoor environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 850–855.

[Sagüés et al., 2006] Sagüés, C., Murillo, A. C., Escudero, F., and Guerrero, J. (2006). From lines to epipoles through planes in two views. *Pattern Recognition*, 39(3):384–393.

[Santos-Victor et al., 1999] Santos-Victor, J., Vassallo, R., and Schneebeli, H. (1999). Topological maps for visual navigation. In *International Confference on Computer Vision Systems (ICCV)*, pages 21–36.

[Saurer et al., 2010] Saurer, O., Fraundorfer, F., and Pollefeys, M. (2010). Visual localization using global visual features and vanishing points. In *Conference on Multilingual and Multimodal Information Access Evaluation (CLEF)*.

[Saxena et al., 2009] Saxena, A., Sun, M., and Ng, A. Y. (2009). Make3D: Learning 3D scene structure from a single still image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):824–840.

[Scaramuzza et al., 2006a] Scaramuzza, D., Martinelli, A., and Siegwart, R. (2006a). A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *IEEE International Conference on Computer Vision Systems (ICCV)*, pages 45–45.

[Scaramuzza et al., 2006b] Scaramuzza, D., Martinelli, A., and Siegwart, R. (2006b). A toolbox for easily calibrating omnidirectional cameras. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5695–5701.

[Schwing et al., 2012] Schwing, A. G., Hazan, T., Pollefeys, M., and Urtasun, R. (2012). Efficient structured prediction with latent variables for general graphical models. In *International Conference on Machine Learning (ICML)*.

[Scotti et al., 2005] Scotti, G., Marcenaro, L., Coelho, C., Selvaggi, F., and Regazzoni, C. S. (2005). Dual camera intelligent sensor for high definition 360 degrees surveillance. *Vision, Image and Signal Processing*, 152(2):250–257.

[SDR Tactical Robots, 2000] SDR Tactical Robots (2000). Tactical surveillance robot. `http://www.sdrtactical.com/Surveillance-Robots/`.

[Singh et al., 2012] Singh, S., Gupta, A., and Efros, A. A. (2012). Unsupervised discovery of mid-level discriminative patches. In *European Conference on Computer Vision (ECCV)*, pages 73–86.

[Sivic and Zisserman, 2003] Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1470–1477.

[Sokal and Michener, 1958] Sokal, R. and Michener, C. (1958). A statistical method for evaluating systematic relationships. *University of Kansas Science Bulletin*, 6(38):1409–1438.

[SONY, 2014] SONY (2014). SONY ActionCam. `http://www.sony.net/Products/actioncam/en-us/?j-short=ActionCam`.

[Stachniss et al., 2005] Stachniss, C., Martinez-Mozos, O., Rottman, A., and Burgard, W. (2005). Semantic labeling of places. In *International Symposium of Robotics Research (ISRR)*.

[Svoboda and Pajdla, 2002] Svoboda, T. and Pajdla, T. (2002). Epipolar geometry for central catadioptric cameras. *International Journal of Computer Vision*, 49(1):23–37.

[Tapus and Siegwart, 2005] Tapus, A. and Siegwart, R. (2005). Incremental robot mapping with fingerprints of places. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2429–2434.

[Thrun and Bucken, 1996] Thrun, S. and Bucken, A. (1996). Integrating grid-based and topological maps for mobile robot navigation. In *National Conference on Artificial Intelligence*, pages 944–950.

[Thrun et al., 2005] Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. The MIT Press.

[Toepfer and Ehlgen, 2007] Toepfer, C. and Ehlgen, T. (2007). A unifying omnidirectional camera model and its applications. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1–5.

[Tomatis et al., 2003] Tomatis, N., Nourbakhsh, I., and Siegwart, R. (2003). Hybrid simultaneous localization and map building: a natural integration of topological and metric. *Robotics and Autonomous systems*, 44(1):3–14.

[Topp and Christensen, 2010] Topp, E. A. and Christensen, H. I. (2010). Detecting region transitions for human-augmented mapping. *IEEE Transactions on Robotics*, 26(4):715–720.

[Tsai and Kuipers, 2012] Tsai, G. and Kuipers, B. (2012). Dynamic visual understanding of the local environment for an indoor navigating robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4695–4701.

[Tuytelaars et al., 2010] Tuytelaars, T., Lampert, C. H., Blaschko, M. B., and Buntine, W. (2010). Unsupervised object discovery: A comparison. *International Journal of Computer Vision*, 88(2):284–302.

[Valgren and Lilienthal, 2010] Valgren, C. and Lilienthal, A. J. (2010). SIFT, SURF & seasons: Appearance-based long-term localization in outdoor environments. *Robotics and Autonomous Systems*, 58(2):149–156.

[Vasuvedan et al., 2007] Vasuvedan, S., Gachter, S., Nguyen, V., and Siegwart, R. (2007). Cognitive maps for mobile robots - an object based approach. *Robotics and Autonomous Systems*, 55(5):359–371.

[Vazquez-Reina et al., 2010] Vazquez-Reina, A., Avidan, S., Pfister, H., and Miller, E. (2010). Multiple hypothesis video segmentation from superpixel flows. In *European Conference on Computer Vision (ECCV)*, pages 268–281.

[Viswanathan et al., 2011] Viswanathan, P., Southey, T., Little, J., and Mackworth, A. (2011). Place classification using visual object categorization and global information. In *Canadian Conference on Computer and Robot Vision*, pages 1–7.

[Wang et al., 2009a] Wang, L., Neumann, U., and You, S. (2009a). Wide-baseline image matching using line signatures. In *IEEE International Conference on Computer Vision (ICCV)*.

[Wang et al., 2009b] Wang, Z., Wu, F., and Hu, Z. (2009b). Msld: A robust descriptor for line matching. *Pattern Recognition*, 42(5):941–953.

[Ward and Joe, 1963] Ward, J. and Joe, H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

[Wulf et al., 2006] Wulf, O., Lecking, D., and Wagner, B. (2006). Robust self-localization in industrial environments based on 3D ceiling structures. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1530–1534.

[Xiao et al., 2012] Xiao, J., Ehinger, K. A., Oliva, A., and Torralba, A. (2012). Recognizing scene viewpoint using panoramic place representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2695–2702.

[Xu et al., 2009] Xu, D., Han, L., Tan, M., and Li, Y. F. (2009). Ceiling-based visual positioning for an indoor mobile robot with monocular vision. *IEEE Transactions on Industrial Electronics*, 56(5):1617–1628.

[Yang et al., 2008] Yang, L., Jin, R., Sukthankar, R., and Jurie, F. (2008). Unifying discriminative visual codebook generation with classifier training for object category recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

[Yang and Newsam, 2011] Yang, Y. and Newsam, S. (2011). Spatial pyramid co-occurrence for image classification. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1465–1472.

[Zender et al., 2008] Zender, H., Mozos, O. M., Jensfelt, P., Kruijff, G.-J., and Burgard, W. (2008). Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems*, 56(6):493–502.