DISS. ETH NO. 17635

OMNIDIRECTIONAL VISION: FROM CALIBRATION TO ROBOT MOTION ESTIMATION

A dissertation submitted to

ETH ZURICH

for the degree of

Doctor of Science

presented by

DAVIDE SCARAMUZZA

M.S. Electronic Engineering, Università di Perugia, Italy

Date of birth: April 2, 1980

citizen of Terni, Italy

Accepted on the recommendation of

Prof. Roland Siegwart (ETH Zurich, thesis director)

Prof. Luc Van Gool (ETH Zurich, examiner)

Prof. Patrick Rives (INRIA Sophia Antipolis, examiner)

2008

Acknowledgements

I would like to express my thanks to Prof. Roland Siegwart, the head of the Autonomous System Laboratory (ASL), without whom I would never have visited and have become a member of the ASL. He deserves my deepest gratitude for adopting me into the group, for providing excellent research and social facilities. Furthermore, to give me the opportunity to to work in such an international group and learn two more languages. I feel very privileged to have had the opportunity to study and work in this excellent research group.

I am greatly indebted to my co-advisor and tutor during my first two years in Lausanne Dr. Agostino Martinelli (alias Veramente Massiccio). He has skillfully and "massicciamente" guided me throughout my research with his supportive attitude, vast scientific knowledge, capacity, and sympathy. His continuous support, patience, willingness, considerable help, and guidance have been of outmost importance for finishing my PhD study.

I am also greatly indebted to my friend and colleague Dr. Annalisa Milella for the numerous discussions and suggestions during her eight months at the ASL as well as the numerous discussions via email, chat, and telephone after her departure.

The work would have been impossible or at least much less enjoyable to complete without the support, assistance, and discussions with my colleagues, visitors of the ASL or former students, in particular: Jan Weingarten, Nicolas Criblez, Ahad Harati, Luciano Spinello, Stefan Gächter, Pierre Lamon, Kristijan Macek, Sascha Kolski, Francesco Mondada, Emanuele Frontoni.

I also gratefully acknowledge grants from the following European Projects where I had the pleasure and opportunity to work (in cronological order from 2008 back to 2005):

1. Robots@Home: An open Platform for Home Robotics,

url: $http://robsens.acin.tuwien.ac.at/robots_at_home/$

2. COGNIRON: The Cognitive Robot Companion,
   url: $http://www.cogniron.org/$
3. BACS: Bayesian Approach to Cognitive Systems,
   url: $http://www.bacs.ethz.ch/$
4. RECSYS: Real-Time Embedded Control of Mobile Systems with Distributed Sensing,
   url: $http://recsys.s3.kth.se/index.php?menu=home$

Finally, I gratefully acknowledge grants from the following international and internal ASL projects:

- SmartTer: A Vehicle for Autonomous Navigation and Mapping in Outdoor Environments,
  url: $http://www.smart-team.ch/$
- ELROB: First European Land Robot Trial,
  url: $http://www.m-elrob.eu/$

Abstract

For mobile robots to be able to work with and for people and thus operate in our everyday environments, they need to be able to acquire knowledge through perception. In other words they need to collect sensor measurements from which they extract meaningful information. This thesis covers some of the essential components of a robot perception system combining omnidirectional vision, odometry, and 3D laser range finders, from modeling to extrinsic calibration, from feature extraction to ego-motion estimation. We covers all these topics from the "point of view" of an omnidirectional camera. The contributions of this work are several and are listed here.

The thesis starts with an overview of the geometry of central omnidirectional cameras and gives also an overview of previous calibration methods. The contributions of this section are three. The first two are a new generalized model for describing both dioptric and catadioptric cameras and a calibration method which takes advantage of planar grids shown around the cameras, like the method in use for standard perspective cameras. The third contribution is the implementation of a toolbox for Matlab (called OCamCalib and freely available on-line), which implements the proposed calibration procedure.

The second part of the thesis is dedicated to the extraction and matching of vertical features from omnidirectional images. Vertical features are usually very predominant in indoor and outdoor structured environments and can then be very useful for robot navigation. The contribution of this part is a new method for matching vertical lines. The proposed method takes advantage of a descriptor that is very distinctive for each feature. Furthermore, this descriptor is invariant to rotation and slight changes of illumination.

The third part of the thesis is devoted to the extrinsic calibration of an omnidirectional camera with the odometry (i.e. wheel encoders) of a mobile robot. The contribution of this part is a new method of automatic self-

calibration while the robot is moving. The method is based on an extended Kalman filter that combines the encoder readings with the bearing angle observations of one ore more vertical features in the environment. Furthermore, an example of robot motion estimation is shown using the so calibrated camera-odometry system.

The fourth part of the thesis is dedicated to the extrinsic calibration of an omnidirectional camera with a 3D laser range finder. The contribution of this method is that it uses no calibration object. Conversely, calibration is performed using laser-camera correspondences of natural points that are manually selected by the user. The novelty of the method resides in a new technique to visualize the usually ambiguous 3D information of range finders. We show that is possible to transform the range information into a new image where natural features of the environment are highlighted. Therefore, finding laser-camera correspondences becomes as easy as image pairing.

The last part of the thesis is devoted to visual odometry for outdoor ground vehicles. We show a new method to recover the trajectory of a calibrated omnidirectional camera over several hundred of meters by combining a feature based with an appearance based approach.

All the contributions of this thesis are validated through experimental results using both simulated and real data.

Abstract (Italiano)

Affinché un robot mobile sia capace di lavorare con e per le persone ed operare nella vita di tutti i giorni, questo deve esser in grado di acquisire conoscenza per mezzo della percezione. In altre parole, deve acquisire misure dai sensori, da cui poi estrarre informazioni significative. Questa tesi copre alcuni dei componenti essenziali del sistema di percezione di un robot, che combina visione omnidirezionale, odometria e laser 3D: dalla modellazione alla calibrazione estrinseca, dall'estrazione di features alla stima del moto. Questi aspetti sono trattati dal "punto di vista" di una telecamera omnidirezionale. I principali contributi di questa tesi sono listati qui di seguito.

La tesi inizia con una revisione generale della geometria di telecamere centrali omnidirezionali e riassume precedenti lavori sulla calibrazione. I contributi di questa sezione sono tre. I primi due sono la concezione di un modello unificato per telecamere diottriche e catadiottriche ed una procedura di calibrazione che utilizza griglie planari mostrate dall'utente intorno alla telecamera, simile ai metodi standard per telecamere prospettiche. Il terzo contributo è l'implementazione di un toolbox per Matlab (detto OCamCalib e disponibile on-line), che implementa la procedura di calibrazione proposta.

La seconda parte di questa tesi è dedicata all'estrazione e all'accoppiamento di linee verticali in immagini omnidirezionali. Linee verticali sono predominatin in ambienti strutturati sia interni che esterni e possono essere molto utili per la navigazione di robot. Il contributo di questa parte è un nuovo metodo per l'accoppiamento di verticali. Tale metodo fa uso di un descrittore che è distintivo per ogni verticale. Inoltre tale descrittore è invariante alla rotazione e a lievi cambiamenti di illuminazione.

La terza parte della tesi è dedicata alla calibrazione estrinseca di una telecamera omnidirezionale con l'odometria di un robot mobile. Il contributo di questa parte è un nuovo metodo di auto-calibrazione durante il moto del

robot. Il metodo è basato sull'uso di un filtro di Kalman esteso che integra le letture odometriche con gli angoli di osservazione di uno o più verticali nell'ambiente. Inoltre si mostra anche un esempio di stima del moto del robot che utilizza il sistema camera-odometria cosi' calibrato.

La quarta parte della tesi è dedicata alla calibrazione estrinseca di una telecamera omnidirezionale con un laser 3D. Il contributo di questo metodo risiede nel fatto che non fa uso di oggetti di calibrazione. Al contrario, la calibrazione viene fatta utilizzando corrispondenze tra punti naturali estratti manualmente dall'utente dagli output di telecamera e laser. La novita' risiede in una tecnica per visualizzare l'ambigua informazione 3D dei laser. Si mostra che è possibile trasformare l'informazione 3D in una nuova immagine 2D in cui i dettagli dell'ambiente risultano enfatizzati. In tal modo la ricerca di corrispondenze diventa facile come quella tra immagini.

L'ultima parte della tesi è dedicata all'odometria visuale per veicoli da strada. Si presenta un nuovo metodo per ricostruire la traiettoria di una telecamera omnidirezionale già calibrata su alcune centinaia di metri combinando un approccio basato sull'estrazione di features con uno basato sull'apparenza delle immagini.

Tutti i contributi di questa tesi sono validati tramite risultati sperimentali su dati simulati e reali.

Contents

Chapter 1

Introduction

1.1 Preface

VISION is an extraordinarily powerful sense. The ability to perceive the environment allows for movement to be regulated by the world. Humans do this effortlessly but still lack the understanding of how perception works. In the case of visual perception, many researchers, from psychologists to engineers, are working on this complex problem. In this work, we want to understand how a robot can use images, which convey only 2D information, in a robust manner to drive its actions in 3D space.

A critical component of any perceptual system, human or artificial, is the sensing modality used to obtain information about the environment. In the biological world, for example, one striking observation is the diversity of "ocular" geometries. The majority of insects and arthropods benefit from a wide field of view and their eyes have a spacevariant resolution. To some extent, the perceptual capabilities of these animals can be explained by their specially adapted eye geometries. Similarly, in this work, we explore the advantages of having large fields of view by using an omnidirectional camera with a 360° azimuthal field of view.

Once images have been acquired by the omnidirectional camera, a question arises as what to do with them. Should they form an internal representation of the world? Furthermore, should they or should they not be combined with the information gathered by other sensors to help a robot to self-localize or autonomously navigate in unknown environments? How can they be used to model the environment? These fundamental questions have

long been addressed by the robotics community and go to the heart of our current research.

1.2 Sensors for robot perception

Navigation is the process used by a mobile robot to move from an initial position to a final position with respect to an initial reference frame. In order for a robot to navigate, it must first localize itself. In general, the term localization means the determination of the locality (position) of an object. In robotics, localization refers to methods through which a robot can calculate or update its position through information gathered from sensors. A robot must achieve localization in its operational environment in order for path planning and navigation algorithms to work effectively.

Mobile robots work with and for people and thus operate in our everyday environments. To do that, they need to be able to acquire knowledge through perception. In other words they need to collect sensor measurements from which they extract meaningful information.

Perception is achieved through sensors. A sensor is a device that measures or detects internal robot conditions (i.e. proprioceptive sensors) or external environmental conditions (exteroceptive sensors). Proprioceptive sensors are, for instance, wheel encoders, compass, inclinometers, accelerometers, and gyroscopes. Exteroceptive sensors are, for instance, cameras, laser range finders, and sonar. There are a wide variety of sensors used in mobile robots. In this work, we concentrate on omnidirectional cameras, odometry, and 3D laser range finders.

1.3 Motivation and objectives

In this thesis, we analyze the preliminary steps of robot perception using camera, odometry, and 3D laser range finder, from modeling to extrinsic calibration, from feature extraction to ego-motion estimation. We covers all these topics from the "point of view" of an omnidirectional camera but the primary goal remains describing the models of these sensors and how to relate them among each other (extrinsic calibration). In fact, while in computer vision many methods are known and are already standards for calibrating multiple cameras or camera arrays, in the robotics world, calibration between different sensors is often poorly documented. In this work, we want to review

these methods and provide novel and practical solutions. Here, we list and briefly describe the objectives of this thesis.

Extrinsic sensor calibration

To obtain accurate environment maps and also localize a mobile robot, every sensor must be precisely modeled and its relation (position and orientation) with the other sensors must be also correctly measured. The estimation of the extrinsic parameters among different sensors has to be very accurate for many applications. For example, in the frame-work of multi-robot localization, where the bearing observations (angle of sight) among the robots are very informative, an error of $1cm$ in estimating the position of the vision sensor with respect to the robot's encoder would produce a bearing angle error of 0.2 deg if the distance between the robots is $3m$. To this end, camera and robot's encoders must be extrinsically calibrated.

Another application is in the contest of automatic environment mapping. Digital 3D models of the environment are needed in autonomous navigation, rescue and inspection robotics, facility management, and architecture. Autonomous mobile robots equipped with 3D laser range finders are well suited for this task. However, to create realistic virtual models, visually-perceived information from the environment has to be acquired and it has to be precisely mapped onto the laser points. To accomplish this task, camera and 3D laser range finder must be also extrinsically calibrated.

Omnidirectional camera modeling and intrinsic calibration

To accomplish the tasks given above, we take advantage of an omnidirectional camera as its use has become largely diffused in the last decades in the robotics community. Omnidirectional cameras, by definition, are cameras that provide a 360° field of view of the scene. Such enhanced field of view can be obtained using multiple synchronized cameras, combinations of perspective cameras and mirrors, or just cameras with wide-angle lenses. In this work, we concentrate only on the last two mentioned types of omnidirectional cameras that are built by combining a standard camera with a shaped mirror or with a wide-angle lens; this cameras are respectively called catadioptric and dioptric cameras.

Catadioptric and dioptric cameras overcome the synchronization problems of panoramic multiple cameras and are also cheaper but, conversely, they have lower resolution. Furthermore, the presence of the mirror (or lens) introduces

distortions that must be accurately modeled in order to produce undistorted perspective views that can be further manipulated by the standard algorithms commonly used for perspective cameras. Thus, before facing the problem of extrinsic sensor calibration, we need to define a unified imaging model for omnidirectional cameras (both catadioptric and dioptric) and also a method to calibrate them.

Robust feature extraction and matching

The previous two objectives described the goal but not the methods to perform calibration. Our methods are based on visual feature extraction. Throughout this thesis, we use several types of features, like Harris, SIFT, and line features. We concentrate particularly on line features as they are more predominant in structure environments; we show that, by using an appropriate descriptor, these line features can me robustly and stably matched under different work conditions.

Ego-motion estimation

The last objective of this thesis is to show that also a single calibrated omnidirectional camera can be used alone, in place of other sensors, to perform robot motion estimation.

1.4 Original contributions

Before this work began, there was no simple way to model and calibrate omnidirectional sensors in both catadioptric and dioptric configurations. We devised a unified imaging model for omnidirectional cameras and a method to calibrate them. This led to four publications [1–4] and to an opensource toolbox for Matlab (called OCamCalib) available on the author's webpage [5].

An another important step is the calibration between camera and robot's encoders. We devised a method of self-calibration that allows to automatically and extrinsically calibrate the camera while the robot is moving. The method is called self-calibration because it requires no user interaction but only the capability of robustly and visually tracking line features among images. This work led to two publications [6, 7].

Another contribution is a robust method to extract and match vertical lines in omnidirectional images, that are used for calibrating the camera with

the encoders and also for motion estimation. The novelty of the method resides in the use of a line descriptor that is invariant to rotation and slight changes of illumination. This work led to a publication [8].

Another important step is the calibration between camera and 3D laser range finder. Although 3D scanners have diffused only in the last years, calibration consisted always in the use of a pattern viewed from both sensors and shown by the user at different positions and orientations. Unlike previous works, we devised a method of calibration from natural scenes, which uses no ad-hoc pattern but only points correspondences that are hand-selected by the user from a single laser-camera acquisition of a normal scene. This work led to a publication [9].

The final contribution regards ego-motion estimation for outdoor ground vehicles. We devised a method that uses point features and an appearance based approach to perform visual odometry. This work led to a publication [10] and at the moment when this PhD thesis is being published, we have been notified that it was also conditionally accepted for the IEEE Transactions of Robotics.

1.5 Outline of the thesis

The outline of this thesis is the following.

1.5.1 Chapter 2

We define the concept of central omnidirectional camera and review several imaging models that have been proposed in the last decades. There, we also introduce our unified imaging model, called Taylor model.

1.5.2 Chapter 3

We deal with omnidirectional camera intrinsic calibration, that is, we want to find the function that links 3D scene points to their 2D projections on the camera image plane. We describe our approach for calibrating our Taylor model, which leads to our opensource toolbox for Matlab.

1.5.3 Chapter 4

We describe how to extract and robustly match vertical lines in omnidirectional images. We define our descriptor and demonstrate that it is very distinctive.

1.5.4 Chapter 5

We deal with extrinsic calibration of an omnidirectional camera with robot's encoders. We define the problem and propose a solution that uses an extended Kalman filter. The method takes advantage of the line tracking technique of Chapter 4. In the end, we give also an example of robot motion estimation using odometry and visual features.

1.5.5 Chapter 6

We face the problem of extrinsic calibration of an omnidirectional camera with a 3D laser range finder. We define the problem and devise a practical method to establish the pose between the two sensors using a single camera-laser acquisition of a natural scene. In the end, we give different examples of mapping for several environments using also different omnidirectional cameras.

1.5.6 Chapter 7

We close the thesis showing an application of a calibrated omnidirectional camera to the problem of visual odometry. We describe our method and give an example of car trajectory recovery using, as only input, images provided by a single omnidirectional camera.

1.6 Structure of the chapters

The structure of each chapter is the following:

- Abstract: summary of the content of the chapter
- Introduction:
  - State of the art: previous works on the same topic
  - Outline of the chapter
- Theory: main content of the paper
- Results:
  - Simulations

  - Real experiments
- Conclusion: summary of the chapter, main contributions, and discussions

Chapter 2

Geometry of central omnidirectional cameras

In this chapter, we give a brief overview of the imaging models that have been proposed to describe central omnidirectional cameras, that is, cameras possessing a single effective viewpoint. We review these models for both catadioptric and dioptric configurations, the former using a combination of mirror and lenses, the latter just lenses with wide field of view (e.g. fisheye lenses). Finally, we propose our unified imaging model (called Taylor model) which encompasses both catadioptric and dioptric systems.

2.1 Introduction

2.1.1 The single effective viewpoint property

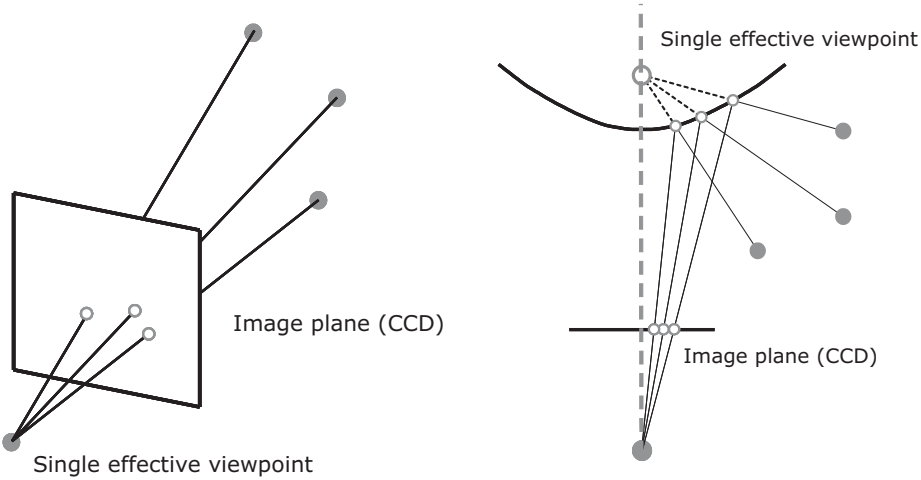
A vision system has a single effective viewpoint whenever it measures the intensity of light traveling along rays which intersect in a single point in 3D (the projection center). Vision systems satisfying the single viewpoint constraint are called central projection systems (Fig. 2.1).
The perspective camera is an example of a central projection system (Fig. 2.1(a)). In this case, the mapping in homogeneous coordinates of points in the scene into points in the image is linear and can be described by a 3×4 projection matrix $\mathbf{P}$ (pin-hole model) [11]. Perspective projection can be modeled by intersecting a plane with a pencil of lines going through the scene points and the projection center.

There are central projection systems whose geometry can not be described using the conventional pin-hole model because of the very high distortion introduced by the imaging device. These systems are central omnidirectional cameras and can appear as a combination of mirror and lenses (i.e. catadioptric, Fig. 2.3(a)) or just lenses (i.e. dioptric, Fig. 2.3(b)).
When an imaging system does not maintain a single viewpoint, then a caustic (i.e. a locus of viewpoints in three dimensions) is formed and the system has to be treated as a non-central one (Fig. 2.2).
In this thesis, we concentrate on central omnidirectional cameras, both dioptric and catadioptric. The reason a single effective viewpoint is so desirable is that it permits the generation of geometrically correct perspective images from the pictures captured by the omnidirectional camera. This is possible because, under the single view point constraint, every pixel in the sensed image measures the irradiance of the light passing through the viewpoint in one particular direction. When the geometry of the omnidirectional camera is known, that is, when the camera is calibrated, one can precompute this direction for each pixel. Therefore, the irradiance value measured by each pixel can be mapped onto a plane at any distance from the viewpoint to form a planar perspective image.
Another reason why the single view point property is important is that it allows applying the known theory of epipolar geometry [11], which easily permits to perform ego-motion estimation and structure from motion from image correspondences only.

2.1.2 State of the art of omnidirectional models

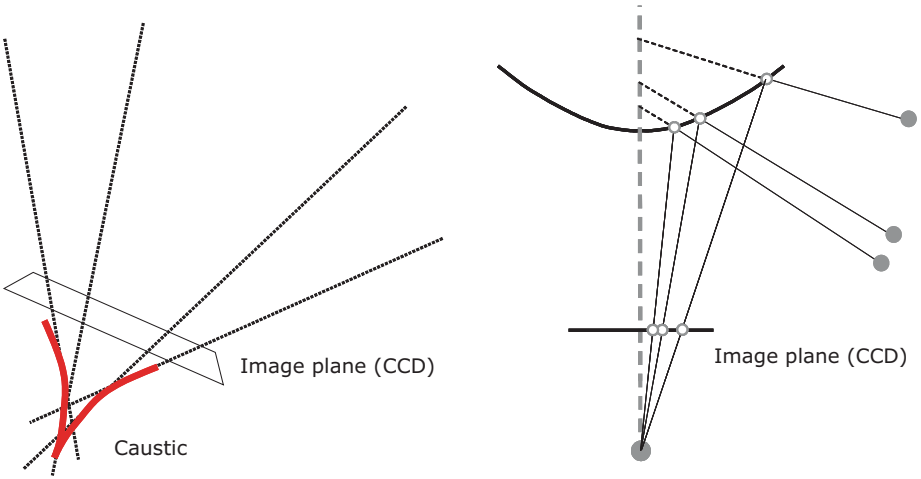
The concept of central catadioptric cameras appeared already in the presentation of René Descartes in 1637 in *Discours de la Methode* [12]. He showed that refractive as well as reflective *ovals* (conical lenses and mirrors) focus light into a single point if they are illuminated from another properly chosen point. The idea was later rephrased, e.g, by Feynman et al. in 1963 [13] or Hecht and Zajac in 1974 [14], and popularized into a modern language and introduced to the computer vision community in 1998 by Baker and Nayar [15–17].

Baker and Nayar derived the complete class of catadioptric sensors that have a single viewpoint and which can be constructed using just a single conventional lens and a single mirror under the assumption of a pinhole camera model. They showed that the 2-parameter family of mirrors that can be used



a b

Figure 2.1: Two examples of vision systems satisfying the single effective viewpoint property: (a) perspective pin-hole camera, (b) omnidirectional camera with hyperbolic mirror.



a b

Figure 2.2: Two examples of vision systems that do not possess a single effective viewpoint: (a) perspective camera, (b) catadioptric omnidirectional camera.

is exactly the class of rotated (swept) conic sections. Within this class of solutions, they identified four possible configurations of camera and mirror that are not degenerate and two configurations that are degenerate:

- The four non-degenerate configurations combine an orthographic camera with a paraboloidal mirror or a perspective camera with with a hyperboloidal, ellipsoidal, or planar mirror (Fig. 2.4).
- Conversely, the two degenerate configurations combine a perspective camera with a spherical or conical mirror (Fig. 2.5).

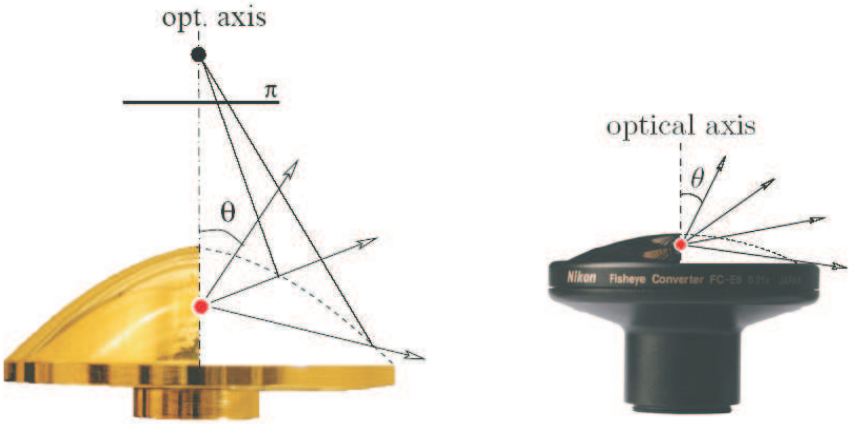
Observe that the degenerate configurations cannot be used to construct cameras with a single effective view point. Indeed, a sphere is the limit of an ellipse when the two focal points coincide; thus, the single viewpoint property would require to place the camera in the center of the sphere (i.e. inside) meaning that the camera would see only itself. Similarly, a conical mirror is the limit of a pinhole camera; thus, to have the single view point property, one should place the camera at the vertex of the cone, meaning that the camera would see nothing (Fig. 2.5).

A unifying theory for all central catadioptric systems was proposed in 2000 by Geyer and Daniilidis [18]. They showed and proved that every catadioptric (parabolic, hyperbolic, elliptical) and standard perspective projection is isomorphic to a projective mapping from a sphere (centered in the effective viewpoint) to a plane with the projection center on the perpendicular to the plane (Fig. 2.6).

In 2004, Ying and Hu [19] indicated that the unified imaging model by Geyer and Daniilidis can be used for some types of fisheye lenses. The approximation of a fisheye lens model by a catadioptric one is usually possible, however, with limited accuracy only.

Finally, in [20], Micusik proposed a new formalism for representing all central imaging models, both dioptric and catadioptric (including perspective models). By his formalism, he showed that it is possible to describe the relation between the image points and the corresponding 3D vectors to the real points by means of two functions h and g. Furthermore, he derived the two functions for the unified imaging model by Geyer and Daniilidis and several fisheye lens cameras.

In 1998, Svoboda et al. [21, 22] first introduced the concept of epipolar geometry to central catadioptric cameras. They showed that the epipolar constraint for central catadioptric cameras holds for 3D ray direction vectors corresponding to image points. They also proved that epipolar lines (known from perspective images) are replaced by conics since the epipolar planes are projected to the image plane as conics.



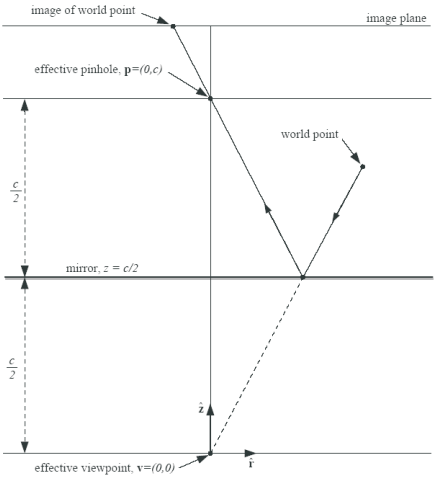
a b

Figure 2.3: (a) Catadioptric and (b) dioptric case (courtesy of Micusik [20]).

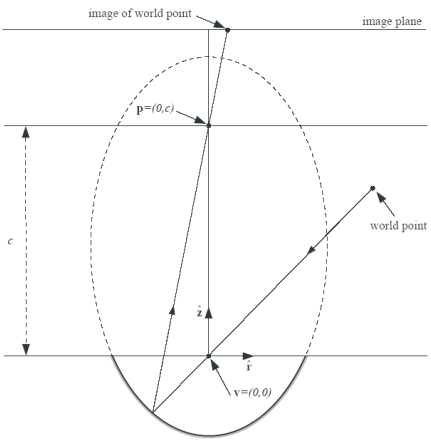
Relations that exist between multiple views of a static scene, where the views can be taken by any mixture of para-catadioptric, perspective or affine cameras, were described by Sturm [23]. The usage of this theory for motion estimation, 3D reconstruction or (self-)calibration was also indicated.

2.1.3 Outline of the chapter

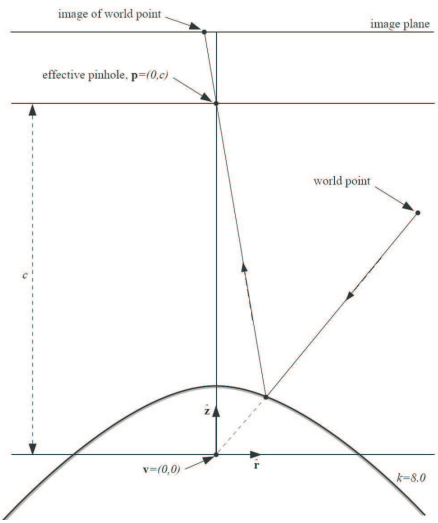
In this chapter, we use the same notation and formalism introduced by Micusik [20]. By using this formalism, we briefly review the unified imaging model by Geyer and Daniilidis for central catadioptric cameras and that proposed by Micusik himself for fisheye cameras (Section 2.2). Then, in Section 2.3 we present our new imaging model (called Taylor model) that is suitable for both dioptric and catadioptric central omnidirectional cameras. The novelty of our approach in comparison to previous works is that we describe the camera imaging model in terms of a Taylor polynomial expansion whose coefficients are the calibration parameters. The contributions of our work were first published in [1, 2] and further developed in [3, 4]. In Chapter 3, we will describe our calibration procedure for the proposed imaging model.



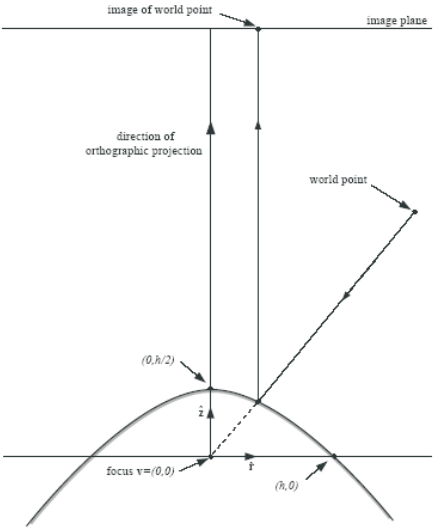
Planar mirror



Ellipsoidal mirror

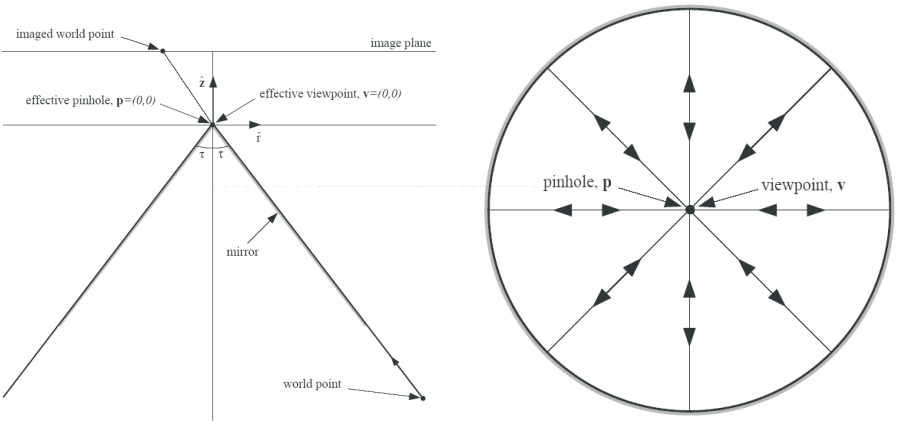


Hyperboloidal mirror



Paraboloidal mirror

Figure 2.4: The four non-degenerate catadioptric configurations derived by Baker and Nayar, which satisfy the single view point property (courtesy of Baker and Nayar [17]).



Conical mirror Spherical mirror

Figure 2.5: The two degenerate catadioptric configurations (courtesy of Baker and Nayar [17]).

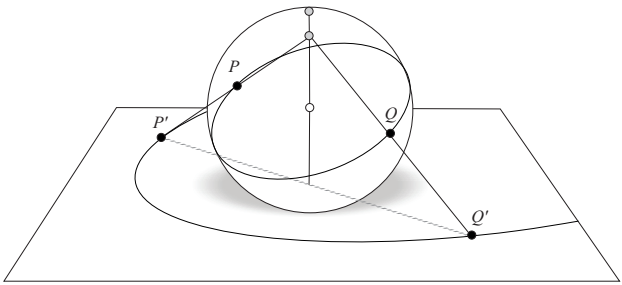


Figure 2.6: Stereographic projection used by Geyer and Daniilidis (courtesy of Geyer and Daniilidis [18]).

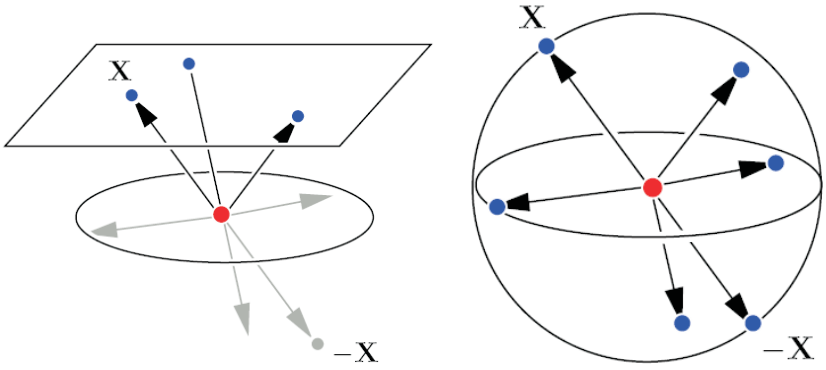


Figure 2.7: Central camera models. (a) The standard perspective model not distinguishing scene points lying on opposite half-lines. (b) The spherical model distinguishing two scene points lying on opposite half-lines (courtesy of Micusik [20]).

2.2 Omnidirectional camera models

The standard perspective camera model maps all scene points $\mathbf{X}$ from a line passing through the optical center of the camera to one image point $\mathbf{x}$ (Fig. 2.7(a)), so that [11]:

$$\lambda \mathbf{x} = \mathbf{P} \cdot \mathbf{X}, \tag{2.1}$$

$\mathbf{X} = [X, Y, Z]$, $\mathbf{x} = [x, y, 1]$ are the normalized image coordinates, and $\mathbf{P} \in \mathbb{R}^{3\times 4}$ is the projection matrix, that is, $\mathbf{P} = [\mathbf{R}|\mathbf{T}]$, where $\mathbf{R} \in SO(3)$ and $\mathbf{T} \in \mathbb{R}^3$ express the relation between the camera reference frame and the world reference frame.
In this representation, every image point is the projection of scene points that are in front as well as behind the camera. Thus, it allows to represent only scene points lying in a half-space including an image plane as a border. Conversely, real omnidirectional cameras with angle of view larger than 180° project points in front of the camera to one point and points behind the camera to a different point. Therefore, omnidirectional cameras have to be represented by half-lines, (Fig. 2.7(b)).

According to the Micusik formalism [20], image points of omnidirectional cameras can be represented in a spherical model as a set of unit vectors in $\mathbb{R}^3$

such that one vector corresponds just to one of half one-dimensional subspaces of $\mathbb{R}^3$. It means that one image point represents all scene points lying on a half-line emanating from the camera center in contrast to the perspective model, where one image point represents all scene points lying on whole line passing through the optical center. Thus, the projection equation for omnidirectional cameras can be written as:

$$\lambda \mathbf{q} = \mathbf{P} \cdot \mathbf{X}, \quad \lambda > 0, \tag{2.2}$$

where $\mathbf{q} = [x, y, z]$ is a unit vector (i.e. $\|\mathbf{q}\| = 1$) representing the image point.

When dealing with omnidirectional cameras, the following assumptions are always commonly taken:

1. The mirror (or fisheye lens for dioptric cameras) is rotationally symmetric with respect to its axis. This symmetry is guaranteed by manufacturing.
2. The mirror (lens) axis is perpendicular to the sensor plane.

In the following, we will assume central omnidirectional cameras. A fisheye lens will be used in the illustrative images; however, equations hold for both dioptric and catadioptric cameras.

Suppose we are observing a scene point $\mathbf{X}$ by an omnidirectional camera (Fig. 2.8, 2.9(a)). By using the spherical model given by Equation (2.2), there exists always a vector $\mathbf{p}'' = \left(\mathbf{x}''^T, z''\right)$ with the same direction as $\mathbf{q}$, which is mapped to the point $\mathbf{u}''$ on the sensor plane so that $\mathbf{u}''$ is collinear with $\mathbf{x}''$. This can be formalised as:

$$\mathbf{p}'' = \begin{bmatrix} h\left(\|\mathbf{u}''\|\right)\mathbf{u}'' \\ g\left(\|\mathbf{u}''\|\right) \end{bmatrix}, \tag{2.3}$$

where h and g are two functions $\mathbb{R} \to \mathbb{R}$, which depend on the distance $\|\mathbf{u}''\|$ of the point $\mathbf{u}''$ to the sensor axis origin (the center of symmetry). This dependency on $\mathbf{u}''$ is due to the assumption that the mirror (lens) is rotationally symmetric.

Functions h, g differ for various types of lenses and mirrors. For fisheye lenses, these functions depend on the type of the lens (e.g. equisolid, equiangular, etc.) and for mirrors they depend on the shape of the mirror (e.g. parabolic, hyperbolic, elliptical). The mapping of vector $\mathbf{p}''$ to the sensor plane point $\mathbf{u}''$ through functions h, g is shown in Fig. 2.9 for a fisheye lens

and a hyperbolic mirror.
In the case of fisheye lenses, it holds that $h = 1$ and thus vector $\mathbf{p}''$ is mapped orthographically to the point $\mathbf{u}''$ on the sensor plane.
In the case of mirrors, vector $\mathbf{p}''$ is mapped to $\mathbf{u}''$ through a perspective camera with optical center $\mathbf{C}$ (the optical center can lie at infinity for parabolic mirrors). We can also say that vector $\mathbf{p}''$ is projected orthographically to the point $h\left(\|\mathbf{u}''\|\right)\mathbf{u}''$ on the sensor plane.

At this point, the reader might be wondering why we need both functions h, g and not just one function g/h. As observed by Micusik [20], the reason is that g can be seen as the function describing the profile of the mirror, while h can be seen as the projection through the perspective camera (e.g. for the orthographic projection $h = 1$). For our Taylor model (Section 2.3), we will actually use g/h.

The mapping to the sensor plane can be specialized to obtain the standard perspective projection and omnidirectional projection as follows:

$$\begin{aligned} \text{Perspective projection:} \quad & \begin{bmatrix} 1\mathbf{u}'' \\ 1 \end{bmatrix} \\ \text{Omnidirectional projection:} \quad & \begin{bmatrix} h\left(\|\mathbf{u}''\|\right)\mathbf{u}'' \\ g\left(\|\mathbf{u}''\|\right) \end{bmatrix} \end{aligned} \tag{2.4}$$

In the remainder of this section, we give the expressions of h and g valid for the catadioptric and dioptric cameras.

2.2.1 Micusik's representation for catadioptric cameras

As we mentioned in Section 2.1.2, the unified model for catadioptric cameras by Geyer and Daniilidis [18] states that every catadioptric (i.e. parabolic, hyperbolic, elliptical) and standard perspective projection is isomorphic to a projective mapping from a sphere (centered in the effective viewpoint) to a plane with the projection center on the perpendicular to the plane (Fig. 2.6). According to the Micusik's formalism, every such mapping can be rewritten to the form of Equation (2.4) with

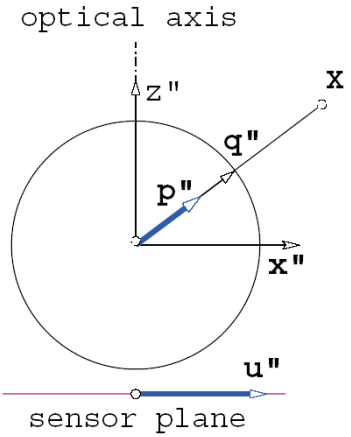


Figure 2.8: The mapping of a scene point $\mathbf{X}$ to the point $\mathbf{u}''$ on the sensor plane (courtesy of Micusik [20]).

$$h\left(\|\mathbf{u}''\|\right) = \frac{l(l+m)+\sqrt{\|\mathbf{u}''\|^2(1-l^2)+(l+m)^2}}{\|\mathbf{u}''\|^2+(l+m)^2},$$

$$g\left(\|\mathbf{u}''\|\right) = \frac{l\left\|\mathbf{u}''\right\|^2+(l+m)\sqrt{\|\mathbf{u}''\|^2(1-l^2)+(l+m)^2}}{\|\mathbf{u}''\|^2+(l+m)^2},$$

where constants l, m, depend on the type of catadioptric projection (i.e. parabolic, hyperbolic, elliptical). Their expressions can be found in [18].

2.2.2 Micusik's representation for dioptric cameras

Recently, a number of high quality, cheap, and widely available lenses with a field of view larger that 180° appeared: for instance the Nikon FC-E8 fisheye converter, COOLPIX digital camera, or the Sigma 8mm-f4-EX fisheye lens for cameras with 35mm film format. Fisheye lenses with so wide a field of view can be regarded as dioptric central omnidirectional cameras (Fig. 2.3).

As we already mentioned in Section 2.1.2, Ying and Hu [19] indicated that the unified imaging model by Geyer and Daniilidis can be used for some types of fisheye lenses. The approximation of a fisheye lens model by a catadioptric one is usually possible, however, with limited accuracy only. Depending on the desired accuracy, models with various number of

parameters can be derived. Here, we resume three models for fisheye lenses that have been derived by Micusik [20, 24], The simpler model is not very accurate, however it is precise enough to reject many bad outliers. The more complicated model is more accurate but is more computational complex and needs more points to be estimated.

Linear model

The linear model assumes that:

$$\theta = a_0 \left\| \mathbf{u}'' \right\| \tag{2.5}$$

This model holds approximately true for fisheye lenses with equiangular projection [25], where a_0 is a parameter and θ is the angle between a ray and the optical axis (Fig. 2.3). The relationship between the 3D vector $\mathbf{p}''$ emanating from the optical center $\mathbf{C}$ towards a scene point and the corresponding point $\mathbf{u}''$ on the sensor plane (Fig. 2.10) can be expressed according to Equation (2.3) in the lens Cartesian coordinate system as follows:

$$\mathbf{p}'' = \begin{bmatrix} \mathbf{u}'' \\ g\left(\left\|\mathbf{u}''\right\|\right) \end{bmatrix} = \begin{bmatrix} \mathbf{u}'' \\ \frac{\left\|\mathbf{u}''\right\|}{\tan\theta} \end{bmatrix} = \begin{bmatrix} \mathbf{u}'' \\ \frac{\left\|\mathbf{u}''\right\|}{\tan(a_0\left\|\mathbf{u}''\right\|)} \end{bmatrix}, \tag{2.6}$$

Non-linear models

Respectively for the Nikon and the Sigma fisheye lenses, Micusik used two more precise 2-parameter non-linear models:

$$\begin{aligned} \text{Nikon:} \quad \theta &= \frac{a_0\left\|\mathbf{u}''\right\|}{1+a_1\left\|\mathbf{u}''\right\|^2} \\ \text{Sigma:} \quad \theta &= \frac{1}{a_1}\arcsin\left(\frac{a_1\left\|\mathbf{u}''\right\|}{a_0}\right), \end{aligned} \tag{2.7}$$

where a_0, a_1 are different for the two lens models. In general, the models may have various forms determined by the lens design and by the desired accuracy. According to Equation (2.7), vector $\mathbf{p}''$ for the Nikon and the Sigma fisheye lenses becomes:

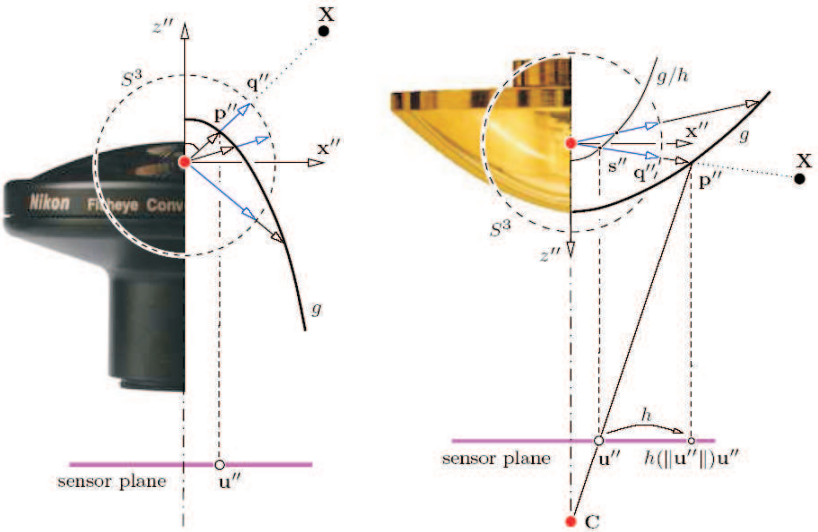


Figure 2.9: The mapping of a scene point $\mathbf{X}$ to the point $\mathbf{u}''$ on the sensor plane: (a) a fisheye lens and (b) a hyperbolic mirror (courtesy of Micusik [20]).

$$\begin{aligned}
\text{Nikon:} \quad \mathbf{p}'' &= \begin{bmatrix} \mathbf{u}'' \\ \frac{\|\mathbf{u}''\|}{\tan\theta} \end{bmatrix} = \begin{bmatrix} \mathbf{u}'' \\ \frac{\|\mathbf{u}''\|}{\tan\left(\frac{a_0\|\mathbf{u}''\|}{1+a_1\|\mathbf{u}''\|^2}\right)} \end{bmatrix} \\
\text{Sigma:} \quad \mathbf{p}'' &= \begin{bmatrix} \mathbf{u}'' \\ \frac{\|\mathbf{u}''\|}{\tan\theta} \end{bmatrix} = \begin{bmatrix} \mathbf{u}'' \\ \frac{\|\mathbf{u}''\|}{\tan\left(\frac{1}{a_1}\arcsin\left(\frac{a_1\|\mathbf{u}''\|}{a_0}\right)\right)} \end{bmatrix}
\end{aligned} \tag{2.8}$$

In [20], Micusik derived also an expression of a_0 as a function of a_1. In this way, he reduced the number of calibration parameters from two to one. He also proposed a calibration procedure to estimate the calibration parameters for each of these models.

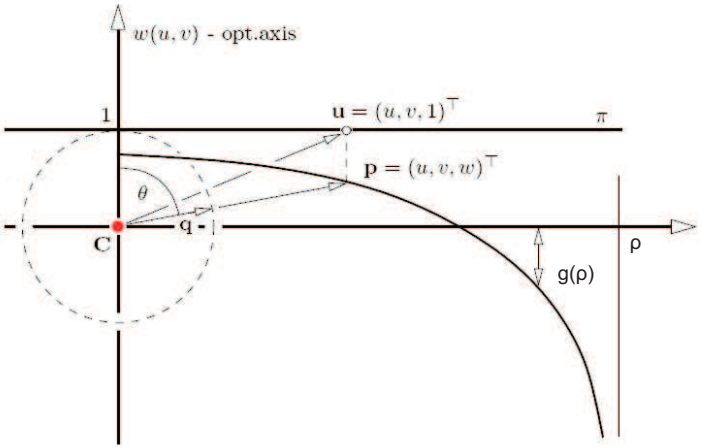


Figure 2.10: The geometrical interpretation of fisheye lens projection. Projection of the vector $\mathbf{q} \rightarrow \mathbf{p} \rightarrow \mathbf{u}$ onto the sensor plane π through function $g(\rho)$ (courtesy of Micusik [20]).

2.2.3 Projection onto the camera plane

In the general central camera model, we identify two distinct reference systems: the camera image plane and the sensor plane . The camera image plane coincides with the camera CCD, where the points are expressed in pixel coordinates. The sensor plane is a hypothetical plane orthogonal to the mirror (or fisheye lens) axis, with the origin located at the plane-axis intersection (i.e. camera optical center).

Until now, we described the projection of the scene onto the sensor plane. However, the camera image plane is never perfectly perpendicular to the sensor axis (i.e. a small misalignment is often present). Furthermore, the digitization process has also to be taken into account (i.e. pixels are non-square and are aligned in a linear but non-rectangular grid). In order to take also into account the orientation of the camera plane with respect to the sensor plane, we should add to our model a three degree-of-freedom rotation $\mathbf{R_c} \in SO(3)$. Concerning the digitization process, we should also introduce the so called intrinsic parameter matrix $\mathbf{K_c} \in \mathbb{R}^{\mathbf{3\times 3}}$. The composition of these two matrices is $\mathbf{H_c} = \mathbf{K_c}\mathbf{R_c}$ that is a Homography tranformation from the sensor plane to the camera plane. As observed in [20], when the misalignment between the camera and sensor plane is small, the Homography

transformation is well approximated by an Affine tranformation that transforms the circular field of view into an elliptical one in the digital image (Fig. 2.11(b), 2.11(c)). The Affine transformation is:

$$\mathbf{u}'' = \mathbf{A}\mathbf{u}' + \mathbf{t}, \tag{2.9}$$

where $\mathbf{u}'$, expressed in pixels, is a point in the camera image plane, $\mathbf{A} \in \mathbb{R}^{2\times 2}$, and $\mathbf{t} \in \mathbb{R}$.
In Fig. 2.11(b), 2.11(c), the two reference axes are shown for the case of a catadioptric system. In the dioptric case, the sign of $\mathbf{u}''$ would be reversed because of the absence of a reflective surface.
From now on, all coordinates will be expressed in the coordinate system placed in $\mathbf{O}$, with the z-axis aligned with the sensor axis (see Fig. 2.11(a)).

The complete image formation model capturing the projection of a scene point $\mathbf{X}$ to the digital image point $\mathbf{u}'$ can be divided into three parts:

1. A central projection of scene point $\mathbf{X}$ to vector $\mathbf{p}''$.
2. A non-perspective optics or mirror reflection described by functions h, g mapping $\mathbf{p}''$ to $\mathbf{u}''$.
3. A digitization process transforming point $\mathbf{u}''$ (on the sensor plane) to $\mathbf{u}'$ (on the digital image plane).

The complete image formation model can be written as:

$$\mathbf{p}'' = \begin{bmatrix} h\left(\|\mathbf{u}''\|\right)\mathbf{u}'' \\ g\left(\|\mathbf{u}''\|\right) \end{bmatrix} = \begin{bmatrix} h\left(\|\mathbf{A}\mathbf{u}'+\mathbf{t}\|\right)(\mathbf{A}\mathbf{u}'+\mathbf{t}) \\ g\left(\|\mathbf{A}\mathbf{u}'+\mathbf{t}\|\right) \end{bmatrix}. \tag{2.10}$$

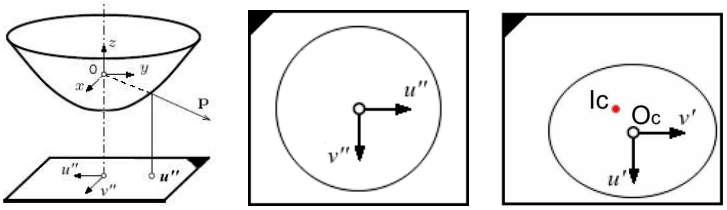
Thus, by using (2.10) and by combining it with Equations (2.2), (2.3), the complete projection equation for omnidirectional cameras is:

$$\lambda\mathbf{p}'' = \lambda \begin{bmatrix} h\left(\|\mathbf{A}\mathbf{u}'+\mathbf{t}\|\right)(\mathbf{A}\mathbf{u}'+\mathbf{t}) \\ g\left(\|\mathbf{A}\mathbf{u}'+\mathbf{t}\|\right) \end{bmatrix} = \mathbf{P}\cdot\mathbf{X}. \tag{2.11}$$

2.3 The Taylor model

In this section, we present our unified model for dioptric and catadioptric central omnidirectional cameras [1,2]. Instead of using two distinct functions h, g, we chose to use just one function g/h. This allows us to write $h = 1$ and thus we have to determine g that verifies the projection equation:

$$\lambda\mathbf{p}'' = \lambda \begin{bmatrix} \mathbf{u}'' \\ g\left(\|\mathbf{u}''\|\right) \end{bmatrix} = \mathbf{P}\cdot\mathbf{X} \tag{2.12}$$



a b c

Figure 2.11: (a) Coordinate system in the catadioptric case. (b) Sensor plane (metric coordinates). (c) Camera image plane (pixel coordinates). (b) and (c) are related by an Affine transformation.

Instead of using a specific expression for g depending on the sensor in use (as in Sections 2.2.1, 2.2.2), we chose a generalized parametric form which is suitable to different kinds of sensors. The reason of doing so is that we want this model to compensate for any misalignment between the focus point of the mirror (or the fisheye lens) and the camera optical center. Furthermore, we desire our generalized function to approximately hold with those sensors where the single viewpoint property is not exactly verified (e.g. generic fisheye cameras). In [1], we proposed the following polynomial form for g:

$$g(\|\mathbf{u}''\|) = a_0 + a_1\|\mathbf{u}''\| + a_2\|\mathbf{u}''\|^2 + ... + a_N\|\mathbf{u}''\|^N \tag{2.13}$$

where the coefficients $a_0, a_1, ..., a_N$ and the polynomial degree N are the calibration parameters; they will be determined in Chapter 3. As we observed in [2], this polynomial description of g can be further simplified by considering that all previous definitions of g (as in Sections 2.2.1, 2.2.2) always satisfy the following:

$$\left.\frac{dg}{d\rho}\right|_{\rho=0} = 0, \tag{2.14}$$

with $\rho = \|\mathbf{u}''\|$. This property holds for hyperbolic, parabolic, elliptical mirrors as well as for fisheye cameras (see [20, 22, 25]). This simplification allows us to impose $a_1 = 0$ and thus (2.13) can be rewritten as:

$$g(\|\mathbf{u}''\|) = a_0 + a_2\|\mathbf{u}''\|^2 + ... + a_N\|\mathbf{u}''\|^N \tag{2.15}$$

As a consequence, the number of calibration parameters reduced from $N+1$ to N.

Using (2.15), we can rewrite the image formation model (2.10) as:

$$\mathbf{p}'' = \begin{bmatrix} h\left(\|\mathbf{u}''\|\right)\mathbf{u}'' \\ g\left(\|\mathbf{u}''\|\right) \end{bmatrix} = \begin{bmatrix} \mathbf{u}'' \\ a_0 + a_2\|\mathbf{u}''\|^2 + ... + a_N\|\mathbf{u}''\|^N \end{bmatrix} \quad \text{with } \|\mathbf{u}''\| = \mathbf{A}\mathbf{u}' + \mathbf{t} \tag{2.16}$$

Thus, using (2.12) our projection equation for central omnidirectional cameras can be read as:

$$\lambda\mathbf{p}'' = \lambda \begin{bmatrix} \mathbf{u}'' \\ a_0 + a_2\|\mathbf{u}''\|^2 + ... + a_N\|\mathbf{u}''\|^N \end{bmatrix} = \mathbf{P}\cdot\mathbf{X} \quad \text{with } \|\mathbf{u}''\| = \mathbf{A}\mathbf{u}' + \mathbf{t} \tag{2.17}$$

In the remainder of this thesis, we will always use the Taylor model. Our calibration procedure to determine the parameters of this model will be described in Chapter 3.

2.4 Conclusion

In this chapter, we explained the concept of central cameras, that is, camera possessing a single effective viewpoint. As we mentioned in Section 2.1.1, the reason the single viewpoint property is so desirable is that it permits the generation of geometrically correct perspective images from the highly distorted omnidirectional images. Furthermore, it allows to extend to omnidirectional cameras the already known epipolar geometry that is valid for standard perspective cameras.

Central omnidirectional cameras exist in dioptric and catadioptric form. The former uses lenses with a field of view larger than 180° (i.e. fisheye lenses), the latter uses a combination of mirrors and lenses. As proved by Baker and Nayar in [16], the class of catadioptric sensors satisfying the single view point property can be built by combining an orthographic camera with a paraboloidal mirror or a perspective camera with with a hyperboloidal, ellipsoidal, or planar mirror. Concerning dioptric cameras, Micusik et all. [20,24], showed that in the recent years fisheye lenses have been built to satisfy the single viewpoint property (e.g. Nikon, Sigma lenses).

In Section 2.2, we used the Micusik's formalism to review the unified imaging model by Geyer and Daniilidis [18] for omnidirectional catadioptric

cameras (e.g. paraboloidal, hyperboloidal, ellipsoidal mirror) and the Micusik's models for some dioptric cameras (e.g. Nikon, Sigma lenses). Finally in Section 2.3, we described our new unified imaging model that uses a simplified Taylor polynomial expansion.

Due to its generality and despite its simplicity, we will see in the next chapters that the Taylor model has great properties such as:

- it is able to fit very well both catadioptric and dioptric cameras;
- especially for dioptric cameras, where an exact analytic model cannot be inferred, the Taylor model provides an approximation whose performance is comparable with those proposed in previous works and can then be used in lieu of them;
- because of the linearity of the model with respect to its coefficients, it leads to a closed form solution for the calibration parameters that can be solved using linear minimization followed by non-linear refinement;
- thanks to the efficiency of computing the calibration parameters, calibration methods can be devised that do not require the visibility of the circular external boundary of the mirror or lens (see 3.2.4 or [26]).

In the remainder of this thesis, we will always use our Taylor model. The calibration procedure to determine the parameters of this model will be described in Chapter 3.

Chapter 3

Omnidirectional camera calibration

In this chapter, we present a novel and flexible technique for calibrating central omnidirectional cameras. This technique takes advantage of the Taylor model introduced in the previous chapter. The proposed method only requires the camera to observe a planar pattern shown at a few different positions and orientations. Either the camera or the pattern can be freely moved. No a priori knowledge of the motion is required, nor a prior initialization of the camera model. A closed form solution is given for the parameters of the camera model. The performance of the approach is evaluated through experiments on both simulated and real data. Furthermore, the implementation of a toolbox for Matlab, which implements the proposed calibration procedure, is described. Compared with classical techniques, which rely on a specific model of the omnidirectional camera, the proposed procedure is independent of the sensor, easy to use, and flexible.

3.1 Introduction

3.1.1 State of the art

ACCURATE calibration of a vision system is necessary for any computer vision task requiring to extract metric information of the environment from 2D images, like in ego-motion estimation and structure from motion.

While a number of calibration methods has been developed for standard perspective cameras [11, 27], little work on omnidirectional cameras has been done.

Previous works on omnidirectional camera calibration can be classified into two different categories. The first one includes methods which exploit prior knowledge about the scene, such as the presence of calibration patterns [28,29] or plumb lines [30]. The second group covers techniques that do not use this knowledge; this includes calibration methods from pure rotation or planar motion of the camera [31], and self-calibration procedures, which are performed from point correspondences and epipolar constraint through minimizing an objective function [24, 32].

All mentioned techniques allow obtaining accurate calibration results but primarily focus on particular sensor types (e.g. hyperboloidal and paraboloidal mirrors or fisheye lenses). Furthermore, some of them require special setting of the scene and expensive equipment [29, 31]. For instance in [29] a fisheye lens with a 183° field of view is used as an omnidirectional sensor. Then, the calibration is performed by using a half-cylindrical calibration pattern perpendicular to the camera sensor, which rotates on a turntable.
In [30, 32], the authors treat the case of a parabolic mirror. In [30], it is shown that vanishing points lie on a conic section which encodes the entire calibration information. Thus, the projections of two sets of parallel lines suffice for the intrinsic camera calibration. However, this property does not apply to non-parabolic mirrors. Therefore, the proposed technique cannot be easily generalized to other kinds of sensors.

In contrast with the techniques mentioned so far, the methods described in [24, 32, 33] fall in the self-calibration category. These methods require no calibration pattern, nor a priori knowledge about the scene. The only assumption is the capability to automatically find point correspondences in a set of panoramic images of the same scene. Then, calibration is directly performed by epipolar geometry by minimizing an objective function. In [32], this is done by employing a parabolic mirror, while in [24, 33] a fisheye lens with a view angle greater than 180° is used. However, besides focusing on particular sensor types, the mentioned self-calibration techniques may suffer in case of tracking difficulties and of a small number of features points [34].

The calibration methods described so far focus on particular sensor types, such as parabolic and hyperbolic mirrors or fish-eye lenses. In contrast with

these methods, in the last years, novel calibration techniques have been developed, which apply to any central omnidirectional camera. For instance, in [33], the authors extend the geometric distortion model and the self-calibration procedure described in [24], including mirrors, fisheye lenses, and non-central cameras. In [19, 35], the authors describe a method for central catadioptric cameras using geometric invariants. They show that any central catadioptric system can be fully calibrated from an image of three or more lines. In [36], the authors present a general imaging model which encompasses most projection models used in computer vision and photogrammetry, and introduce theory and algorithms for a generic calibration concept. Finally, in [37, 38] the authors presented a calibration method for calibrating central omnidirectional cameras which uses a checkerboard-like calibration pattern; the authors used the unified imaging model by Geyer and Daniilidis [18].

3.1.2 Motivation and outline

The work described in this chapter also focuses on calibration of any central omnidirectional camera but aims at providing a technique that is very easy to apply also for the inexpert user. Indeed, this technique requires the use of a chessboard-like pattern that is shown by the user at a few different positions and orientations. The calibration points are the corner points of the chessboard and they are detected automatically using the "chessboard detection" algorithm described in [39].

The procedure described in this chapter is an extension of the work we published in [1–4]. On the author's webpage, we also provide a toolbox for Matlab that implements the proposed calibration procedure [5]. At the same time as this toolbox was written and the first results were published [1], we found that Mei had also implemented a similar toolbox for Matlab [40]. Although Mei's toolbox uses a different imaging model, we will briefly discuss the main differences with respect to our toolbox in Section 3.5.

The outline of this chapter is the following. Section 3.2 describes the proposed calibration procedure. Section 3.3 shows several experimental results both on simulated and real data. Section 3.4 introduces our Matlab toolbox (named OCamCalib) that implements the proposed calibration procedure.

3.2 Camera calibration

According to the Taylor imaging model that has been introduced in Section 2.3, to calibrate an omnidirectional camera we need to estimate parameters $\mathbf{A}$, $\mathbf{t}$, a_0, a_2, ..., and a_n so that all vectors $g(\mathbf{A}\mathbf{u}' + \mathbf{t})$ satisfy Equation 2.17.

In our approach, we decided to separate the estimation of these parameters into two stages. In the first one, we estimate the affine parameters $\mathbf{A}$, $\mathbf{t}$. In the other one, we estimate coefficients a_0, a_2, ..., and a_n.
Parameters $\mathbf{A}$, $\mathbf{t}$ describe the Affine transformation that relates the sensor plane to the camera plane (Fig. 2.11(b) and 2.11(c)). $\mathbf{A}$ is the stretch matrix and $\mathbf{t}$ is the translation vector $\overline{\mathbf{O_c}\mathbf{C_c}}$ (Fig. 2.11(c)).
To estimate $\mathbf{A}$, $\mathbf{t}$, we introduce a method which, unlike other previous works, does not require the visibility of the circular external boundary of the mirror (sketched by the ellipse in Fig. 2.11(c)). This method is based on an iterative procedure that starts by setting $\mathbf{A}$ equal to the identity matrix $\mathbf{I}$ and $\mathbf{t} = 0$. This assumption means that the camera plane and the sensor plane initially coincide. The correct elements of $\mathbf{A}$ will be estimated afterwards by non-linear refinement, while $\mathbf{t}$ will be estimated by an iterative search algorithm. This approach will be detailed in Section 3.2.4 and 3.2.5.
According to this, from now on we assume $\mathbf{A} = \mathbf{I}$ and $\mathbf{t} = 0$, which means $\mathbf{u}'' = \mathbf{u}'$. Thus, by substituting this relation in Equation (2.17), our projection equation can be read as follows:

$$\lambda \mathbf{p}'' = \lambda \begin{bmatrix} u' \\ v' \\ a_0 + a_2 {\rho'}^2 + ... + a_N {\rho'}^N \end{bmatrix} = \mathbf{P} \cdot \mathbf{X} \tag{3.1}$$

where $\rho' = \|\mathbf{u}'\|$ and u', v' are the pixel coordinates of the image point $\mathbf{u}'$. Also observe that now only parameters a_0, a_2, ..., a_n need to be estimated. From now on, we will refer to these parameters as intrinsic parameters.

During the calibration procedure, a planar pattern of known geometry is shown at different unknown positions and orientations which are related to the sensor coordinate system by a rotation matrix $\mathbf{R} \in \mathbf{SO(3)}$ and a translation $\mathbf{T} \in \mathbb{R}^{\mathbf{3}}$. $\mathbf{R}$ and $\mathbf{T}$ will be referred to as extrinsic parameters.
Let I^i be an observed image of the calibration pattern, $\mathbf{M}_j^i = [X_j^i, Y_j^i, Z_j^i]$ the 3D coordinates of its points in the pattern coordinate system, and $\mathbf{m}_j^i = [u_j^i, v_j^i]$ the correspondent pixel coordinates in the image plane. In this notation, superscript i indicates the observed pattern and subscript j indicates the j-th point on the i-th pattern.

Since we assumed the pattern to be planar, without loss of generality we have $Z_j^i = 0$. Then, equation (3.1) becomes:

$$\begin{aligned}
\lambda_j^i \cdot \mathbf{p}_j^i &= \lambda_j^i \cdot \begin{bmatrix} u_j^i \\ v_j^i \\ a_0 + a_2 {\rho_j^i}^2 + ... + a_N {\rho_j^i}^N \end{bmatrix} \\
&= \mathbf{P}^i \cdot \mathbf{X}_j^i \\
&= \begin{bmatrix} \mathbf{r}_1^i & \mathbf{r}_2^i & \mathbf{r}_3^i & \mathbf{T}^i \end{bmatrix} \cdot \begin{bmatrix} X_j^i \\ Y_j^i \\ 0 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{r}_1^i & \mathbf{r}_2^i & \mathbf{T}^i \end{bmatrix} \cdot \begin{bmatrix} X_j^i \\ Y_j^i \\ 1 \end{bmatrix} \qquad (3.2)
\end{aligned}$$

where $\mathbf{r}_1^i$, $\mathbf{r}_2^i$, $\mathbf{r}_3^i$ are the column vectors of $\mathbf{R^i}$.
Therefore, in order to solve for camera calibration, the extrinsic parameters have also to be determined for each pose of the calibration pattern.

3.2.1 Estimation of the extrinsic parameters

Before describing how to determine the extrinsic parameters, let us eliminate the dependence from the depth scale λ_j^i. This can be done by multiplying both sides of equation (3.2) vectorially by $\mathbf{p}_j^i$:

$$\begin{aligned}
\lambda_j^i \cdot \mathbf{p}_j^i \times \mathbf{p}_j^i &= \mathbf{p}_j^i \times \begin{bmatrix} \mathbf{r}_1^i & \mathbf{r}_2^i & \mathbf{T}^i \end{bmatrix} \cdot \begin{bmatrix} X_j^i \\ Y_j^i \\ 1 \end{bmatrix} = 0 \\
\triangleq \begin{bmatrix} u_j^i \\ v_j^i \\ a_0 + a_2 {\rho_j^i}^2 + ... + a_N {\rho_j^i}^N \end{bmatrix} &\times \begin{bmatrix} \mathbf{r}_1^i & \mathbf{r}_2^i & \mathbf{T}^i \end{bmatrix} \cdot \begin{bmatrix} X_j^i \\ Y_j^i \\ 1 \end{bmatrix} = 0. \qquad (3.3)
\end{aligned}$$

Now, let us focus on a particular observation i of the calibration pattern. From (3.3), we have that each point $\mathbf{p}_j$ on the pattern contributes three homogeneous equations (we removed superscript i to facilitate the reading):

$$v_j\ (r_{31}X_j + r_{32}Y_j + t_3)\ -\ g(\rho_j)\ (r_{21}X_j + r_{22}Y_j + t_2)\ =\ 0 \quad (3.4)$$

$$g(\rho_j)\ (r_{11}X_j + r_{12}Y_j + t_1)\ -\ u_j\ (r_{31}X_j + r_{32}Y_j + t_3)\ =\ 0 \quad (3.5)$$

$$u_j\ (r_{21}X_j + r_{22}Y_j + t_2)\ -\ v_j\ (r_{11}X_j + r_{12}Y_j + t_1)\ =\ 0 \quad (3.6)$$

with $g(\rho_j) = a_0 + a_2{\rho_j}^2 + ... + a_N{\rho_j}^N$. Observe that here X_j, Y_j, Z_j are known and so are u_j, v_j. Also, observe that only (3.6) is linear in the unknown r_{11}, r_{12}, r_{21}, r_{22}, t_1, t_2. Thus, by stacking all the unknown entries of (3.6) into a vector, we can rewrite Equation (3.6) for L points of the calibration pattern as a system of linear equations:

$$\mathbf{M} \cdot \mathbf{H} = \mathbf{0}, \quad (3.7)$$

where

$$\mathbf{H} = [r_{11}, r_{12}, r_{21}, r_{22}, t_1, t_2]^T$$

and

$$M = \begin{bmatrix} -v_1X_1 & -v_1Y_1 & u_1X_1 & u_1Y_1 & -v_1 & u_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -v_LX_L & -v_LY_L & u_LX_L & u_LY_L & -v_L & u_L \end{bmatrix}$$

A linear estimate of $\mathbf{H}$ can be obtained by minimizing the least-squares criterion $\min \|\mathbf{M} \cdot \mathbf{H}\|^2$, subject to $\|\mathbf{H}\|^2 = 1$. This is accomplished by using the Singular Value Decomposition (SVD). The solution of (3.7) is known up to a scale factor which can be determined uniquely since vectors $\mathbf{r}_1$, $\mathbf{r}_2$ are orthonormal. Because of the orthonormality, the unknown entries r_{31}, r_{32} can also be computed uniquely.

To resume, the first calibration step allows us to determine the extrinsic parameters r_{11}, r_{12}, r_{21}, r_{22}, r_{31}, r_{32}, t_1, t_2 for each pose i of the calibration pattern except for the translation parameter t_3. Parameter t_3 will be computed in the next step which concerns the estimation of the intrinsic parameters.

3.2.2 Estimation of the intrinsic parameters

In the previous step, we exploited Equation (3.6) to find the camera extrinsic parameters. Now, we substitute the estimated values in Equations (3.4), (3.5), and solve for the camera intrinsic parameters a_0, a_2, ..., a_n that describe the shape of the imaging function g. At the same time, we also compute the unknown t_3^i for each pose of the calibration pattern.

Similarly to what we did above, we stack all the unknown entries of (3.4), (3.5) into a vector and rewrite the equations as a system of linear equations. But now, we incorporate all K observations of the calibration pattern. We obtain the following system:

$$\begin{bmatrix} A_j^1 & A_j^1\rho_j^{1\,2} & \cdots & A_j^1\rho_j^{1\,N} & -v_j^1 & 0 & \cdots & 0 \\ C_j^1 & C_j^1\rho_j^{1\,2} & \cdots & C_j^1\rho_j^{1\,N} & -u_j^1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ A_j^K & A_j^K\rho_j^{K\,2} & \cdots & A_j^K\rho_j^{K\,N} & 0 & 0 & \cdots & -v_j^K \\ C_j^K & C_j^K\rho_j^{K\,2} & \cdots & C_j^K\rho_j^{K\,N} & 0 & 0 & \cdots & -u_j^K \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_2 \\ \vdots \\ a_N \\ t_3^1 \\ t_3^2 \\ \vdots \\ t_3^K \end{bmatrix} = \begin{bmatrix} B_j^1 \\ D_j^1 \\ \vdots \\ B_j^K \\ D_j^K \end{bmatrix} \quad (3.8)$$

where

$$\begin{aligned} A_j^i &= r_{21}^i X_j^i + r_{22}^i Y_j^i + t_2^i \,, \\ B_j^i &= v_j^i \left(r_{31}^i X_j^i + r_{32}^i Y_j^i \right) \,, \\ C_j^i &= r_{11}^i X_j^i + r_{12}^i Y_j^i + t_1^i \,, \\ D_j^i &= u_j^i \left(r_{31}^i X_j^i + r_{32}^i Y_j^i \right) \,. \end{aligned}$$

The linear least-squares solution of (3.8) can be obtained through the pseudoinverse-matrix method and so the intrinsic parameters a_0, a_2, ..., a_n

are determined.

To compute the best polynomial degree N, we start from $N = 2$; then, we increase N by unitary steps and compute the average reprojection error of all calibration points. The procedure stops when the error is below a given threshold ϵ.

3.2.3 Linear refinement of the intrinsic and extrinsic parameters

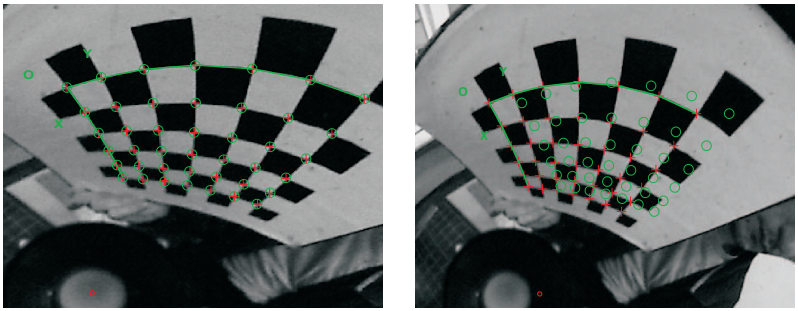
To resume, the second linear minimization step described in Section 3.2.2 computes the intrinsic parameters of the camera and simultaneously calculate the remaining extrinsic t_3^i .
The next two steps, which are going to be described here, aim at refining this primary estimation. This refinement is still performed by linear minimization. In Section 3.2.5, we will apply a non-linear refinement based on the maximum likelihood criterion. The structure of the linear refinement algorithm is the following:

1. The first step uses intrinsic parameters a_0, a_2, ..., a_n that were estimated in Section 3.2.2 and recomputes all extrinsic parameters r_{11}, r_{12}, r_{21}, r_{22}, r_{31}, r_{32}, t_1, t_2, t_3 by solving all together Equations (3.4), (3.5),(3.6). The problem leads to a linear homogeneous system which can be solved, up to a scale factor, using SVD. Then, the scale factor is determined uniquely by exploiting the orthonormality between vectors $\mathbf{r}_1$, $\mathbf{r}_2$.
2. In the second step, the extrinsic parameters recomputed in the first stage are substituted into Equations (3.4), (3.5) to further refine the intrinsic parameters. The problem leads to a linear system, which can be solved as usual by using the pseudoinverse-matrix method.

3.2.4 Detection of the center of distortion

A peculiarity of our calibration technique is that it requires the minimum user interaction. One of the features that accomplishes this task is its capability of identifying the center of the distortion $\mathbf{O_c}$ (Fig. 2.11(c)) even when the external boundary of the mirror (or lens) is not visible in the image.

At the beginning of Section 3.2, we made the following assumptions for $\mathbf{A}$ and $\mathbf{t}$, namely $\mathbf{A} = \mathbf{I}$ and $\mathbf{t} = 0$. Then, we derived the equations for solving



a b

Figure 3.1: (a) When the position of the center is correct, the 3D points of the checker board do correctly back-project (green rounds) onto the calibration points (red crosses). (b) Conversely, when the position of the center is wrong, the points do not back-project onto the real calibration points.

for the intrinsic and extrinsic parameters, which are valid only under those assumptions.
In Fig. 3.1(a), can be seen what happens when the position of the center is correct. The red crosses are the input calibration points selected by the user. The green rounds are the 3D points reprojected onto the images according to the intrinsic and extrinsic parameters estimated by the calibration. As can be seen, the 3D points perfectly overlay the input points, meaning that the calibration worked properly. Fig. 3.1(b) shows the result when the input position of the center is wrong, that is, when the reprojection error is large. Motivated by this observation, we performed many calibration trials using different center positions and, for each trial, we computed the Sum of Squared Reprojection Errors (SSRE). As a result, we verified that the SSRE always has a global minimum at the correct center location.
This result leads us to an exhaustive search of the center of distortion $\mathbf{O_c}$, which stops when the difference between two potential center locations is smaller than a certain ϵ (we used $\epsilon = 0.5$ pixels). The algorithm is the following:

1. At each step of this iterative search, a fixed number of candidate center locations is uniformly selected from a given image region (see Fig. 3.2).
2. For each one of these points, calibration is performed using that point as a potential center location and SSRE is computed.
3. The point providing the minimum SSRE is taken as a potential center.

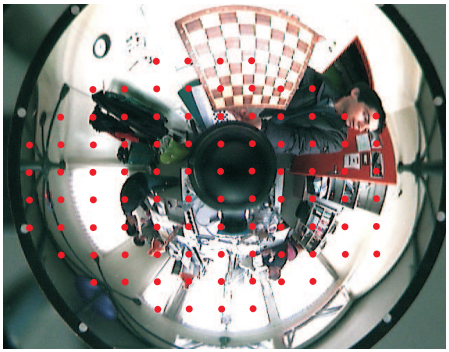


Figure 3.2: An omnidirectional image used for calibration with a chessboard used as a calibration pattern. The red points identify the candidate center locations taken during the first step of the algorithm. At each step, the candidate points occupy a smaller and smaller region around the final convergence point.

4. The search proceeds by selecting other candidate locations in the region around that point and steps 1, 2, and 3 are repeated until the stop-condition is satisfied.

Observe that by using a binary search algorithm the computational cost of this iterative search is very low. Indeed, each calibration trial, which implements the technique described in the previous sections, can be performed in Matlab in a negligible amount of time. Using eleven iterations, the algorithm takes less than 2 seconds on a normal notebook to stop.

At this point, the reader might be wondering how we estimate the elements of matrix $\mathbf{A}$. In fact at the beginning of Section 3.2 we assumed $\mathbf{A} = \mathbf{I}$. The iterative algorithm mentioned above exhaustively searches the location of the center $\mathbf{O_c}$ without modifying $\mathbf{A}$. The reason of doing so is that the eccentricity of the external boundary of an omnidirectional image is usually close to zero, which means $\mathbf{A} \approx \mathbf{I}$. Therefore, we chose to estimate $\mathbf{A}$ in a second stage by using a non-linear minimization method. This is described in Section 3.2.5.

3.2.5 Non-linear refinement

The linear solution given in Sections 3.2.3 and 3.2.4 is obtained through minimizing an algebraic distance which is not physically meaningful. Because of this, we chose to refine the calibration parameters through maximum likelihood inference.

Let us assume that we are given K images of the calibration pattern, each one containing L corner points. Furthermore, let us assume that the image points are corrupted by independent and identically distributed noise. Then, the maximum likelihood estimate can be obtained by minimizing the following functional:

$$E = \sum_{i=1}^{K} \sum_{j=1}^{L} \left\| \mathbf{u}_j^i - \hat{\mathbf{u}}(\mathbf{R}^i, \mathbf{T}^i, \mathbf{A}, \mathbf{O_c}, a_0, a_2, ..., a_N, \mathbf{X}_j^i) \right\|^2 \qquad (3.9)$$

where $\hat{\mathbf{u}}(\mathbf{R}^i, \mathbf{T}^i, \mathbf{A}, \mathbf{O_c}, a_0, a_2, ..., a_N, \mathbf{X}_j^i)$ is the reprojection of the scene point $\mathbf{X}_j^i$ on the i-th pattern according to Equation (3.2) and $\mathbf{R}^i$ and $\mathbf{T}^i$ describe the orientation and position of the pattern. In our implementation, $\mathbf{R}^i$ is parameterized by a 3×1 vector related to $\mathbf{R}^i$ by the Rodrigues formula. Observe that in 3.9 we incorporated both the stretch matrix $\mathbf{A}$ and the center of distortion $\mathbf{O_c}$.

By minimizing the functional defined in (3.9), we actually find the calibration parameters which minimize the reprojection error. In order to speed up the convergence, we decided to split the non-linear minimization into two steps. The first one refines the extrinsic parameters, ignoring the intrinsic ones. The second step uses the extrinsic parameters just estimated and refines the intrinsic ones. By performing many simulations, we found that this splitting does not affect the final result with respect to a global minimization.

To minimize (3.9), we used the Levenberg-Marquadt algorithm [41, 42] as implemented in the Matlab function *lsqnonlin*. The algorithm requires an initial guess which can be obtained using the linear technique described in Section 3.2.3. As a first guess for $\mathbf{A}$, we used the identity matrix $\mathbf{I}$, while for $\mathbf{O_c}$ we used the position estimated through the iterative procedure explained in Section 3.2.4.

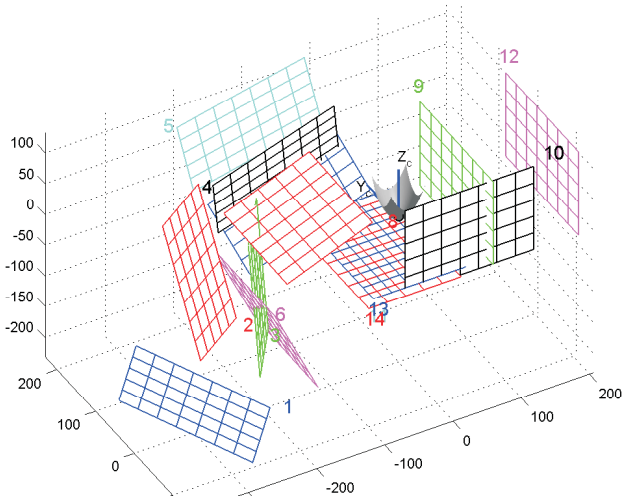


Figure 3.3: A picture of our simulator showing several calibration patterns and the virtual omnidirectional camera at the axis origin.

3.3 Results

In this section, we evaluate the performance of the proposed calibration procedure through several experimental results on both simulated and real data.

3.3.1 Simulations

The reason for using simulation is that we can monitor the actual performance of the calibration and compare the results with a known ground truth. The simulator we developed allows us to choose both the intrinsic parameters (i.e. imaging function g) and the extrinsic parameters (i.e. the rotation and translation matrices of the simulated pattern, that is, $\mathbf{R}^i$, $\mathbf{T}^i$). Furthermore, we can also choose the size of the virtual pattern and the number of calibration points as in the real case. A pictorial image of the simulated scenario is shown in Fig. 3.3.

Our virtual calibration pattern is a planar grid containing 48 corner points which are arranged in 6 rows and 8 columns (in the remainder of this section these points will be also referred to as calibration points). The size of the simulated pattern is $150 \times 210mm$. Concerning g, we chose a 4th order poly-

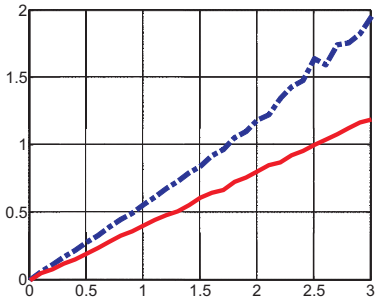


Figure 3.4: The RMS reprojection error as a function of the noise level σ. For every σ, 100 calibration trials were performed. Results with the linear minimization alone (dashed line in blue) and with the non-linear refinement (solid line in red). Both units are in pixel.

nomial whose parameters are set according to those obtained by calibrating a real omnidirectional camera. Then, we set to 900×1200 pixels the image size of the virtual camera.

Performance with respect to the noise level

In this simulation experiment, we studied the robustness of our calibration technique in case of inaccuracy in detecting the calibration points. To this end, we generated 14 synthetic poses of the calibration pattern and all calibration points were reprojected onto the camera plane. Then, Gaussian noise with zero mean and standard deviation σ was added to the reprojected image points. We varied the noise level from $\sigma = 0.1$ pixels to $\sigma = 3.0$ pixels and for each σ we performed 100 independent calibration trials. The results shown are the average.

Figure 3.4 plots the Root Mean Square (RMS) reprojection error as function of σ. The reprojection error is defined as the distance in pixels between the reprojected 3D points and the input image points. Figure 3.4 shows both the plots obtained using either the linear minimization method alone or with non-linear refinement. As can be seen, the error increases linearly with the noise level in both cases. Observe that the error of the non-linear estimation is always less than that of the linear estimation. Furthermore note that when $\sigma = 1.0$ pixels, which is larger than the noise introduced by the user in prac-

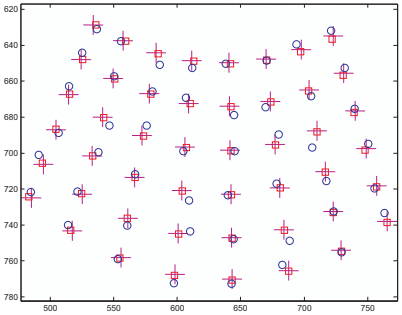


Figure 3.5: An image of the calibration pattern, projected onto the simulated omnidirectional image. Calibration points are affected by noise with $\sigma = 3.0$ pixels (blue rounds). Ground truth (red crosses). Reprojected points after the calibration (red squares).

tical situations (e.g. when clicking on the points), the error of the non-linear method is less than 0.4 pixels.

Figure 3.5 shows the points of a simulated grid after reprojection onto the image. The ground truth is represented by red crosses, while the blue rounds represent the calibration points perturbed by noise with $\sigma = 3.0$ pixels. Note that despite the large amount of noise, calibration was able to compensate for the large error introduced. Indeed, after calibration the reprojected calibration points (red squares) appear very close to the ground truth.

We also wanted to evaluate the accuracy in estimating the extrinsic parameters $\mathbf{R}^i$ and $\mathbf{T}^i$ of each pattern. To this end, Fig. 3.6 plots the mean absolute error (measured in mm) of the grid origin (i.e. X_0^i, Y_0^i, Z_0^i) as a function of σ, while Fig. 3.7 plots the mean absolute error of the plane normal. The results are shown for all 14 poses of the pattern.

Observe that the position error in most cases increases linearly with σ. Furthermore, observe that the error ranges on average between 0 and $4mm$. There is however one case (the first plot upper left in Fig. 3.6) where the position error reaches $40mm$; in fact, this error belongs to the most angled and distant view of the pattern (i.e. the pattern labeled as no. 1 in Fig. 3.3). The same comment concerns Fig. 3.7. In this case, the error of the plane normal ranges on average between 0 and 2 degrees. In some cases the error

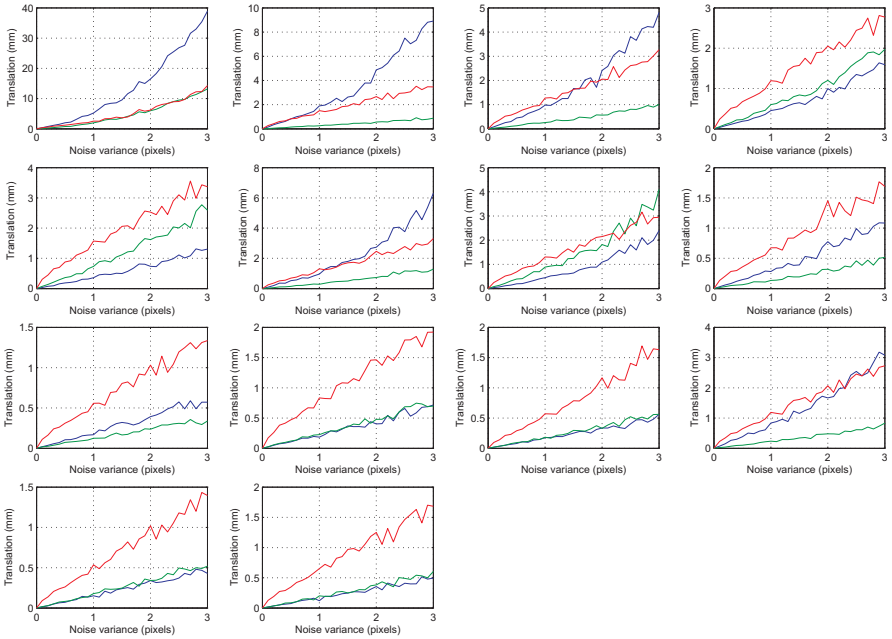


Figure 3.6: The absolute error (mm) of the grid positions vs. σ (pixels). The error along the x, y, and z coordinates is represented respectively in red, blue, and green.

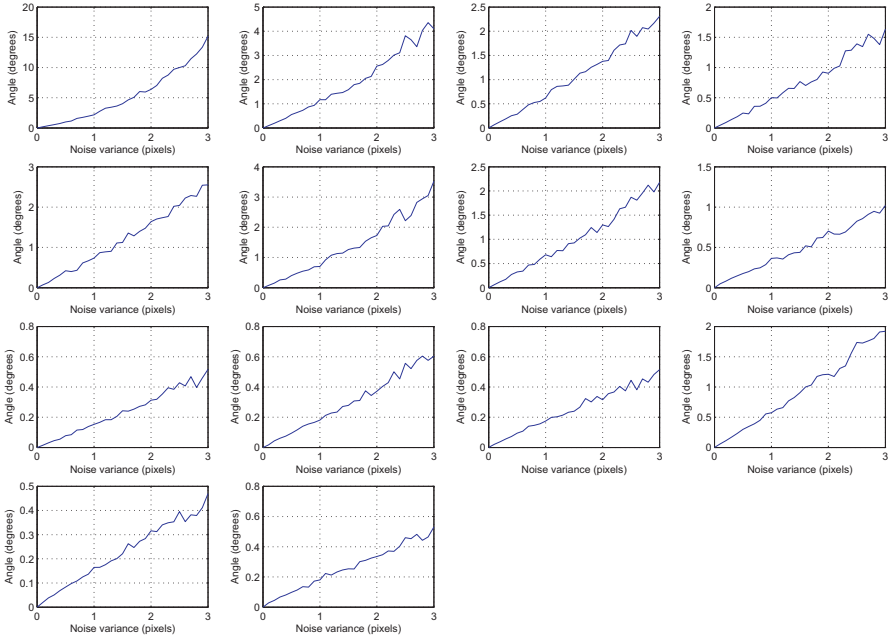


Figure 3.7: The absolute error (degrees) of the plane normals vs. σ (pixels).

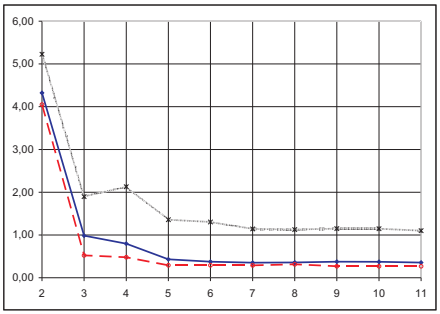


Figure 3.8: Mean reprojection error vs. the number of images for different polynomial degrees: 2nd order (black $\times$), 3rd order (blue $\cdot$) and 4th order (red $\circ$).

is even smaller than 1 degree. As before, the largest error occurs for the first plot upper left.

3.3.2 Real experiments

Performance w.r.t. the number of images and to the polynomial degree

In this section, we present some calibration results on real data. In these experiments, we calibrated a catadioptric system composed of a KAIDAN 360° One VR hyperbolic mirror and a SONY CCD camera the resolution of 900×1200 pixels. In the first experiment, we investigated the performance of our technique with respect to the number of images of the calibration grid for a given polynomial degree. We varied the number of images from 2 to 11, and for each set we performed calibration and computed the RMS reprojection error. The RMS error as a function of the number of images is plotted in Fig. 3.8 for different polynomial degrees. Note that the error decreases as the number of images increases. Furthermore, by using a 4th-order polynomial we obtain the minimum RMS error (equal to 0.27 pixels when using 11 images). A 3rd-order polynomial also provides a similar performance when more than four images are taken. Conversely, when using a 2nd-order polynomial, the RMS error is larger. Thus, in practical situations we always used a 4th-order polynomial.

In Table 3.1, the RMS error and the standard deviation (STD) after non-

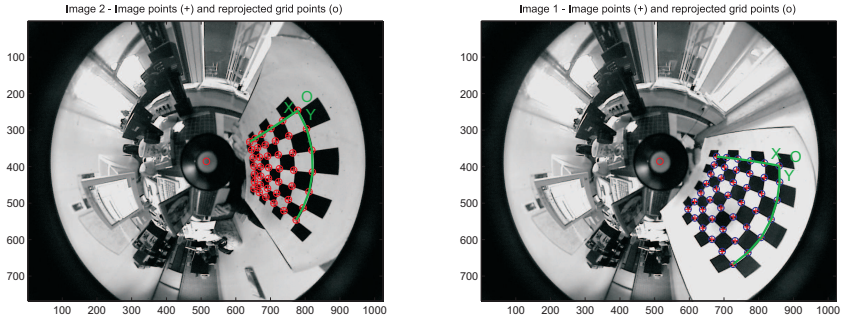


Figure 3.9: Two sample images of the calibration pattern used during our experiments. The red "+" are the input points, while the "*o*" are the reprojected points.

linear refinement are shown for all eleven images contained in the dataset; the results are obtained using a 4th-order polynomial. In Fig. 3.10, the cartesian distribution of the reprojection error for all images of the set is shown; different colors are used to distinguish the contribution of each image.

Table 3.1: Reprojection error for the hyperbolic mirror

Pattern	RMS (*pixels*)	STD (*pixels*)
1	0.27	0.21
2	0.31	0.17
3	0.29	0.22
4	0.30	0.15
5	0.24	0.12
6	0.28	0.14
7	0.30	0.14
8	0.30	0.19
9	0.28	0.16
10	0.27	0.16
11	0.34	0.17

Performance with different cameras

As we mentioned in Section 2.3, our camera model encompasses both catadioptric and dioptric central omnidirectional cameras. To evaluate the cal-

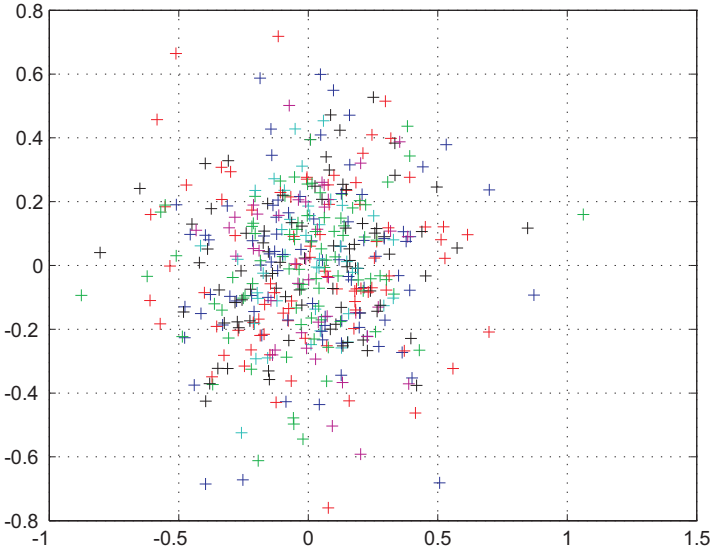


Figure 3.10: Reprojection error

ibration performance on different cameras, we tested the Nikon and Sigma fisheye lenses and parabolic and hyperbolic mirrors. For each sensor, we collected ten images and applied the calibration procedure described in this chapter. The values of RMS error and standard deviation for each camera are shown in Table 3.2.

Table 3.2: Reprojection error for different cameras

Pattern	RMS (*pixels*)	STD (*pixels*)
Hyperbolic mirror	0.28	0.20
Parabolic mirror	0.31	0.18
Nikon fisheye	0.30	0.23
Sigma fisheye	0.33	0.18

Structure from motion

An indirect method to evaluate the quality of calibration for a real camera is by recovering the 3D structure of an object from its images and checking then the quality of the reconstruction. This problem is well known in the

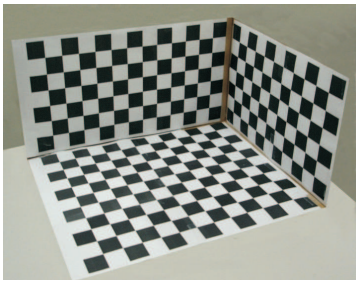


Figure 3.11: Trihedron

computer vision community as “structure from motion”.

The object we used in this experiment was a trihedron made up of three orthogonal chessboard-like patterns of known geometry (Fig. 3.11). After calibrating the camera, we took two images of the trihedron from two different unknown positions (Fig. 3.12). Then, we selected manually several point correspondences from both views and we applied the 8-point algorithm [43,44] followed by bundle adjustment. The results of the reconstruction, before and after rendering, are shown in Fig. 3.13 and 3.14 respectively.

As the reconstruction with one single camera can be done up to a scale factor, we recovered the scale factor by comparing the average size of a reconstructed checker with the real size of a checker. In the end, we computed the angles between the three planes fitting the reconstructed points and we found the following values: 94.6°, 86.8° and 85.3°. Moreover, the average distances of these points from the fitted planes were respectively 0.05 cm, 0.75 cm and 0.07 cm. Finally, being the size of each checker 6.0 cm $\times$ 6.0 cm, we also calculated the dimension of every reconstructed checker and we found an average error of 0.3 cm. These results comply with the expected orthogonality of the surfaces and the size of the checkers in the ground truth.

3.4 Implementation of the OCamCalib Toolbox

The reason we implemented the OcamCalib Toolbox for Matlab is to allow any user to easily and quickly calibrate his own omnidirectional camera. The OCamCalib toolbox can be freely downloaded from our webpage (or google

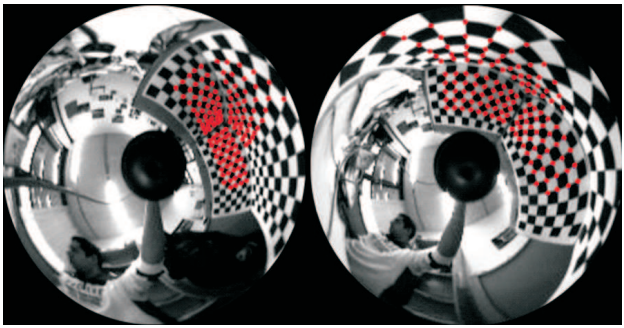


Figure 3.12: Input images for “structure from motion” with the points to be recovered (in red).

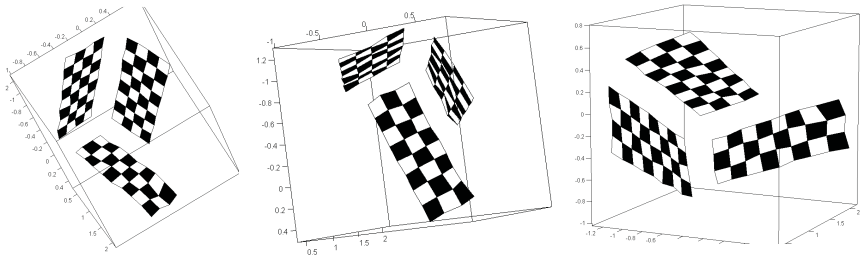


Figure 3.13: Result of “structure from motion”.

Figure 3.14: Three views of two recovered faces after rendering.

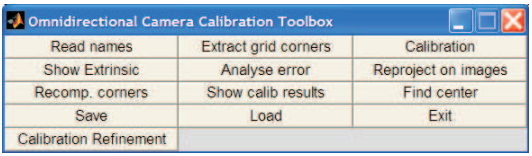


Figure 3.15: The graphical user interface of the OCamCalib Toolbox

for "ocamcalib"). The outstanding features of the toolbox are the following:

1. Capability of calibrating different kinds of central omnidirectional cameras without any knowledge about the parameters of the camera or about the shape of the mirror.
2. Automatic detection of the center.
3. Visual feedback about the quality of the calibration result by reprojecting the 3D points onto the input images.
4. Completely automatic detection of the corner points of the pattern.

The user interface of the toolbox is depicted in Fig. 3.15. After having collected a few pictures of a chessboard shown all around the omnidirectional camera (see figure 3.9), the images can be loaded for calibration (i.e. use "Read names"). In the second step, the user can start selecting the corner points of the pattern using the "Extracting grid corners" tool. By this tool, the toolbox attempts to find all corner points of each view of the pattern by following the algorithm described in [39]. In case it fails, the user is asked to click on the missing points. To achieve high accuracy in the selection of the input points, the clicking is assisted by a Harris corner detector [45].

In the third step, the calibration can be done by means of two tools. The "Calibration" tool will ask the user to specify the position of the center in case he knows it; if not, the user can directly use the "Find center" tool, which applies the center detection algorithm described in 3.2.4. In both cases, calibration is done using the linear estimation technique mentioned in Section 3.2.3. The optimal calibration parameters in the maximum likelihood sense can be estimated by the "Calibration Refinement" tool, which implements the non-linear minimization method described in Section 3.2.5. After the previous steps, the user can choose among several tools:

1. "Show Extrinsic" visualizes the reconstructed 3D poses of the grid in

the camera reference frame (as in Fig. 3.3).
2. "Analyze error" visualizes the reprojection error of each calibration point along the xy-axes (Fig. 3.10).
3. "Reproject on images" reprojects all the 3D points onto the images according to the calibrated parameters (Fig. 3.9).
4. "Recompute corners" attempts to recompute automatically the position of every corner point that was hand-selected by the user. This is done by using the reprojected 3D points as initial guess locations for the corners.

After the calibration, all parameter can be accessed through structure *ocammodel*. The calibrated camera model can then be used for other applications by means of the following two functions:

1. $\mathbf{m} = world2cam(\mathbf{M}, ocammodel)$ reprojects the 3D point $\mathbf{M}$ onto the image and returns its pixel coordinates $\mathbf{m}$.
2. $\mathbf{M} = cam2world(\mathbf{m}, ocammodel)$ for every image point $\mathbf{m}$ returns the 3D coordinates of the correspondent vector $\mathbf{M}$ emanating from the single effective viewpoint. This function is the inverse of the previous one.

3.5 Conclusion

In this chapter, we presented a new technique and toolbox for calibrating central omnidirectional cameras, which use the new unified imaging model (i.e. Taylor model) presented in Section 2.3.
The proposed procedure is very fast and completely automatic as the user is only asked to collect a few images of a chessboard-like pattern, while the detection of the input points is performed automatically by the toolbox itself. No a priori knowledge of the motion of the pattern is required, nor a prior initialization of the camera model. The calibration parameters (i.e. the coefficients of the Taylor polynomial, the Affine transformation, and the extrinsic parameters $\mathbf{R}^i$, $\mathbf{T}^i$) are estimated by solving a four-step least-squares linear minimization problem (Section 3.2.3), followed by a non-linear refinement which is based on the maximum likelihood criterion (Section 3.2.5).

Unlike all previous methods, we described also an algorithm to compute the center of distortion in omnidirectional images without exploiting the visibility of the external border of the mirror (or lens). The center is automati-

cally computed starting from the input points themselves (Section 3.2.4).

Furthermore, we used simulated data to study the robustness of our calibration technique in case of inaccuracy in detecting the calibration points. We showed that the non-linear refinement significantly improves the calibration accuracy and that accurate results can be obtained by using only a few images (Section 3.3.1).

Then, we calibrated four different and real omnidirectional cameras, both dioptric and catadioptric (Section 3.3.2). The calibration was very accurate as for all cameras we obtained a reprojection error of about 0.3 pixels with an image resolution of 900×1200 pixels. We also showed the accuracy of the result with a "structure from motion" experiment (Section 3.3.2).

In Section 3.4, we provided a toolbox for Matlab (named OCamCalib) which implements the entire calibration procedure; this toolbox is available on-line at the authors' web page [5].
Furthermore, as we mentioned in Section 3.1.2, recently a similar toolbox for Matlab for calibrating central omnidirectional cameras has been implemented by Mei [40]. In contrast to our method, Mei's toolbox takes advantage of the unified imaging model by Geyer and Daniilidis [18].
A comparison between the two methods is not the purpose of this chapter, however, except for the different imaging models they use, the two methods mainly differ in the way the center of distortion is computed. In Mei's toolbox this is done by exploiting the visibility of the circular external border of the mirror; in particular, a circle is fitted to the border and the coordinates of the center are then extracted from this circle. Conversely, our method does not rely on the visibility of the mirror border; indeed, it just uses the coordinates of the input points to run the iterative algorithm described in Section 3.2.4. This allows our method to work also when the mirror boundary is not visible. Nevertheless, a comparison of the performance between ours and Mei's method has been recently published by Frank et all. in [26].

Chapter 4

Feature extraction and matching

This chapter presents a new and robust method for extracting and matching vertical features in omnidirectional images. Matching robustness is achieved by creating a descriptor which is unique and distinctive for each feature. Furthermore, the proposed descriptor is invariant to rotation and slight changes of illumination. The robustness of the approach is validated through real experiments with a wheeled robot equipped with an omnidirectional camera. We show that vertical lines are very well extracted and tracked during the motion of the robot.

4.1 Introduction

4.1.1 State of the art

One of the most important problems in vision based robot navigation systems is the search for correspondences in images taken from different viewpoints. In the last decades, the feature correspondence problem has been largely investigated for standard perspective cameras. Furthermore, several works have provided robust solutions for wide-baseline stereo matching, structure from motion, ego-motion estimation, and robot navigation (see [46–53]). Some of these works normalize the region around each detected feature using a local affine transformation, which attempts to compensate for the distortion introduced by the perspective projection. However, such methods cannot be

directly applied to images taken by omnidirectional imaging devices because of the non-linear distortions introduced by their large field of view.

In order to apply those methods, one needs first to generate a perspective view out of the omnidirectional image, provided that the imaging model is known and that the omnidirectional camera possesses a single effective viewpoint [15]. An application of this approach can be found in [54]. There, the authors generate perspective views from each region of interest of the omnidirectional image. This image unwrapping removes the distortions of the omnidirectional imaging device and enables the use of state-of-the-art wide-baseline algorithms designed for perspective cameras.
Nevertheless, other researchers have attempted to apply to omnidirectional images standard feature detectors and matching techniques which have been traditionally employed for perspective images. In [55], for instance, the authors check the candidate correspondences between two views using RANSAC algorithm.

Finally, other works have been developed, which extract one-dimensional features from new images called Epipolar plane images, under the assumption that the camera is moving on a flat surface [56]. These images are generated by converting each omnidirectional picture into a 1D circular image, which is obtained by averaging the scan lines of a cylindrical panorama. Then, 1D features are extracted directly from such kinds of images.

In this chapter, we deal with real world vertical features because they are predominant in structured environments. In our experiments, we used a wheeled robot equipped with a catadioptric omnidirectional camera with the mirror axis perpendicular to the plane of motion (Fig. 5.4). If the environment is flat, this implies that all world vertical lines are mapped to radial lines on the camera image plane.

The use of vertical line tracking is not new in the Robotics community. Since the beginning of machine vision, roboticians have been using vertical lines or other sorts of image measure for autonomous robot localization or place recognition.
Several works dealing with automatic line matching have been proposed for standard perspective cameras and can be divided into two categories: those that match individual line segments; and those that match groups of line segments. Individual line segments are generally matched on their geometric attributes (e.g. orientation, length, extent of overlap) [57–59]. Some such

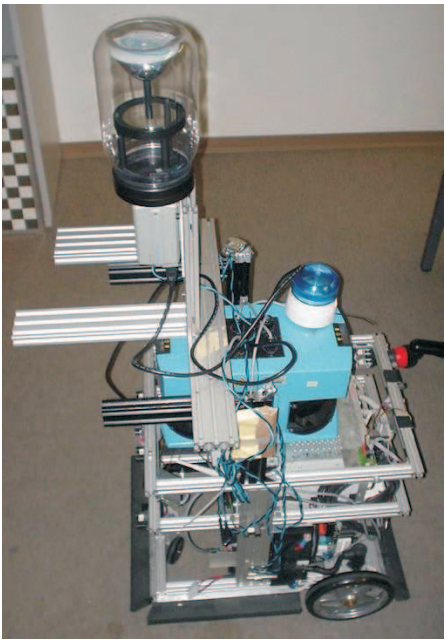


Figure 4.1: The robot used in our experimets equipped with encoder sensors on the wheels, an omnidirectional camera, and two laser range finders

as [60–62] use a nearest line strategy which is better suited to image tracking where the images and extracted segments are similar. Matching groups of line segments has the advantage that more geometric information is available for disambiguation. A number of methods have been developed around the idea of graph-matching [63–66]. The graph captures relationships such as "left of", "right of", cycles, "collinear with" etc, as well as topological connectedness. Although such methods can cope with more significant camera motion, they often have a high complexity and again they are sensitive to error in the segmentation process.

Besides these methods, other approaches to individual line matching exist, which use some similarity measure commonly used in template matching and image registration (e.g. Sum of Squared Differences (SSD), simple or Normalized Cross-Correlation (NCC), image histograms [67]).
An interesting approach was proposed in [68]. Besides using the topological information of the line, the authors also used the photometric neighbourhood

of the line for disambiguation. Epipolar geometry was then used to provide a point to point correspondence on putatively matched line segments over two images and the similarity of the line's neighbourhoods was then assessed by cross-correlation at the corresponding points.

A novel approach, using the intensity profile along the line segment, was proposed in [69]. Although the application of the method was to wide baseline point matching, the authors used the intensity profile between two distinct points (i.e. a line segment) to build a distinctive descriptor. The descriptor is based on affine invariant Fourier coefficients that are directly computed from the intensity profile.

The methods cited above were defined for perspective images but the same concepts have been also used by roboticians in omnidirectional images under certain circumstances. The use of omnidirectional vision even facilitated the task because of the 360° field of view (see [70–72]). However, to match vertical lines among different frames only mutual and topological relations have been used (e.g. neighborhood or ordering constraints) sometimes along with some of the similarity measures cited above (e.g. SSD, NCC).

4.1.2 Outline

In this chapter, we describe how we built our robust descriptor for vertical lines. We show that the descriptor is unique and very distinctive for each feature and is invariant to rotation and slight changes of illumination. We characterize the performance of the proposed descriptor on a large image dataset by taking into account the sensitiveness to image noise and to other different parameters of the descriptor. The robustness of the descriptor is validated through real experiments using our robot. This chapter extends our previous work [8].

The outline of the chapter is the following. First, we describe our procedure to extract vertical lines (Section 4.2) and build the feature descriptor (Section 4.3). In Section 4.4, we provide our matching rules. In Section 4.5, we compare it with other similarity measures, while the analysis of the performance and the results of tracking are respectively presented in Sections 4.6 and 4.7.

4.2 Vertical line extraction

Our platform consists of a wheeled robot equipped with an omnidirectional camera looking upwards (see Fig. 5.4). In our arrangement, we set the camera-mirror system perpendicular to the floor where the robot moves. This setting guarantees that all vertical lines are mapped to radial lines on the camera image plane (Fig. 4.2) In this section, we detail our procedure to extract prominent vertical lines. Our procedure consists of five steps.

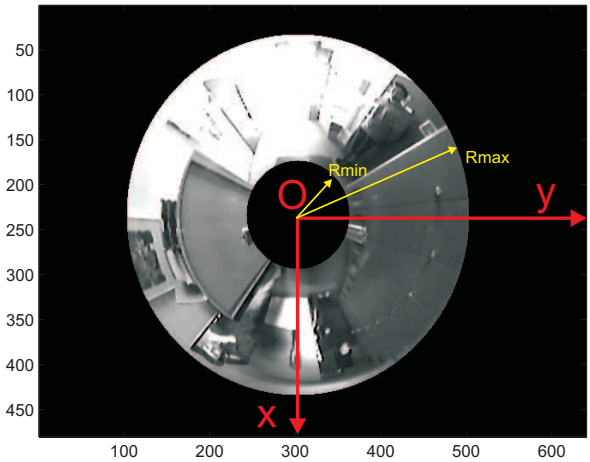


Figure 4.2: An image taken by our omnidirectional camera.

The first step towards vertical line extraction is the detection of the image center (i.e. the point where all radial lines intersect in). As the circular external boundary of the mirror is visible in the image, we used a circle detector to determine the coordinates of the center. Note that because the diameter of the external boundary is known and does not change dramatically during the motion, the detection of the center can be done very efficiently and with high accuracy on every frame (this guarantees to cope also with the vibrations of the platform).

The second step is the computation of the image gradients. We compute the two components $\mathbf{I_x}$, $\mathbf{I_y}$ of the image gradient by convolving the input im-

age $\mathbf{I}$ with the two Sobel masks. From $\mathbf{I_x}$, $\mathbf{I_y}$, we can calculate the magnitude $\mathbf{M}$ and the phase $\mathbf{\Phi}$ of the gradients as

$$\mathbf{M} = \sqrt{\mathbf{I_x}^2 + \mathbf{I_y}^2}, \quad \mathbf{\Phi} = atan2(\mathbf{I_y}, \mathbf{I_x}). \tag{4.1}$$

Then, we do a thresholding on $\mathbf{M}$, $\mathbf{\Phi}$ by retaining those vectors whose orientation looks towards the image center up to $\pm 5^\circ$. This 10° tolerance allows us to handle the effects of floor irregularities on the appearance of vertical lines. After this thresholding, we apply edge thinning and we obtain the binary edge map depicted in Fig. 4.3.

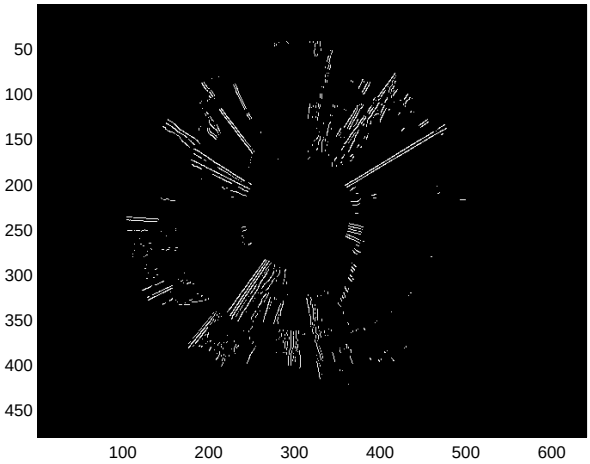


Figure 4.3: Edge image of Fig. 4.2.

The third step consists in detecting the most reliable vertical lines. To this end, we divide the omnidirectional image into 720 predefined uniform sectors, which give us an angular resolution of 0.5°. By summing up all binary pixels that vote for the same sector, we obtain the histogram shown in Fig. 4.4. Then, we apply non-maxima suppression to identify all local peaks.

The final step is histogram thresholding. As observed in Fig. 4.3, there are many potential vertical lines in structured environments. In order to keep

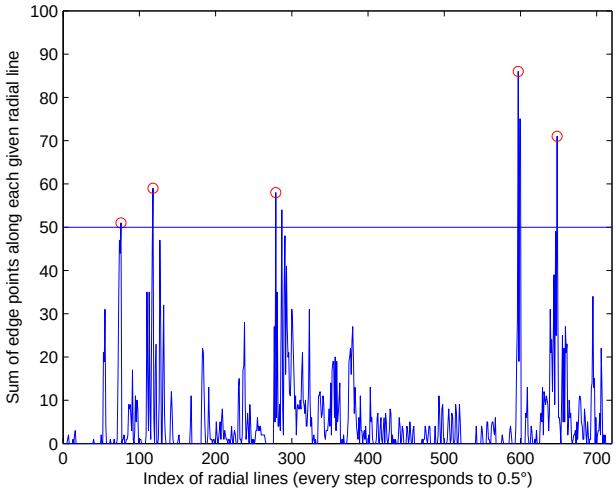


Figure 4.4: Number of binary pixels voting for a given orientation angle.

the most reliable and stable lines (e.g. Fig. 4.5), we put a threshold on the line length. As observed in Fig. 4.4), we set our threshold equal to 50% of the maximum allowed line length, i.e. $R_{max} - R_{min}$. Obviously, this choice is purely arbitrary and a different criterion could be used depending on the purpose (for instance, one can decide to have always a constant number of lines per each frame). Another criterion is to have this threshold adaptive. This can be done by computing the mean $\bar{l}$ and the standard deviation σ_l of all line lengths in the observed image and keeping only those lines whose length l satisfies $l \geq \bar{l} + 3\sigma_l$.

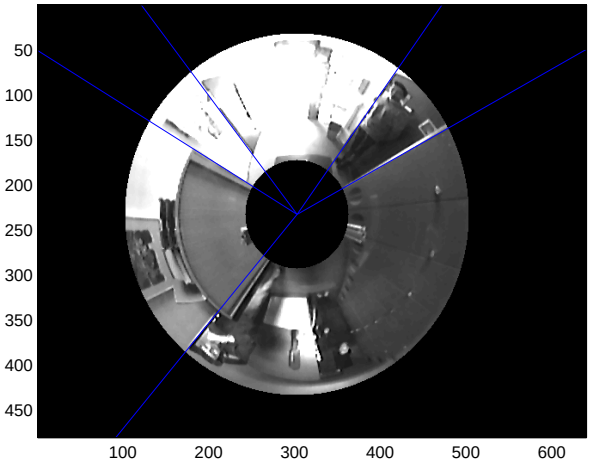


Figure 4.5: Extraction of the most reliable vertical features from an omnidirectional image.

4.3 Building the descriptor

In Section 4.4, we will describe our method for matching vertical lines between consecutive frames while the robot is moving. To make the feature correspondence robust to false positives, each vertical line is given a descriptor which is unique and distinctive for each feature. Furthermore, this descriptor is invariant to rotation and slight changes of illumination. In this way, finding the correspondent of a vertical line can be done by looking for the line with the closest descriptor. In the next subsections, we describe how we built our descriptor.

4.3.1 Rotation invariance

Given a radial line, we divide the space around it into three equal non-overlapping circular areas such that the radius r_a of each area is equal to $(R_{max} - R_{min})/6$ (see Fig. 4.7).
Then, we smooth each area with a Gaussian window (Fig. 4.6) with $\sigma_G = r_a/3$ and compute the image gradients (magnitude $\mathbf{M}$ and phase $\mathbf{\Phi}$) within

each of these areas.
Concerning rotation invariance, this is achieved by redefining the gradient phase $\mathbf{\Phi}$ of all points relatively to the radial line's angle θ (see Fig. 4.7).

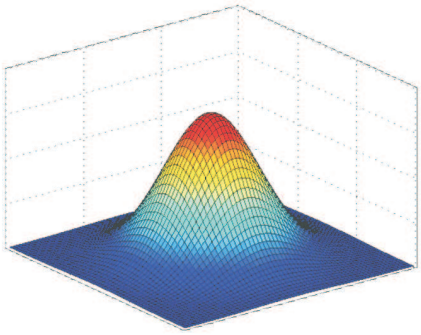


Figure 4.6: Gaussian smoothing filter

4.3.2 Orientation histograms

To make the descriptor robust to false matches, we split each circular area into two parts and consider each one individually (Fig. 4.8). In this way, we preserve the information about what we have on the left and right sides of the feature.

For each side of each circular area, we compute the gradient orientation histogram (Fig. 4.9). The whole orientation space (from -π to π) is divided into N_b equally spaced bins. In order to decide how much of a certain gradient magnitude m belongs to the adjacent inferior bin b and how much to the adjacent superior bin, each magnitude m is weighted by the factor $(1 - w)$, where

$$w = N_b \frac{\varphi - b}{2\pi}, \tag{4.2}$$

with φ being the observed gradient phase in radians. Thus, $m(1-w)$ will vote for the adjacent inferior bin, while mw will vote for the adjacent superior bin.

According to what we mentioned so far, each bin contains the sum of the weighted gradient magnitudes which belong to the correspondent orientation interval. We observed that this weighted sum made the orientation histogram

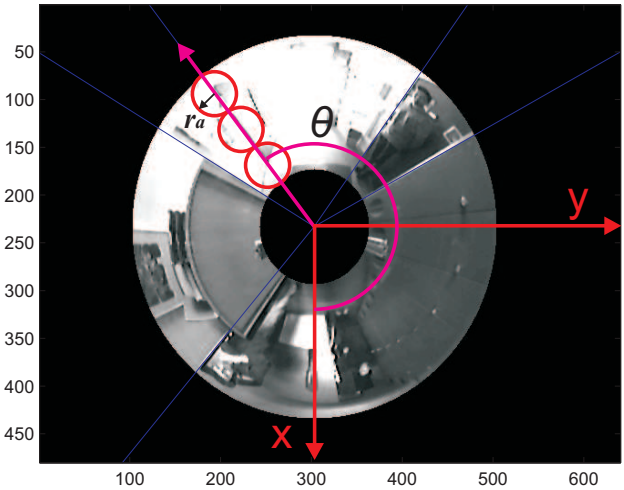


Figure 4.7: Extraction of the circular areas. To achieve rotation invariance, the gradient phase $\mathbf{\Phi}$ of all points is redefined relatively to the radial line's angle θ.

more robust to image noise. Finally, observe that the orientation histogram is already rotation invariant because the gradient phase has been redefined relatively to the radial line's angle (Section 4.3.1).

To resume, in the end we have three pairs of orientation histograms:

$$\begin{aligned} \mathbf{H_1} &= [\mathbf{H_{1,L}}, \mathbf{H_{1,R}}] \\ \mathbf{H_2} &= [\mathbf{H_{2,L}}, \mathbf{H_{2,R}}] \\ \mathbf{H_3} &= [\mathbf{H_{3,L}}, \mathbf{H_{3,R}}] \end{aligned} \tag{4.3}$$

where subscripts L, R identify respectively the left and right section of each circular area.

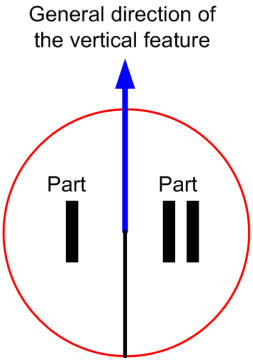


Figure 4.8: The two sections of a circular area.

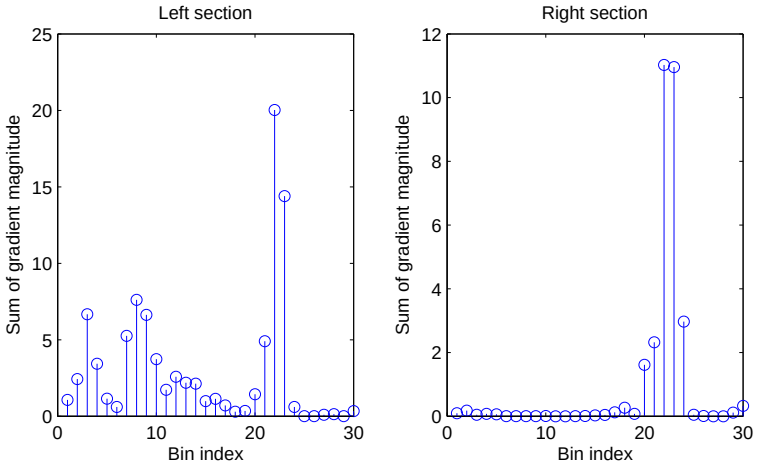


Figure 4.9: An example of gradient orientation histograms for the left and right sides of a circular area.

4.3.3 Building the feature Descriptor

From the computed orientation histograms, we build the final feature descriptor by stacking all three histogram pairs as follows:

$$\mathbf{H} = [\mathbf{H_1}, \mathbf{H_2}, \mathbf{H_3}] \tag{4.4}$$

To have slight illumination invariance, we pre-normalize each histogram $\mathbf{H_i}$ to have unit length. This choice relies on the hypothesis that the image intensity changes linearly with illumination. However, non-linear illumination changes can also occur due to camera saturation or due to illumination changes that affect 3D surfaces with different orientations by different amounts. These effects can cause a large change in relative magnitude for some gradients, but are less likely to affect the gradient orientations. Therefore, we reduce the influence of large gradient magnitudes by thresholding the values in each unit histogram vector so that each bin is no larger than 0.1, and then renormalizing to unit length. This means that matching the magnitudes for large gradients is no longer as important, and that the distribution of orientations has greater emphasis. The value of 0.1 was determined experimentally and will be justified in Section 4.6.

To resume, our descriptor is an N-element vector containing the gradient orientation histograms of the circular areas. In our setup, we extract 3 circular areas from each vertical feature and use 32 bins for each histogram; thus the length of the descriptor is

$$N = 3areas \cdot 2parts \cdot 32bins = 192 \tag{4.5}$$

Observe that all feature descriptors are the same length.

4.4 Feature matching

As every vertical feature has its own descriptor, its correspondent in consecutive images can be searched among the features with the closest descriptor. To this end, we need to define a dissimilarity measure (i.e. distance) between two descriptors.

In the literature, several measures have been proposed for the dissimilarity between two histograms $\mathbf{H} = \{h_i\}$ and $\mathbf{K} = \{k_i\}$. These measures can be divided into two categories. The *bin-by-bin* dissimilarity measures only compare contents of corresponding histogram bins, that is, they compare h_i and

k_i for all i, but not h_i and k_i for $i \neq j$. The *cross-bin* measures also contain terms that compare non-corresponding bins. Among the *bin-by-bin* dissimilarity measures, fall the Minkoski-form distance, the Jeffrey divergence, the χ^2 statistics, and the Bhattacharya distance. Among the *cross-bin* measures, one of the most used is the Quadratic-form distance. An exhaustive review of all these methods can be found in [73–75].

In our work, we tried the dissimilarity measures mentioned above but the best results were obtained using the L_2 distance (i.e. Euclidean distance) that is a particular case of the Minkoski-form distance. Therefore, in our experiments we used the Euclidean distance as a measure of the dissimilarity between descriptors, which is defined as:

$$d(\mathbf{H}, \mathbf{K}) = \sqrt{\sum_{i=1}^{N} |h_i - k_i|^2} \tag{4.6}$$

By definition of distance, the correspondent of a feature, in the observed image, is expected to be the one, in the consecutive image, with the minimum distance. However, if a feature is no longer present in the next image, there will be a closest feature anyway. For this reason, we defined three tests to decide whether a feature correspondent exists and which one the correspondent is. Before describing these tests, let us introduce some definitions.

Let $\{\mathbf{A_1}, \mathbf{A_2}, \ldots, \mathbf{A_{N_A}}\}$ and $\{\mathbf{B_1}, \mathbf{B_2}, \ldots, \mathbf{B_{N_B}}\}$ be two sets of feature descriptors extracted at time t_A and t_B respectively, where N_A, N_B are the number of features in the first and second image.
Then, let

$$D_i = \{d(\mathbf{A_i}, \mathbf{B_j}), j = 1, 2, \ldots, N_B)\} \tag{4.7}$$

be the set of all distances between a given $\mathbf{A_i}$ and all $\mathbf{B_j}$ ($j = 1, 2, \cdots, N_B$). Finally, let $minD_i = \min_i (D_i)$ be the minimum of the distances between given $\mathbf{A_i}$ and all $\mathbf{B_j}$.

4.4.1 First test

The first test checks that the distance from the closest descriptor is smaller than a given threshold, that is:

$$minD_i = F_1. \tag{4.8}$$

Table 4.1: The distances between the descriptor $\mathbf{A_1}$ at time t_A and all descriptors $\mathbf{B_j}$, $j = 1, 2, .., N_B$ at time t_B

B1	B2	B3	B4	B5	B6	B7
2.38	5.42	4.55	5.79	5.66	6.17	5.43

Table 4.2: The parameters used by our algorithm with their empirical values

$F_1 = 1.05$	$F_2 = 0.75$	$F_3 = 0.8$

By this criterion, we actually set a bound on the maximum acceptable distance to the closest descriptor.

4.4.2 Second test

The second test checks that the distance from the closest descriptor is smaller enough than the mean of the distances from all other descriptors, that is:

$$minD_i = F_2 \cdot < D_i > \tag{4.9}$$

where $< D_i >$ is the mean value of D_i and F_2 clearly ranges from 0 to 1. This criterion comes out of experimental results. In Table 4.1, we show an example of real comparison among the distances between descriptor $\mathbf{A_1}$ at time t_A and all descriptors $\mathbf{B_j}$ at time t_B. Observe that descriptor $\mathbf{B_1}$ is the correct correspondent of $\mathbf{A_1}$. Also note that its distance is smaller than the mean of all other distances.

4.4.3 Third test

Finally, the third test checks that the distance from the closest descriptor is smaller than the distance from the second closest descriptor:

$$minD_i = F_3 \cdot SecondSmallestDistance, \tag{4.10}$$

where F_3 clearly ranges from 0 to 1. As in the previous test, the third test raises from the observation that, if the correct correspondence exists, then there must be a big gap between the closest and the second closest descriptor.

Factors F_1, F_2, F_3 are to be determined experimentally. The empirical values used in our experiments are shown in Table 4.2. The choice of these values will be motivated in Section 4.6.

4.5 Comparison with other image similarity measures

A good method to evaluate the distinctiveness of the descriptors in the observed image is to compute a similarity matrix $\mathbf{S}$ where each element $\mathbf{S}(i,j)$ contains the distance between the ith and jth descriptor. That is,

$$\mathbf{S}(i,j) = d(\mathbf{H}_i, \mathbf{H}_j), \tag{4.11}$$

where $\mathbf{H}_i$ is the descriptor of the ith radial line and distance d is defined as in (4.6). Observe that to build this matrix we compute the the radial line's descriptor for every $\theta \in [0°, 360°]$. We used a θ increment of $1°$ and thus $i = 1, 2, \ldots, 360$. Furthermore, note that $\mathbf{S}$ is symmetric and that $\mathbf{S}(i,j) = 0$ for $i = j$. The similarity matrix computed for the image of Fig. 4.7 is shown in Fig. 4.11.

In this section, we want to compare our descriptor with other two image similarity measures that are very used in image registration but are also commonly used for matching individual lines, that is, Sum of Squared Differences (SSD) and Normalized Cross-Correlation (NCC) (their definitions can be found in [67]). When using SSD and NCC for comparing two patterns, the pattern descriptor can be seen as the pattern intensity. In our case, we take as a pattern the rectangular region around the observed radial line as shown in Fig. 4.10. As we did to build the similarity matrix for our descriptors, we compare given pattern $\mathbf{P}_i$ with pattern $\mathbf{P}_j$ using either SSD or NCC and build the respective similarity matrices, that is:

$$\mathbf{S}_{SSD}(i,j) = SSD(\mathbf{P}_i, \mathbf{P}_j), \tag{4.12}$$

$$\mathbf{S}_{NCC}(i,j) = NNC(\mathbf{P}_i, \mathbf{P}_j), \tag{4.13}$$

The two similarity matrices for the image in Fig. 4.7 is shown in Fig. 4.12 and 4.13. Concerning the size win of the patterns for computing SSD and NCC, we chose $win = 2r_a$. Observe that this choice is reasonable as $2r_a$ is also the size (diameter) of the three circular areas used to build our descriptor. Furthermore observe that, for SSD, maximum similarity between two patterns occurs when SSD=0; conversely, for NNC, maximum similarity (correlation) occurs when NCC=1 (this explains why the diagonal axis in Fig. 4.13 is white instead of black.

To interpret the similarity matrix, consider points along the diagonal axis in Fig. 4.11. Each point is perfectly similar to itself, so all the points on the

diagonal are dark. Starting from a given point on the diagonal, one can compare how its correspondent descriptor relates to its neighbors forward and backward by tracing horizontally or vertically on the matrix. To compare given descriptor $\mathbf{H}_i$ with descriptor $\mathbf{H}_{i+n}$, simply start at point (i, i) on the matrix and trace horizontally to the right to $(i, i + n)$.

In the similarity matrix for SSD, one can see large blocks of dark which indicate that there are repeating patterns in the image or that the patterns are poorly textured. Rectangular blocks of dark that occur off the diagonal axis indicate reoccurring patterns. This can be better understood by observing Fig. 4.10. As can be seen, there are poorly textured objects and repeating structure.

Similar comments can be done regarding the similarity matrix for NCC, but we have to invert word "dark" with "light", due to the inverse definition of NCC. However, observe that the behavior of NCC is better than SSD: first, the size of the blocks along or off the diagonal axis is smaller; then, points on the diagonal are much lighter than points off the diagonal.

Regarding the similarity matrix of our descriptor, the diagonal axis is well demarcated, in fact points on the diagonal are much darker than those off the diagonal; the contrast with the regions off the diagonal is higher than NCC. Finally, observe that blocks along or off the diagonal axis are much smaller or lighter than SSD and NCC; this indicates that even on poorly textured surfaces our descriptor is distinctive.

In the concept, our method is similar to SIFT [76], which also uses gradient histograms to build distinctive descriptors of image keypoints.

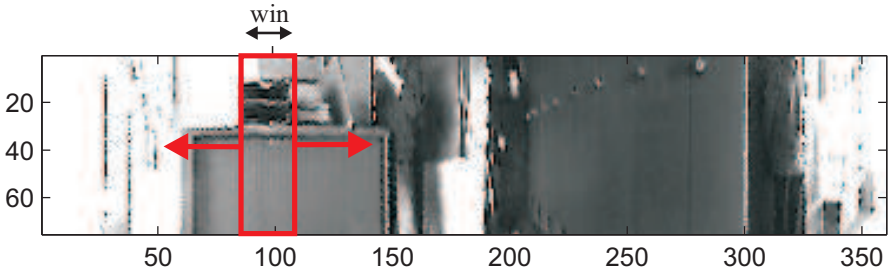


Figure 4.10: This is the same image of Fig. 4.7 after unwrapping into a cylindrical panorama. The rectangular region used to compute SSD and NCC is also shown.

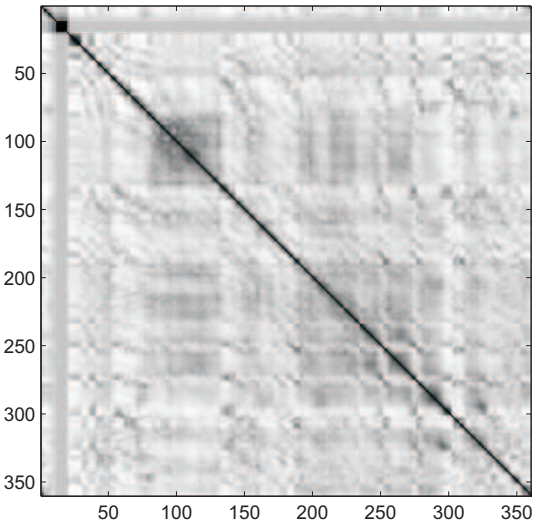


Figure 4.11: Similarity matrix for descriptors.

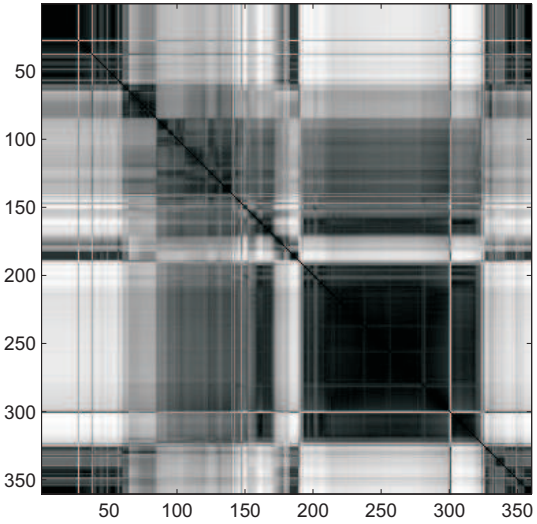


Figure 4.12: Similarity matrix for SSD.

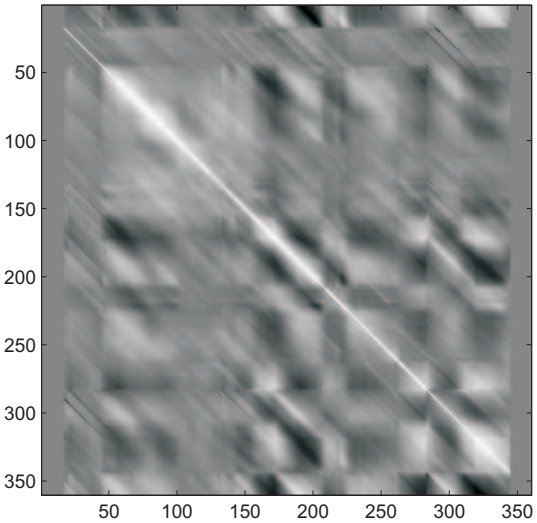


Figure 4.13: Similarity matrix for NCC.

4.6 Performance Evaluation

In this section, we characterize the performance of our descriptor on a large image dataset by taking into account the sensitiveness to different parameters, which are image saturation, pixel noise, number of histogram bins, and use of overlapping circular areas. Furthermore, we also motivate the choice of the empirical values for F_1, F_2, and F_3, which are shown in Table 4.2.

Ground truth

To generate the ground truth for testing our descriptor, we used a database of 850 omnidirectional pictures that is a subset of the whole video sequence used in Section 4.7. About 10 verticals were extracted in average from each image; then we matched each feature individually against the whole database using the matching method of the previous section. To insure that matching was correct, we visually inspected every single correspondence individually. Correspondent features were labeled with the same ID. The images were taken from the hallway of our department. Figure 4.21 shows six sample images from our dataset. The images show that the illumination conditions vary strongly. Due to big windows, a mixture of natural and artificial lighting produces difficult lighting conditions like highlights and specularities. With regard to the viewpoint change, the pictures of our dataset were taken such that each vertical line could be continuously observed for at least 2 meters of motion.

Image saturation

As we mentioned in Section 4.3.3, we threshold the values of the histogram vectors to reduce the influence of image saturation. The percent of correct matches for different threshold values is shown in Fig. 4.14. The results show the percent of verticals that find a correct match to the single closest neighbor among the whole database. As the graph shows, the maximum percent of correct matches is reached when using a threshold value of 0.1. In the remainder of this paper, we will always use this value.

Image noise

The percent of correct matches for different amounts of gaussian image noise (from 0% to 10%) is shown in Fig. 4.15. Again, the results show the percent of correct matches found using the single nearest neighbor among the

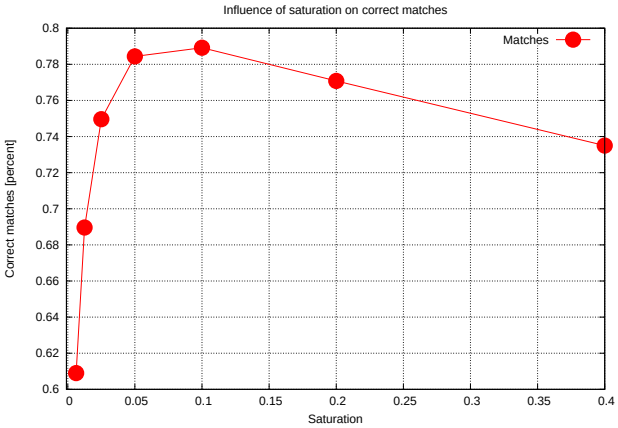


Figure 4.14: Influence of saturation on correct matches.

all database. As this graph shows, the descriptor is resistant even to large amount of pixel noise.

Histogram bins and circular areas

There are two parameters that can be used to vary the complexity of our descriptor: the number of orientation bins, N_b, in the histograms, and the number of circular areas. Although in the explanation of the descriptor we used 3 non overlapping circular areas, we evaluated the effect of using 5 overlapping areas with 50% overlap between two circles. The results are shown in Fig. 4.16. As the graph shows, there is a slight improvement in using 5 overlapping areas (the amelioration is only 1%). Also, the performance is quite similar using 8, 16, or 32 orientation bins (we used powers of 2 because they can be computed more efficiently). Following this considerations, the best choice would seem to use 3 areas and 8 histograms bins in order to reduce the dimension of the descriptor. Conversely, as in this graph the percent of correct matches is found only using the nearest closest descriptor, we observed that the best matching results, when using the rules of Section 4.4, are obtained with 32 orientations. Thus, in our implementation we used 3 areas and 32 histogram bins.

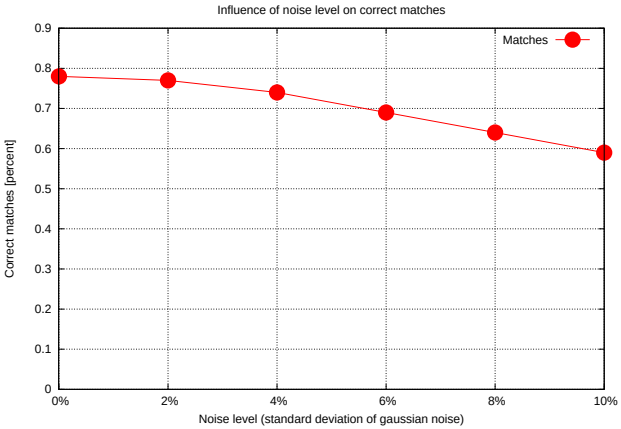


Figure 4.15: Influence of noise level (%) on correct matches. The correct matches are found using only the nearest descriptor in the database.

Matching rules

Figure 4.17 shows the Probability Density Function (PDF) for correct and incorrect matches in terms of the distance to the closest neighbor of each keypoint. In our implementation of the first rule, we chose $F_1 = 1.05$. As observed in the graph, by this choice we reject all matches in which the distance to the closest neighbor is greater than 1.05, which eliminates 50% of the false matches while discarding less than 5% of correct matches.

Similarly, Fig. 4.18 shows the PDFs in the terms of the ratio of closest to average-closest neighbor of each keypoint. In our implementation of the second rule, we chose $F_2 = 0.75$. As observed in the graph, by this choice we reject all matches where the ratio between the closest neighbor distance and the mean of all other distances is greater than 0.75, which eliminates 45% of the false matches while discarding less than 8% of correct matches.

Finally, Fig. 4.19 shows the PDFs in terms of the ratio of closest to second-closest neighbor of each keypoint. In our implementation of the third rule, we chose $F_3 = 0.8$; in this way we reject all matches in which the distance ratio is greater than 0.8, which eliminates 92% of the false matches while discarding less than 10% of correct matches.

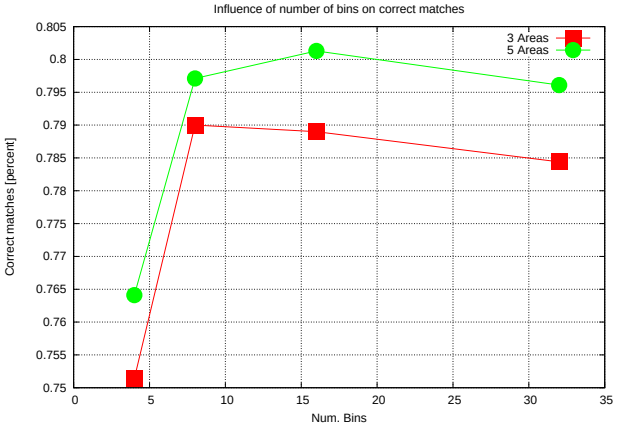


Figure 4.16: Influence of number of bins on correct matches.

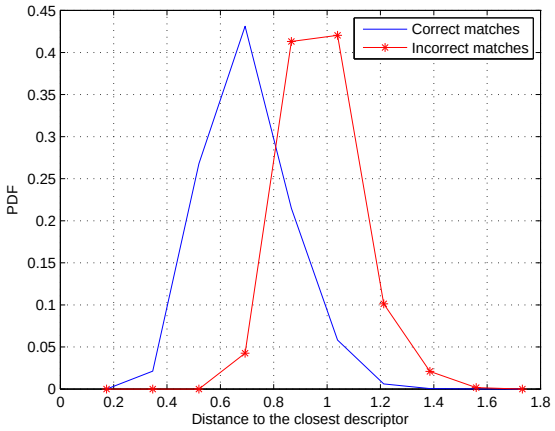


Figure 4.17: The probability density function that a match is correct according to the first rule.

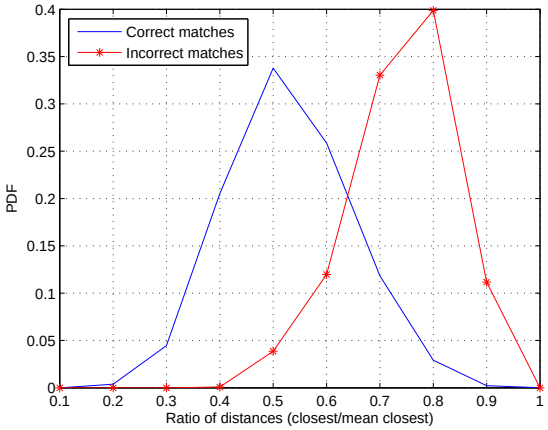


Figure 4.18: The probability density function that a match is correct according to the second rule.

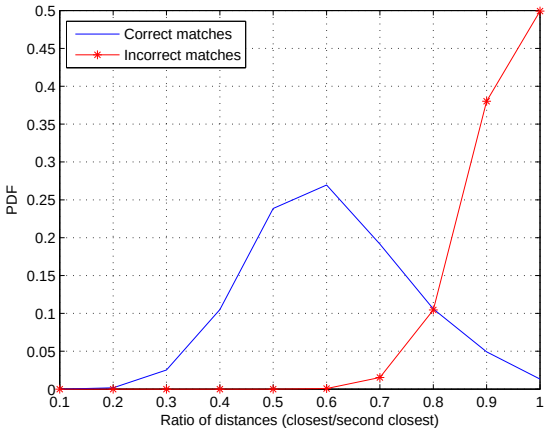


Figure 4.19: The probability density function that a match is correct according to the third rule.

Figure 4.20: Floorplan of the institute showing the robot path (in red).

4.7 Results

In our experiments, we adopted a mobile robot with a differential drive system equipped with encoder sensors on the wheels. Furthermore, we equipped the robot with an omnidirectional camera (the same used also in Section 3.3.2) consisting of a KAIDAN 360° One VR hyperbolic mirror and a SONY CCD camera. We set the image resolution to 640×480 pixels. A picture of the all settings is depicted in Fig. 5.4. In this section, we show the performance of our feature extraction and matching method by capturing pictures from our robot in a real indoor environment.

The robot was moving at about 0.15 m/s and was acquiring frames at 3 Hz, meaning that during straight paths the traveled distance between two consecutive frames was 5 cm. The robot was moved in the hallway of our institute along the red path shown in Fig. 4.20. 1852 frames were extracted during the whole path. Figure 4.21 shows six sample images from the dataset.

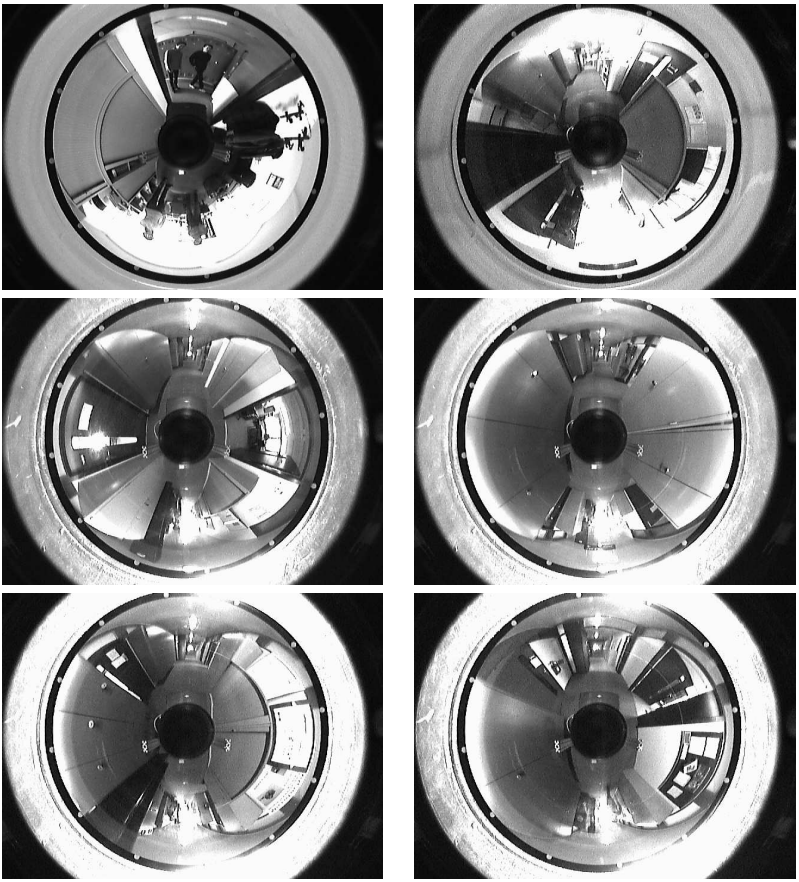


Figure 4.21: Omnidirectional images taken at different locations.

The images show that the illuminations conditions vary strongly. Due to big windows, a mixture of natural and artificial lighting produces difficult lighting conditions like highlights and specularities.

The result of feature tracking is shown only for the first 150 frames in Fig. 4.22. This result was obtained using only the three matching rules described in Sections 4.4.1, 4.4.2, 4.4.3. No other constraint, like mutual and topological relations, has been used. This plot refers to a short path of the whole trajectory while the robot was moving straight (between frame no. 0 and

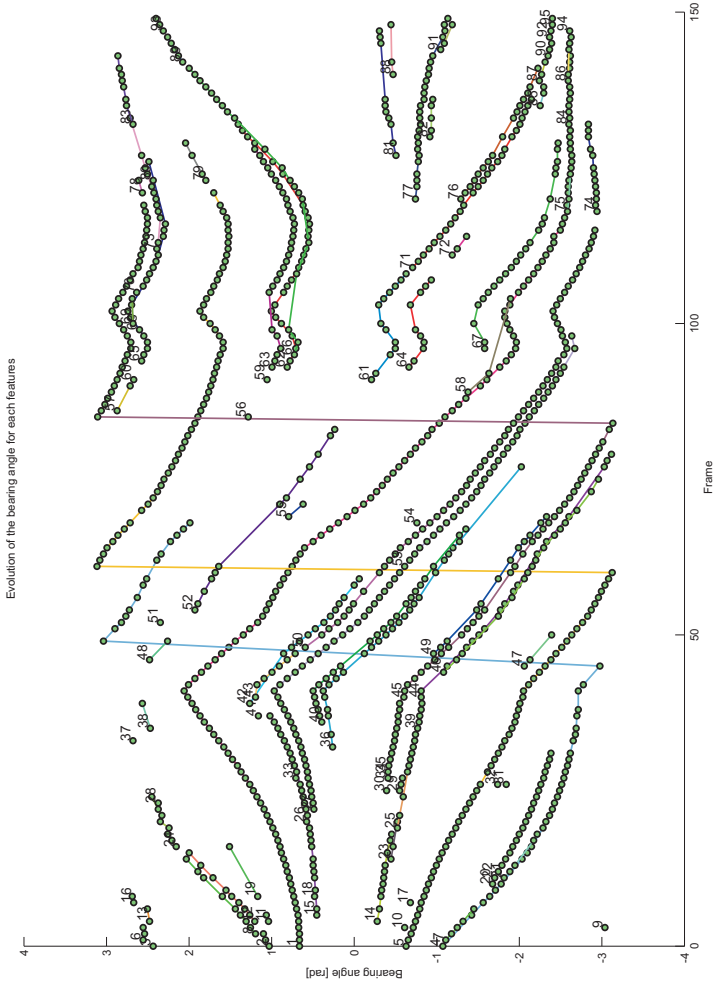


Figure 4.22: Feature tracking during the motion of the robot. In y-axis is the angle of sight of each feature and in the x-axis the frame number. Each circle represents a feature detected in the observed frame. Lines represent tracked features. Numbers appear only when a new feature is detected.

46), then doing a 180° rotation (between frame no. 46 and 106), and moving straight again. As observed, most of the features are correctly tracked over the time. Indeed, most of the lines appear smooth and homogeneous. The

lines are used to connect features that belong to the same track. When a new feature is detected, this feature is given a label with progressive numbering and a new line (i.e. track) starts from it. In this graph, there are three false matches that occur at the points where two tracks intersect (e.g. at the intersection between tracks no. 1 and 58, between track no. 84 and 86, and between track no. 65 and 69). Observe that the three huge jumps in the graph are not false matches; they are only due to the angle transition from $-\pi$ to π.

Observe that our method was able to match features even when their correspondents were not found in the previous frames. This can be seen by observing that sometimes circles are missing on the tracks (look for instance at track no. 52). When a correspondence is not found in the previous frame, we start looking into all previous frames (actually up to twenty frames back) and stop when the correspondence is found.

By examining the graph, can be seen that some tracks are suddenly given different numbers. For instance, observe that feature no. 1 - that is the first detected feature that starts at frame no. 0 - is correctly tracked until frame no. 120 and is then labeled as feature no. 75. This is because at this frame no correspondence was found and then the feature was labeled as a new entry (but in fact is a false new entry). Another example is feature no. 15 that is then labeled as no. 18 and no. 26. By a careful visual inspection, one can find only a few other examples of false new entries. Indeed, tracks that at a first glance seem to be given different numbers, belong in fact to other features that are very close to the observed one.

After visually inspecting every single frame of the whole video sequence (composed of 1852 frames), we found 37 false matches and 98 false new entries. Comparing these errors to the 7408 corresponding pairs detected by the algorithm over the whole video sequence, we had 1.8% of mismatches. Furthermore, we found that false matches occurred every time the camera was facing objects with repetitive texture (like in Fig 4.10 or in the fourth image of Fig. 4.21). Thus, ambiguity was caused by the presence of vertical elements which repeat almost identical in the same image. On the other hand, a few false new entries occurred when the displacement of the robot between two successive images was too large. However, observe that when a feature matches with no other feature in previous frames, it is better to believe this feature to be new rather than commit a false matching. The evolution of the recognition rate during the whole video sequence is shown in Table 4.3.

As we already mentioned above, the results reported in this section were obtained using only the three matching rules described in Sections 4.4.1, 4.4.2, 4.4.3. Obviously, the performance of tracking could be further improved by adding other constraints like mutual and topological relations among features.

Table 4.3: Recognition rate

Frame interval	Number of matches	Rate of correct matches (%)	Rate of false matches (%)	Rate of false new entries (%)
0-200	735	97.48	0.53	1.98
200-400	972	98.58	0.20	1.22
400-600	823	98.68	0.35	0.96
600-800	857	97.83	0.80	1.37
800-1000	685	98.13	0.57	1.29
1000-1200	740	98.40	0.26	1.33
1200-1400	906	98.26	0.43	1.30
1400-1600	784	97.75	0.62	1.62
1600-1852	771	98.34	0.76	1.89

4.8 Conclusion

In this chapter, we presented a robust method for matching vertical lines among omnidirectional images. The basic idea to achieve robust feature matching consists in creating a descriptor which is unique and distinctive for each feature. Furthermore, this descriptor is invariant to rotation and slight changes of illumination.

We characterized the performance of the descriptor on a large image dataset by taking into account the sensitiveness to the different parameters of the descriptor. The robustness of the approach was also validated through a real navigation experiment with a mobile robot equipped with an omnidirectional camera. The performance of tracking was very good as many features were correctly detected and tracked over long time. Furthermore, because the results were obtained using only the three matching rules described in Section 4.4, we expect that the performance would be notably improved by adding other constraints like mutual and topological relations among features.

The visual tracking procedure described in this chapter will be used in Chapter 5 to automatically and extrinsically calibrate a camera with the reference system of a mobile robot. An example of vision based robot motion estimation will be also given.

Chapter 5

Calibration between camera and odometry

This chapter presents a new technique to determine the pose of an omnidirectional camera with respect to the robot's reference system. The proposed technique takes advantage of an extended Kalman filter which fuses the encoder readings with the image coordinates of one or more features tracked during the motion. The novelty of the technique is that the two sensors are self-calibrated while the robot is moving. Furthermore, no a priori knowledge about the environment is required. The strategy is theoretically validated through an observability analysis which takes into account the system nonlinearities. This analysis shows that the system contains all the necessary information to perform self-calibration. The performance of the proposed method is evaluated through several simulations and experiments performed on a real robot equipped with encoder sensors and an omnidirectional camera.

5.1 Introduction

WHEN a mobile robot is equipped with a vision sensor, it is sometimes required to determine accurately the extrinsic parameters that establish the pose of the camera with respect to the robot's reference system (i.e. the odometry system). The estimation of the extrinsic parameters has to be very accurate for many applications. For example, in the frame-work

of multi-robot localization, where the bearing observations among the team members are very informative [77, 78], an error of $1cm$ in estimating the position of the vision sensor with respect to the robot's frame would produce a bearing error of 0.2 deg if the distance between the robots is $3m$.

In this chapter, we present a completely new strategy based on the Extended Kalman Filter (EKF) to automatically perform the estimation of the extrinsic parameters while the robot is moving. Although in this thesis we deal with omnidirectional cameras, more in general our strategy can be adopted to calibrate any robot bearing sensor. We call this strategy "self-calibration" because we use no prior knowledge about the environment and because the calibration is done while the robot is moving. The only prerequisite is the capability of robustly and visually tracking one or more features in the space. In our experiments, we will use the feature extraction and matching method that has been described in Chapter 4.

This chapter is organized as follows. In Section 5.2, we define the problem, while the strategy to perform the self-calibration is introduced in Section 5.3. To simplify the explanation, we will first describe the method for the case of tracking a single feature and then we will extend the method to the general case of multiple features (5.6). Finally, the proposed method will be theoretically validated through an observability analysis which takes into account the system nonlinearities (Section 5.4) and experimentally validated through real experiments and accurate simulations (Section 5.5).

5.2 The problem

In order to simplify the problem, we do the following assumptions; in particular, we assume that the robot is moving in a flat environment and that it is equipped with an omnidirectional camera whose z-axis is parallel to the z-axis of the robot, that is, the mirror axis is perpendicular to the floor (the same assumptions were done also in Chapter 4). Furthermore, we assume that the omnidirectional camera is already intrinsically calibrated (as described in Chapter 3). According to this, the three-dimensional camera-odometry calibration problem becomes a two-dimensional problem.

The first goal is the estimation of the three parameters ϕ, ρ, and ψ which characterize the transformation between the two references frames attached respectively on the robot and on the camera (see Fig. 5.1).
The second goal is to perform the calibration automatically and during the

motion of the robot.
The available data are the robot wheels displacements delivered by the encoder sensors and the bearing angles of several features in the camera reference frame (β in Fig. 5.1).
From now on, we will explain the method for the case of one single feature. In Section 5.6, we will extend the equations to the case of tracking multiple features.

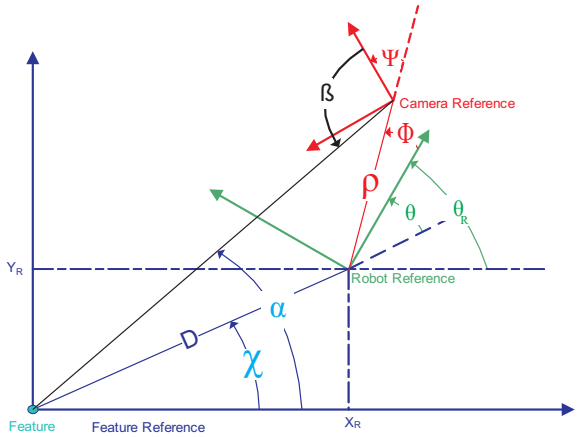


Figure 5.1: The two reference frames respectively attached to the robot and to the camera. All parameters used in this chapter are also indicated.

As we consider the case of a mobile robot moving in a 2D environment, its configuration is described through the state $\mathbf{X_R} = [x_R, y_R, \theta_R]^T$ containing its position and orientation (as indicated in Fig. 5.1). Furthermore, we consider the case of a robot equipped with a differential drive system. The robot configuration $\mathbf{X_R}$ can then be estimated by integrating the encoder data. In particular, we have:

$$\begin{cases} x_{R_{i+1}} &= x_{R_i} + \delta\rho_i cos\left(\theta_{R_i} + \frac{\delta\theta_i}{2}\right) \\ y_{R_{i+1}} &= y_{R_i} + \delta\rho_i sin\left(\theta_{R_i} + \frac{\delta\theta_i}{2}\right) \\ \theta_{R_{i+1}} &= \theta_{R_i} + \delta\theta_i \end{cases} , \tag{5.1}$$

where the quantities $\delta\rho$ and $\delta\theta$ are related to the displacements $\delta\rho_R$ and $\delta\rho_L$ (respectively of the right and left wheel) directly provided by the encoders through:

$$\delta\rho = \frac{\delta\rho_R + \delta\rho_L}{2} \quad , \quad \delta\theta = \frac{\delta\rho_R - \delta\rho_L}{b} \tag{5.2}$$

where b is the distance between the wheels.

Regarding the bearing angle β we obtain the following analytical expression (see Fig. 5.1):

$$\beta = \pi - \psi - \theta_R - \phi + \alpha \tag{5.3}$$

with

$$\alpha = \tan^{-1}\left(\frac{y_R + \rho sin(\theta_R + \phi)}{x_R + \rho cos(\theta_R + \phi)}\right) \tag{5.4}$$

5.3 EKF based calibration

An intuitive procedure to determine the three parameters ϕ, ρ, and ψ could be to use the data from the encoders to estimate the robot configuration (provided that the initial robot configuration is known). Then, by measuring the bearing angle β at several different robot configurations (at least three) would be possible to obtain the parameters ϕ, ρ, and ψ by solving a non linear system in three unknowns. However, the drawback of this method is that, when the robot configuration is estimated by using only the encoder data, the error integrates over the path. This means that this procedure can be applied only for short paths and therefore the achievable accuracy on the estimation of ϕ, ρ, and ψ is limited. Furthermore, the initial robot configuration has to be known with high accuracy.

One way to overcome these problems is to integrate the encoder data with the bearing angle measurements to estimate the robot configuration. This can be performed by introducing an augmented state $\mathbf{X_a}$ containing the robot configuration and the calibration parameters ϕ, ρ, and ψ:

$$\mathbf{X_a} = [x_R, y_R, \theta_R, \phi, \rho, \psi]^T \tag{5.5}$$

An EKF can be adopted to estimate the state $\mathbf{X_a}$. The inputs $\mathbf{u}$ of the dynamics of this state are directly provided by the encoder data and the observations $\mathbf{z}$ are the bearing angles provided by the vision sensor. According to this, we can write:

$$\mathbf{X}_{\mathbf{a}_{i+1}} = f\left(\mathbf{X}_{\mathbf{a}_i}, \mathbf{u}_i\right) \tag{5.6}$$

$$z_i = h\left(\mathbf{X}_i\right) + w_i, \tag{5.7}$$

where:

$$\mathbf{u} = [\delta\rho_R, \delta\rho_L]^T; \tag{5.8}$$

$$f\left(\mathbf{X}_{\mathbf{a}_{i+1}}, \mathbf{u}_i\right) = \begin{bmatrix} x_{R_i} + \delta\rho_i cos\left(\theta_{R_i} + \frac{\delta\theta_i}{2}\right) \\ y_{R_i} + \delta\rho_i sin\left(\theta_{R_i} + \frac{\delta\theta_i}{2}\right) \\ \theta_{R_i} + \delta\theta_i \\ \phi_i \\ \rho_i \\ \psi_i \end{bmatrix} \tag{5.9}$$

$$h\left(\mathbf{X}_i\right) = \tan^{-1}\left(\frac{y_R + \rho sin(\theta_R + \phi)}{x_R + \rho cos(\theta_R + \phi)}\right) + \pi - \psi - \theta_R - \phi; \tag{5.10}$$

Note that function h in (5.10) is the function describing the dependency of the bearing angle β on the state $\mathbf{X_a}$ and is given by expressions (5.3) and (5.4). w is a stochastic quantity representing the measurement error which is assumed to be zero mean, with a Gaussian distribution and independent of the encoder errors. Furthermore, $< w_i\ w_j >= 0$ when $i \neq j$.

The previous equations, along with an odometry statistical error model (5.25), allow to implement an EKF to estimate $\mathbf{X_a}$. However, before implementing this filter, it is desirable to check if the system contains all the necessary information to perform the estimation with an error which is bounded. To answer this question, we have to carry out an observability analysis. Indeed, when a system is observable, it contains all the necessary information to perform the estimation with an error which is bounded [79]. The value of

this bound obviously depends on the accuracy of the sensors.

In the next section, we will perform this observability analysis (by taking into account the system non-linearities) and we will show that actually the state $\mathbf{X_a}$ is not observable. On the other hand, the state $\mathbf{X_a}$ contains the robot configuration whose estimation is not our goal, indeed we just want to estimate the three parameters ϕ, ρ and ψ. In Section 5.4, we will show that by introducing a new state $\mathbf{X}$ as in (5.14) the system becomes observable. This will be possible by avoiding the dependency of both the state and the observation on the full robot configuration. To avoid this dependency observe that, by manipulating (5.3) and (5.4), we can rewrite the observation β as:

$$\beta = \tan^{-1}\left(\frac{-\rho sin(\theta+\phi)}{-D-\rho\ cos(\theta+\phi)}\right) - \theta - \phi - \psi \tag{5.11}$$

where

$$\theta = \theta_R - tan^{-1}\left(\frac{y_R}{x_R}\right) \tag{5.12}$$

and

$$D = \sqrt{x_R^2 + y_R^2} \tag{5.13}$$

In Section 5.4, we will show that if instead of the state in (5.5) we consider the state

$$\mathbf{X} = [D, \theta, \phi, \rho, \psi]^T, \tag{5.14}$$

the system is observable. Therefore, in the remainder of this section we implement an EKF which estimates the state $\mathbf{X}$.

5.3.1 Implementing the EKF

From Equations (5.1), (5.12), (5.13), we obtain the following dynamics for the state $\mathbf{X}$ (which relates the state to the encoder data):

$$\begin{cases} D_{i+1} &= D_i + \delta\rho_i cos\theta_i \\ \theta_{i+1} &= \theta_i + \delta\theta_i - \frac{\delta\rho_i}{D_i} sin\theta_i \\ \phi_{i+1} &= \phi_i \\ \rho_{i+1} &= \rho_i \\ \psi_{i+1} &= \psi_i \end{cases} \tag{5.15}$$

The EKF estimates the state $\mathbf{X}$ by fusing the information coming from the encoder data and the bearing angle observations. To implement the standard equations of this filter we need to compute the two Jacobians $\mathbf{F_x}$ and $\mathbf{F_u}$ of the dynamics in (5.15) with respect to the state $\mathbf{X}$ and with respect to encoder readings ($\delta\rho_R$ and $\delta\rho_L$). Finally, we need to compute the Jacobian $\mathbf{H}$ of the observation function in (5.11) with respect to $\mathbf{X}$. These matrices are required to implement the EKF [80]. By a direct computation we obtain:

$$F_x = \begin{bmatrix} 1 & -\delta\rho\ sin\theta & 0 & 0 & 0 \\ \frac{\delta\rho}{D^2}\ sin\theta & 1 - \frac{\delta\rho}{D} cos\theta & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.16}$$

$$F_u = \begin{bmatrix} \frac{cos\theta}{2} & \frac{cos\theta}{2} \\ \frac{1}{b} - \frac{sin\theta}{2D} & -\frac{1}{b} - \frac{sin\theta}{2D} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{5.17}$$

and

$$H = \tag{5.18}$$

$$= \left[\frac{-\rho sin(\theta+\phi)}{D^2 + 2\rho D cos(\theta+\phi) + \rho^2}, \ \frac{-D\rho\ cos(\theta+\phi) - D^2}{D^2 + 2\rho D cos(\theta+\phi) + \rho^2}, \right.$$

$$\left. \frac{-D\rho\ cos(\theta+\phi) - D^2}{D^2 + 2\rho D cos(\theta+\phi) + \rho^2}, \ \frac{D sin(\theta+\phi)}{D^2 + 2\rho D cos(\theta+\phi) + \rho^2}, \ -1 \right]$$

5.4 Observability analysis

In control theory, a system is defined observable when it is possible to recover its initial state by knowing, in a given time interval, the control inputs and the outputs [79]. The observability property has a very practical meaning. When a system is observable, it contains all the necessary information to perform the estimation with an error which is bounded [79].

This section consists of two subsections. In the former we perform the observability analysis for the system described by the state $\mathbf{X_a}$. In this case we will show that the state is not observable. In the latter, we consider the state $\mathbf{X}$ in (5.14) and we will show that it is observable. In both cases, our analysis takes into account the system non-linearities. Indeed, the observability analysis changes dramatically from linear to non-linear systems [79]. First of all, in the non-linear case the observability is a local property. For this reason, in the non-linear case the concept of the *local distinguishability property* was introduced by Hermann and Krener [81]. The same authors introduced also a criterion, *the observability rank condition*, to verify if a system has this property. This criterion plays a very important role since in many cases a non-linear system, whose associated linearized system is not observable, has however the local distinguishability property. Regarding the localization problem, this was proved in [82] and [83]. Note that it is the distinguishability property which implies that the system contains the necessary information to have a bounded estimation error (actually, provided that the locality is large enough with respect to the sensor accuracy).

We now want to remind some concepts in the theory by Hermann and Krener in [81]. We will adopt the following notation. We indicate the K^{th} order Lie derivative of a field Λ along the vector fields $v_{i_1}, v_{i_2}, ..., v_{i_K}$ with $L^K_{v_{i_1},v_{i_2},...,v_{i_K}}\Lambda$. Note that the Lie derivative is not commutative. In particular, in $L^K_{v_{i_1},v_{i_2},...,v_{i_K}}\Lambda$ it is assumed to differentiate along v_{i_1} first and along v_{i_K} at the end.

Let us indicate with Ω the space spanned by all the Lie derivatives $L^K_{f_{i_1},f_{i_2},...,f_{i_K}} h(X)|_{t=0}$ $(i_1, ..., i_K = 1, 2, ..., M$ and the functions f_{i_j} are defined in (5.22)).
Furthermore, we denote with $d\Omega$ the space spanned by the gradients of the generators of Ω.
In this notation, the observability rank condition can be expressed in the following way: *The dimension of the observable sub-system at a given X_0 is*

equal to the dimension of $d\Omega$.

5.4.1 Observability analysis for the state $\mathbf{X_a}$

The dynamics of our system is described through the following equations:

$$\begin{cases} \dot{x}_R = v\ cos\theta_R \\ \dot{y}_R = v\ sin\theta_R \\ \dot{\theta}_R = \omega \\ \dot{\phi} = \dot{\rho} = \dot{\psi} = 0 \end{cases} \tag{5.19}$$

Our system is affine in the input variables, i.e. the previous equations can be written in the following compact form:

$$\dot{\mathbf{X}}_\mathbf{a} \ = \ f(\mathbf{X_a}, \mathbf{u}) = \sum_{k=1}^{M} f_k(\mathbf{X_a})\mathbf{u}_k \tag{5.20}$$

where M is the number of the input controls (which are independent). In our case $M = 2$ and the controls are v and ω. Since these controls are linearly related to the true controls, which are the wheels velocities, for our analysis we can use v and ω as the controls.

We found that the computation becomes significantly easier if, for the robot position, we adopt the polar coordinates instead of the cartesian ones. In these coordinates, the robot configuration is $\mathbf{R} = [D, \chi, \theta_R]^T$ with $x_R = D\ cos\chi$ and $y_R = D\ sin\chi$.

Note that $\chi = \theta_R - \theta$ (see Fig. 5.1). The dynamics defined in (5.19) becomes:

$$\begin{cases} \dot{D} = v\ cos(\theta_R - \chi) \\ \dot{\chi} = \frac{v}{D}\ sin(\theta_R - \chi) \\ \dot{\theta}_R = \omega \\ \dot{\phi} = \dot{\rho} = \dot{\psi} = 0 \end{cases} \tag{5.21}$$

By identifying it with (5.20), we have $u_1 = v$, $u_2 = \omega$ and

$$f_1 = \left[cos\ \theta, \frac{sin\theta}{D}, 0, 0, 0, 0\right]^T, \quad f_2 = [0, 0, 1, 0, 0, 0]^T \tag{5.22}$$

The observation is defined by Equations (5.3) and (5.4) or by Equation (5.11). We remark that this second expression depends on χ and θ_R only through the difference $\theta = \theta_R - \chi$. Since also the two vector fields f_1 and f_2 depend on χ and θ_R only through θ, all the elements in $d\Omega$ have the structure:

$$w = [a_1, a_2, a_3, a_4, a_5, a_6]^T \quad with \quad a_3 = -a_2$$

Therefore, the dimension of $d\Omega$ cannot be larger than 5 and the system is not observable.

5.4.2 Observability analysis for the state X

The dynamics of our system is described through the following equations:

$$\begin{cases} \dot{D} = v\ cos\theta \\ \dot{\theta} = \omega - \frac{v}{D} sin\theta \\ \dot{\phi} = \dot{\rho} = \dot{\psi} = 0 \end{cases} \tag{5.23}$$

As in the previous case, we have the two independent input controls $u_1 = v$ and $u_2 = \omega$. By adopting the compact notation as in (5.20) we have:

$$f_1 = \left[cos\ \theta, -\frac{sin\theta}{D}, 0, 0, 0\right]^T, \quad f_2 = [0, 1, 0, 0, 0]^T \tag{5.24}$$

.
The observation is defined by Equation (5.11).

In order to prove that the state X is observable, it is sufficient to extract 5 independent vectors from the space $d\Omega$. In the Appendix, we compute the following Lie derivatives of the observation function: $L^0\beta$, $L^1_{f_1}\beta$, $L^1_{f_2}\beta$, $L^2_{f_1 f_2}\beta$, $L^2_{f_2 f_2}\beta$. Then, we prove that the correspondent gradients are independent. Therefore, the dimension of $d\Omega$ is 5 and the system is observable.

5.5 Results

In this section we present the results obtained through simulations (Section 5.5.1) and real experiments (Section 5.5.2) performed in order to validate the strategy introduced in Section 5.3.

5.5.1 Simulations

We simulated a mobile robot with a differential drive system equipped with encoder sensors and a vision system able to track a single feature (e.g. a vertical line) in the environment.

The adopted model to characterize the odometry error is that proposed by Chong and Kleeman [84]. Accordingly to this model, the translations of the right/left wheel as estimated by the encoder sensors are Gaussian random variables satisfying the following relation:

$$\delta\rho_{R/L} = \delta\rho^{a}{}_{R/L} + \nu_{R/L} \qquad \nu_{R/L} \sim N(0, K|\delta\rho_{R/L}|) \tag{5.25}$$

In other words, both $\delta\rho_R$ and $\delta\rho_L$ are assumed Gaussian random variables whose mean values are given by the actual values (respectively, $\delta\rho^{a}{}_{R}$ and $\delta\rho^{a}{}_{L}$) and whose variances increase linearly with the travelled distance. Furthermore, it is assumed that $\delta\rho_R$ and $\delta\rho_L$ are uncorrelated. With respect to the Chong-Kleeman model, we do not consider systematic errors (i.e. we assumed an odometry system perfectly calibrated).

In our simulations, we generated the encoder errors $\nu_{R/L}$ accordingly to the Gaussian distribution in (5.25). We set the parameter $K = 10^{-6}m$ accordingly with the values obtained in previous experiments [84,85]. The data from the encoders are delivered at a frequency equal to 100 Hz. The speed of the robot (v_R) is set equal to 0.2 ms^{-1}.

Regarding the vision sensor, we simulated a sensor able to return the bearing angle β with a Gaussian error. In particular, the variance is assumed to be $\sigma_\beta^2 = (1\ deg)^2$, accordingly with experimental results obtained with an omnidirectional camera [6]. The data from this sensor are delivered at 10 Hz.

We set the three parameters characterizing the robot vision sensor reference transformation equal to the following values: $\phi = \psi = 30\ deg$ and

$\rho = 0.1\ m$.

We simulated two kinds of robot trajectories. In both cases the initial robot configuration is $[2,\ 0,\ \pi/2]^T$ and the feature to track is at the origin. The first kind of trajectory is random and one example is shown in Fig. 5.2(a). In this case the robot moves for $1000s$. Each $dt = 0.01s$ the robot is moved by generating randomly the translation of the right and the left wheel. In particular, each translation is generated as a random value, whose mean value is equal to $v_R \cdot dt$ and whose variance is equal to $0.01 \cdot v_R \cdot dt$. Furthermore, the distance between the wheel is set equal to $0.25\ m$.

The second trajectory is shown in Fig. 5.3(a). This trajectory consists of pure rotations and pure translations. The length of each segment is about $1\ m$ while each rotation is approximately $450\ deg$. The robot moves for $100s$.

Figures 5.2(b), 5.2(c) and 5.2(d) refer to the trajectory shown in Fig. 5.2(a). They show the results obtained by implementing the EKF introduced in Section 5.3 to estimate respectively ρ, ϕ and ψ. The filter is initialized by setting the following values for the three parameters: $\phi = \psi = 0\ deg$ and $\rho = 0\ m$. It is possible to see that all the parameters converge to the actual values. However, the error is still of the order of $2\ deg$ for the two angles and $1\ cm$ for ρ after $200m$ of navigation. We obtained similar results by moving the robot along other random trajectories.

Figures 5.3(b), 5.3(c) and 5.3(d) refer to the trajectory shown in Fig. 5.3(a). In this case the convergence is much faster. In particular, only after $4m$ of navigation, the error is of the order of $0.1\ deg$ for the two angles and $0.1\ cm$ for ρ. We performed many simulations and we found that by moving the robot along this trajectory, the convergence is much faster than by using other trajectories. We remark that this trajectory is not only much more preferable with respect to the other ones because of the fast convergence, but it is also experimentally feasible because the robot navigates close enough to the feature making its detection easy and with high accuracy [6].

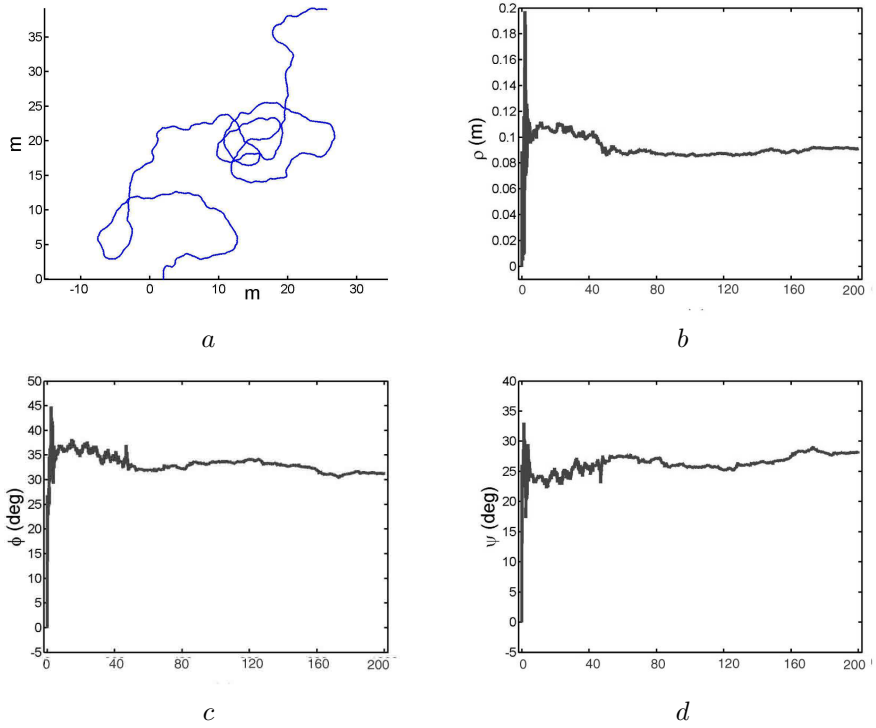


Figure 5.2: Results obtained by implementing the strategy introduced in Section 5.3 to estimate the three parameters ρ (b), ϕ (c) and ψ (d). The considered robot trajectory is displayed in (a).

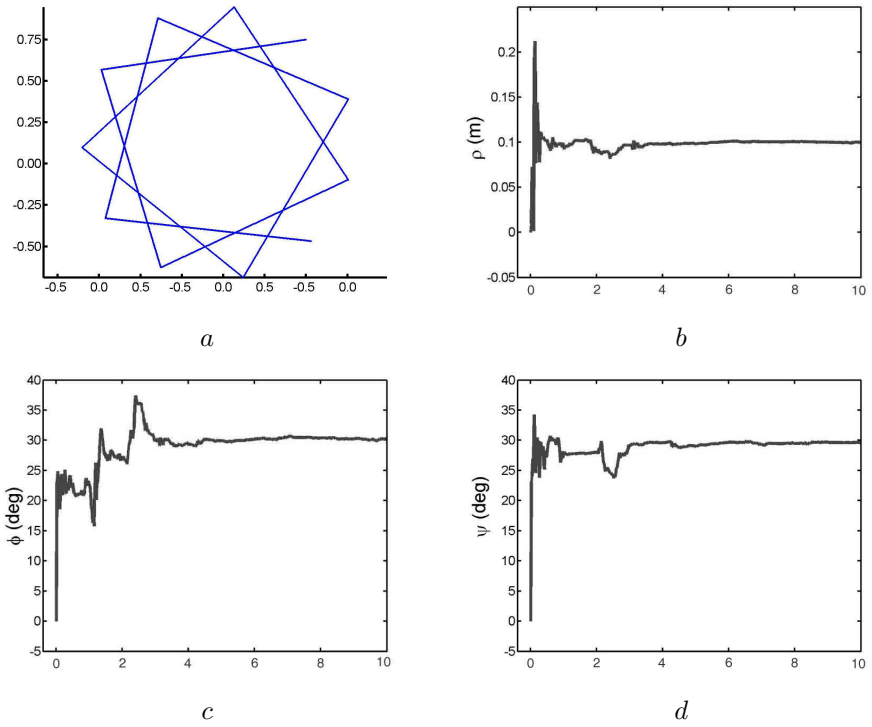


Figure 5.3: Results obtained by implementing the strategy introduced in Section 5.3 to estimate the three parameters ρ (b), ϕ (c) and ψ (d). The considered robot trajectory is displayed in (a). At each vertex the robot performs a pure rotation of approximately 450 *deg*.

5.5.2 Real experiments

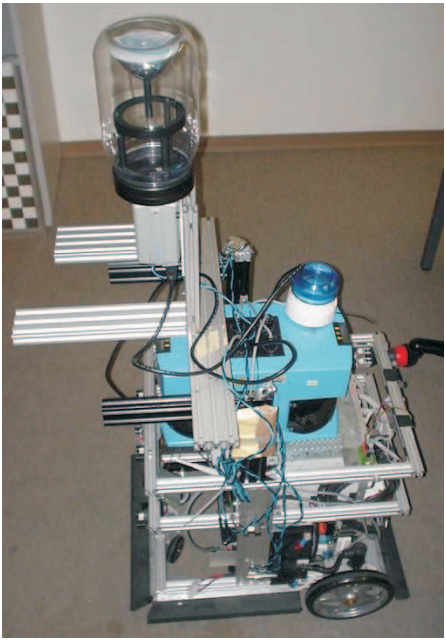


Figure 5.4: The robot used in our experimets equipped with encoder sensors on the wheels, an omnidirectional camera, and two laser range finders

For the experiments, we adopted a mobile robot with a differential drive system equipped with encoder sensors on the wheels. We equipped the robot with a calibrated omnidirectional camera. This camera is the same as used in the experiments of Section 3.3.2 and was intrinsically calibrated according to the method described in the Chapter 3. Furthermore, two laser range finders (model SICK LMS 200) were also installed on the robot. These laser scanners are used in our experiments just for comparison and are considered already calibrated with the odometry system according to the specifications provided by the manufacturer. A picture of the all settings is depicted in Fig. 5.4.

As a landmark to track, we placed a pole in the middle of the test room and we applied the vertical feature extraction and matching method described in Chapter 4. The robot was driven along a squared trajectory around the

pole, similar to the one used in the simulations (see Fig. 5.3(a)). At each vertex of the path, the robot performed a pure rotation of approximately 450 *deg*.

The camera was set on the robot with the following settings: $\phi \simeq -\pi$, $\rho \simeq 0.07m$ and $\psi \simeq -\frac{\pi}{2}$, all of them were measured manually. Then, the encoder data and the bearing measurements were fused according to the calibration procedure of Section 5.3.

The values of the three parameters estimated during the motion are plotted as a function of the distance in Fig. 5.5. It is possible to see that after 3 meters of navigation they start to converge to a stable value. The final estimated parameters were $\phi = -3.11rad$, $\rho = 0.074m$, and $\psi = -1.58rad$. Note that they are consistent with the values manually measured. Also observe that the plot of ρ starts at 0 since we took the robot origin as initial value for the EKF. Nevertheless, its estimation converges to the expected value.

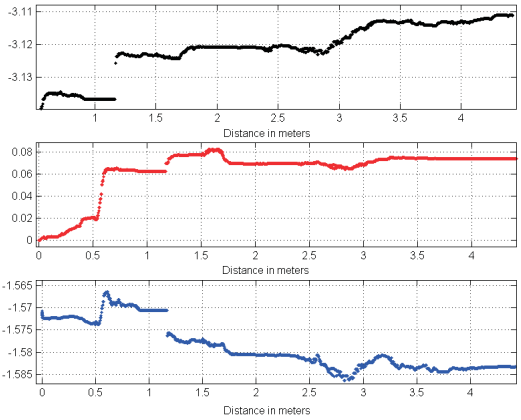


Figure 5.5: The plots of ϕ (black), ρ (red), and ψ (blue) estimated during the motion versus distance. The adopted units are in this case radians and meters.

5.6 Extension to multiple features

In this section, we extend the equations derived so far for the case one single feature to the general case of tracking multiple features. To this end, observe that in the multiple features' case the state vector $\mathbf{X}$ (5.14) can be rewritten as:

$$\mathbf{X} = [D^I, \theta^I, D^{II}, \theta^{II}, \ldots, D^Z, \theta^Z, \phi, \rho, \psi]^T, \tag{5.26}$$

where the superscript identifies the observed feature.

Using (5.15), the dynamics of the system is

$$\mathbf{X}_{i+1} \;=\; f\left(\mathbf{X}_i, \mathbf{u}\right) = \begin{bmatrix} D_i^I + \delta\rho_i cos\theta_i^I \\ \theta_i^I + \delta\theta_i - \frac{\delta\rho_i}{D_i^I} sin\theta_i^I \\ D_i^{II} + \delta\rho_i cos\theta_i^{II} \\ \theta_i^{II} + \delta\theta_i - \frac{\delta\rho_i}{D_i^{II}} sin\theta_i^{II} \\ \vdots \\ D_i^Z + \delta\rho_i cos\theta_i^Z \\ \theta_i^Z + \delta\theta_i - \frac{\delta\rho_i}{D_i^Z} sin\theta_i^Z \\ \phi_i \\ \rho_i \\ \psi_i \end{bmatrix} \tag{5.27}$$

Then, we have to determine the Jacobians $\mathbf{F_x}$ and $\mathbf{F_u}$ of the dynamics. Thus, from (5.16-5.17) we have

$$F_x = \begin{bmatrix} A^I & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & A^{II} & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & A^Z & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.28}$$

and

$$F_u = \begin{bmatrix} B^I \\ B^{II} \\ \vdots \\ B^Z \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.29}$$

with

$$A^i = \begin{bmatrix} 1 & -\delta\rho\ sin\theta^i \\ \frac{\delta\rho}{D^{i2}}\ sin\theta^i & 1 - \frac{\delta\rho}{D^i} cos\theta^i \end{bmatrix}, \tag{5.30}$$

$$B^i = \begin{bmatrix} \frac{cos\theta^i}{2} & \frac{cos\theta^i}{2} \\ \frac{1}{b} - \frac{sin\theta^i}{2D^i} & -\frac{1}{b} - \frac{sin\theta}{2D^i} \end{bmatrix} \tag{5.31}$$

Regarding the observations z, from (5.11) we have:

$$h\left(\mathbf{X}_i\right) = \begin{bmatrix} \beta^I \\ \beta^{II} \\ \vdots \\ \beta^Z \end{bmatrix} = \begin{bmatrix} \tan^{-1}\left(\frac{-\rho sin(\theta^I+\phi)}{-D^I-\rho\ cos(\theta^I+\phi)}\right) - \theta^I - \phi - \psi \\ \tan^{-1}\left(\frac{-\rho sin(\theta^{II}+\phi)}{-D^{II}-\rho\ cos(\theta^{II}+\phi)}\right) - \theta^{II} - \phi - \psi \\ \vdots \\ \tan^{-1}\left(\frac{-\rho sin(\theta^Z+\phi)}{-D^Z-\rho\ cos(\theta^Z+\phi)}\right) - \theta^Z - \phi - \psi \end{bmatrix} \tag{5.32}$$

And thus for the Jacobians $\mathbf{H}$ of the observations, we have:

$$\mathbf{H} = \begin{bmatrix} H1^I & H2^I & 0 & 0 & \cdots & 0 & 0 & H3^I & H4^I & -1 \\ 0 & 0 & H1^{II} & H2^{II} & \cdots & 0 & 0 & H3^{II} & H4^{II} & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & H1^Z & H2^Z & H3^Z & H4^Z & -1 \end{bmatrix} \tag{5.33}$$

where, from (5.18)

$$H1^i = \frac{-\rho sin(\theta^i + \phi)}{{D^i}^2 + 2\rho D^i cos(\theta^i + \phi) + \rho^2}, \tag{5.34}$$

$$H2^i = \frac{-D^i \rho\ cos(\theta^i + \phi) - {D^i}^2}{{D^i}^2 + 2\rho D^i cos(\theta^i + \phi) + \rho^2}, \tag{5.35}$$

$$H3^i = \frac{-D^i \rho\ cos(\theta^i + \phi) - {D^i}^2}{{D^i}^2 + 2\rho D^i cos(\theta^i + \phi) + \rho^2}, \tag{5.36}$$

$$H4^i = \frac{D^i sin(\theta^i + \phi)}{{D^i}^2 + 2\rho D^i cos(\theta^i + \phi) + \rho^2}\ . \tag{5.37}$$

5.6.1 Results

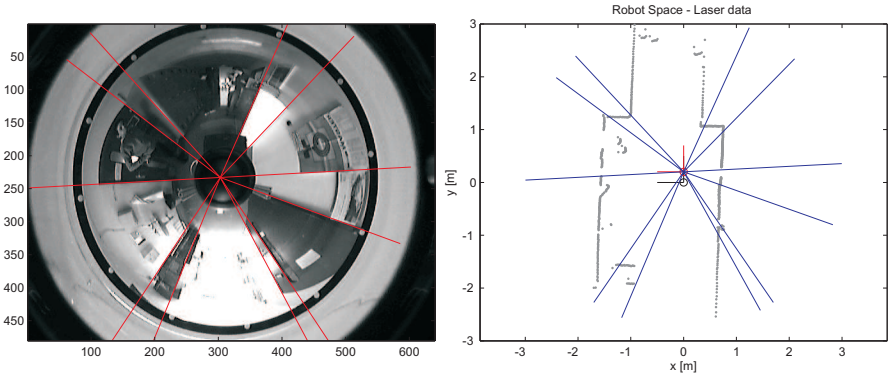


Figure 5.6: Scenario before calibration: (left) vertical features; (right) rays of bearing. The rays corresponding to the vertical features do not accurately intersect the corners in the laser scan.

In this section, we present some calibration results for the case of tracking several features.

As we did in Section 5.5.2, we positioned our omnidirectional camera on the robot and we measured manually its position relative to the robot. We measured the following values: $\phi \simeq 0\ rad$, $\rho \simeq 0.2\ m$, $\psi \simeq 0\ rad$. The scenario is shown in Fig. 5.6. In this figure, several vertical features used for the self-calibration experiment are highlighted. Furthermore, we show also a 2D map of the test environment, which is provided by a laser range

finder (remember we use the laser data just for the purpose of comparison as a ground truth). The robot reference system (in black) and the camera reference system (in red) are also indicated. The rays departing from the camera origin show the directions of the bearings of the vertical features. In this figure, the position of the camera relative to the robot is shown prior calibration, according to the values we measured manually. Note that, because the calibration is inaccurate, the rays of bearing do not intersect properly the corners in the map. However, we used these preliminary parameters for initializing the EKF. The trajectory chosen for the experiments consisted of a straight path, approximately 2.3 m long, and a 180 deg rotation about the center of the wheels. The trajectory is depicted in Fig. 5.8.

The values of the three parameters estimated during the motion are plotted versus the number of frame in Fig. 5.9. The covariances σ_ϕ, σ_ρ, σ_ψ are also plotted. Observe that after about 65 frames (corresponding to 2.3 m of navigation) the parameters start suddenly to converge to a stable value. The resulting estimated parameters are $\phi = -0.34rad$, $\rho = 0.23m$ and $\psi = 0.33rad$. The sudden jump starting at frame no. 60 actually occurs when the robot starts to rotate. As we pointed already out in Section 5.5.1, the convergence seems to be very fast when the robot performs trajectories formed of short straight paths and pure rotations. During the straight paths, the estimation of the parameters ϕ, ρ, and ψ is not good, especially when the direction of motion is parallel to the direction of the observation (this is quite intuitive to see). Conversely, when the robot performs a pure rotation, the parameters start to converge.

In Fig. 5.7, the scenario after calibration can be compared with the scenario prior calibration of Fig. 5.6. Observe, that the calibration parameters are estimated with high accuracy. Indeed, the rays of bearing intersect with the expected corners present in the laser map.

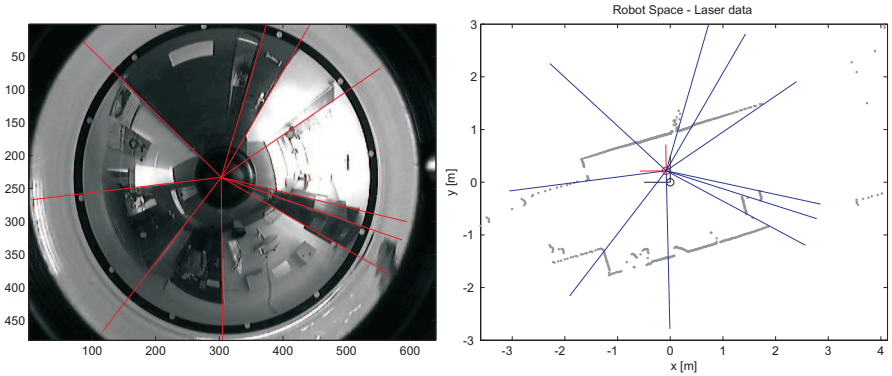


Figure 5.7: Scenario after calibration: (left) vertical features; (right) rays of bearing. The rays corresponding to the vertical features accurately intersect the corners in the laser scan.

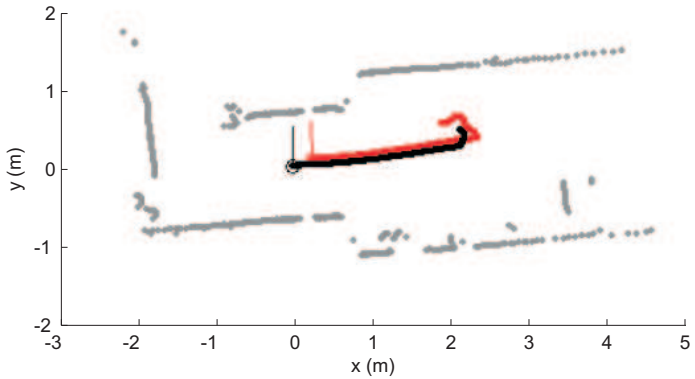


Figure 5.8: The path performed by the robot during self-calibration, i.e. straight path followed by a rotation about its center.

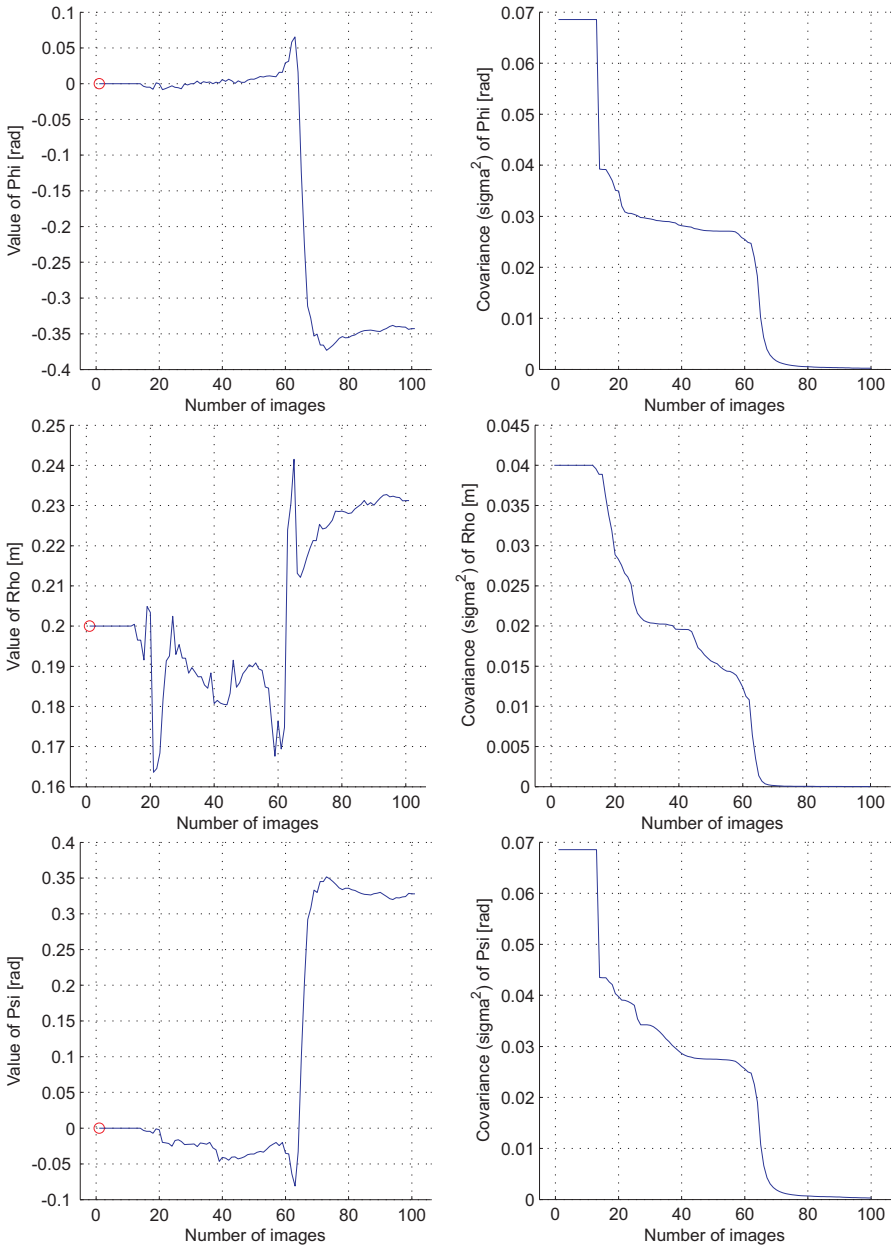


Figure 5.9: ϕ, ρ, ψ, ${\sigma_\phi}^2$, ${\sigma_\rho}^2$, ${\sigma_\psi}^2$ as a function of the frame number. The distance traveled between two frames is about 3.8 cm.

5.6.2 Robot motion estimation

We applied our line feature tracker and calibrated camera to an important problem of autonomous navigation, that is robot motion estimation. In this section, we show only the results as the theory is similar to that derived for self-calibration. We used a standard EKF to estimate the motion of the robot. In particular, the EKF estimates the vector:

$$X = [x_r, y_r, \theta_r, X_1, Y_1, ..., X_{N_O}, Y_{N_O}]^T \tag{5.38}$$

where $[x_r, y_r, \theta_r]$ is the robot configuration, X_i, Y_i are the Cartesian coordinates of the i-feature in the map, and N_O is the number of observed features. The bearing observations provided by the omnidirectional camera consist of the vector z whose components are:

$$\begin{aligned} z_1 &= arctan(\frac{Y_1 - y_r}{X_1 - x_r}) \\ &\vdots \\ z_{N_O} &= arctan(\frac{Y_{N_O} - y_r}{X_{N_O} - x_r}) \end{aligned} \tag{5.39}$$

To initialize a new feature in the map, consecutive bearing observations as in 5.32, which refer to the same feature, are integrated with the odometry. Then, the estimation is improved by integrating the information coming from all the bearing observations through the EKF. The result is shown in Fig. 5.10 where both the robot trajectory and the position of the features are shown.

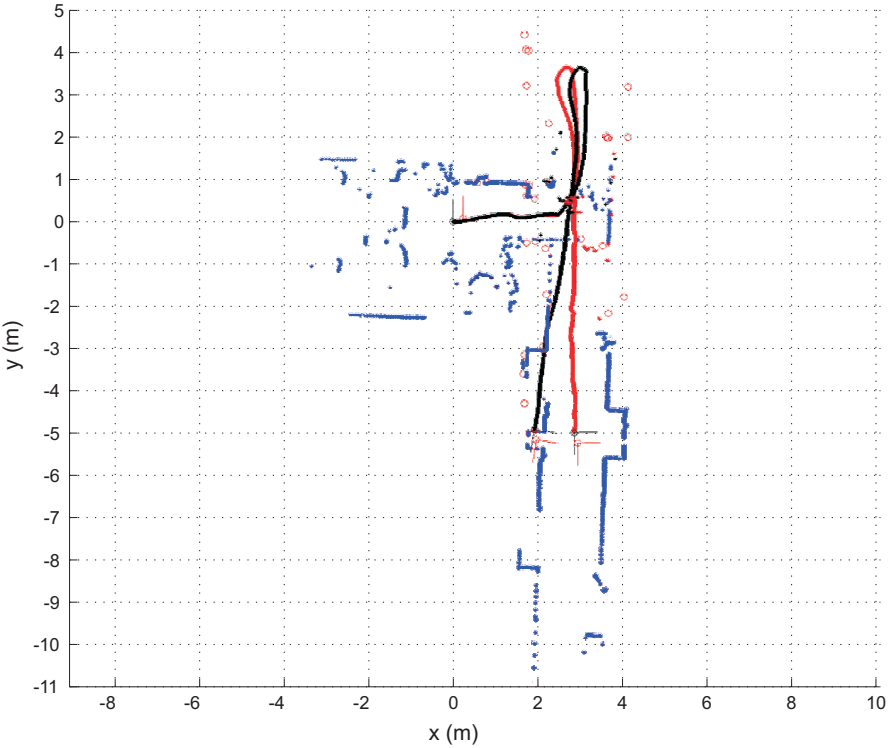


Figure 5.10: The results obtained by implementing a simple EKF based robot motion estimation which uses the proposed feature tracker. The black line is the trajectory estimated by using the odometry alone. The red line is the trajectory estimated by the EKF using both odometry and vertical lines. The blue points represent the map ground truth provided by a laser range finder. The red circles are the detected verticals features.

5.7 Conclusion

The contribution of this chapter is the development of a novel method to estimate the rigid body transformation between an omnidirectional camera and the reference frame of a robot (i.e. odometry). The parameters to be estimated (also called calibration parameters) are the position and orientation of the camera in the robot frame.

In our analysis, we restricted the problem to the simpler 2D case where the robot undergoes pure planar motion and the omnidirectional camera is perpendicular to the plane of motion. Under these assumptions, the problem consists in estimating the parameters ϕ, ρ, and ψ that describe the transformation between the two systems (see Fig. 5.1).

The novelty of the method is the use of an Extended Kalman Filter that automatically estimates the calibration parameters while the robot is moving. We call this method self-calibration because no a priori knowledge about the environment is required. Its implementation only requires the capability of robustly and visually tracking one or more features in the space. Although, experiments have been conducted using an omnidirectional camera, more in general the proposed method can be adopted to calibrate any robot bearing sensor.

The proposed strategy was deeply validated through a theoretical analysis. In particular, an observability analysis which takes into account the system nonlinearities was carried out and clearly shows that the system contains all the necessary information to estimate the calibration parameters. Furthermore, many accurate simulations and experiments performed with a real robot and an omnidirectional camera fully validated this strategy. In particular, they show that by choosing suitable trajectories it is possible to estimate the parameters with high accuracy by moving the robot along very short paths (few meters).

This work has pointed out several areas that deserve further investigations:

- apply optimal control methods in order to find the best robot trajectory which minimizes the error of the estimated parameters;
- consider the effect of a systematic component on the odometry;
- in the case of an omnidirectional camera, introduce other two parameters (e.g. two angles) to take into account the orientation of the mirror

axis (here assumed to be perpendicular to the plane of motion).

Chapter 6

Calibration between camera and 3D laser range finder

In this chapter, we face the problem of the extrinsic calibration between an omnidirectional camera and a 3D laser range finder. Our contribution is a new method that allows to calibrate the two sensors on the fly. Indeed, our approach does not require any calibration object, conversely it uses point correspondences which are manually selected by the user from a scene viewed by the two sensors. The proposed method relies on a novel technique to visualize the range information obtained from a 3D scanner. This technique converts the visually ambiguous 3D range information into a 2D map where natural features of a scene are highlighted. We show that by enhancing the features the user can easily find the corresponding points of the camera image points. Therefore, visually identifying laser-camera correspondences becomes as easy as image pairing. Once point correspondences are given, extrinsic calibration is done using the well-known PnP algorithm followed by a non-linear refinement process. We show the performance of our approach through experimental results.

6.1 Introduction

6.1.1 State of the art

ONE of the basic issues of mobile robotics is the automatic mapping of the environments. Digital 3D models of the environment are needed in autonomous navigation, rescue and inspection robotics, facility management, and architecture. Autonomous mobile robots equipped with 3D laser range finders are well suited for this task.

Recently, several techniques for acquiring three-dimensional data with 2D range scanners installed on a mobile robot have been developed (see [86–88]). A popular approach is to use multiple scanners that point towards different directions [86]. An alternative is to use pan/tilt devices that sweep the range scanner in an oscillating way [89], [87]. More recently, techniques for rotating 2D range scanners have been developed [88].

However, to create realistic virtual models, visually-perceived information from the environment has to be acquired and it has to be precisely mapped onto the range information. To accomplish this task, camera and 3D laser range finder must be extrinsically calibrated, that is, the rigid transformation between the two reference systems must be estimated.

Most previous works on extrinsic laser-camera calibration concern calibration of perspective cameras to 2D laser scanners (see [90–92]). Furthermore, some of these works use visible lasers [93]. In contrast to previous works, in this chapter we consider the extrinsic calibration of a general camera with a 3D laser range finder where the laser points are invisible to the camera.

Because of the recent development of 3D laser scanners, only little work about extrinsic calibration of camera and 3D scanners exists. Furthermore, the process of external calibration is often poorly documented. This process usually requires some modification of the scene by introducing landmarks that are visible by both the camera and the laser. For instance, some approaches use high reflectance surfaces while other approaches use 3D spheres as markers. In the latter, using a fitting procedure along with prior knowledge of the radius of the sphere, the 3D coordinates of the sphere center can be estimated quite accurately on condition that there are sufficient laser returns off its surface. The spheres can also be easily detected in an image. Then, extrinsic calibration can be done using these 3D-2D correspondence

pairs of the sphere centers. However, the spheres need to be quite big to be accurately detected by the laser. Thus, this method is not portable.

Well-documented work about extrinsic calibration of camera and 3D scanners can be found in [94,95]. However, in [94], the authors deal with the case of visible laser traces. Conversely, for the case of invisible laser in [95], the authors propose a method for fast extrinsic calibration of a camera and a 3D scanner which makes use of a checkerboard calibration target, like the one commonly used for the internal calibration of a camera (as in Chapter 3). Furthermore, they provide a useful laser-camera calibration toolbox for Matlab that implements their procedure [96]. Their method requires the user to collect a few laser-camera acquisitions where the calibration grid is shown at different positions and orientations. Then, the user is asked to manually select the region of points in the camera and laser images, which contain the grid. Finally, a plane is fitted to the selected points of each view pair and calibration is done by minimizing the difference in orientation and distance of the planes observed by the two sensors. This technique however needs several camera-laser acquisitions of the grid for a sufficiently accurate external calibration of the system.

6.1.2 Motivation and outline

The work described in this chapter also focuses on the extrinsic calibration of a camera and a 3D laser range finder but the primary difference is that we do not use any calibration pattern. We use only point correspondences that the user hand-selects from a single laser-camera acquisition of a natural scene. As we use no calibration target, we name our technique self-calibration.

This work was motivated while working in the First European Land-Robot Trial [97]. In that contest, we presented an autonomous Smart car (Fig. 6.1) equipped with several 3D laser range finders and cameras (both omnidirectional and perspective cameras). The goal was to produce 3D maps of the environment along with textures [98].

Especially when working in outdoor environments, doing several laser-camera acquisitions of a calibration pattern can be a laborious task. For each acquisition, the pattern has to be moved to another position and this process usually takes time. Furthermore, weather conditions (e.g. wind, fog, low visibility) can sometimes perturb or even alter the calibration settings.

Figure 6.1: Our experimental robotic platform SMARTER with which we participated in ELROB (the first European Land Robot Trial) [97].

Hence, the calibration must be done quickly. Because of this, we developed the procedure presented in this chapter. The advantages are that now we need only a single laser-camera acquisition and that the calibration input are point correspondences manually selected from a laser-camera acquisition of a natural scene.
Once point correspondences are given, the extrinsic calibration problem becomes a camera pose-estimation problem which is well known in computer vision and can be solved using standard methods.

The difficulty resides in visually identifying the point correspondences because range images in general lack in point features. To bypass this problem, we process the range data so that we can highlight discontinuities and orientation changes along specific directions. This processing transforms the range image into a new image that we call Bearing Angle image (BA). Using BA images, we will show that visually identifying point correspondences between

the laser and the camera outputs becomes as easy as image pairing.

To show the generality of the methodology, in our experiments we will use an omnidirectional camera. The BA images and the application of the method to an omnidirectional camera are the two main contributions of this chapter; the results have been published in [9].

The chapter is organized as follows. Section 6.2 describes the projection model of the system camera-laser. Section 6.3 defines the concept of BA images and explains how to compute them. Section 6.4.2 describes the calibration procedure. Finally, section 6.5 presents some calibration results.

6.2 Camera-laser projection model

6.2.1 Camera model

Without loss of generality, in this chapter we consider central omnidirectional cameras but the same considerations also apply to perspective cameras. Furthermore, we assume that our camera is already calibrated (e.g. as in Chapter 3) and, thus, for every pixel point $\mathbf{u'} = [u', v']$ on the camera image plane we can recover the orientation of the vector $\mathbf{q}$ emanating from the single effective viewpoint to the corresponding scene point $\mathbf{X}$ (see Equations (2.2), (2.10) and Fig. 2.8). The inverse mapping is also possible; that is, given $\mathbf{q}$ we can remap it to $\mathbf{u'}$ (see Equations 2.2, 2.11).

Formally, we can express the direct and inverse relation between $\mathbf{q}$ and $\mathbf{u'}$ through function F and its inverse F^{-1} as:

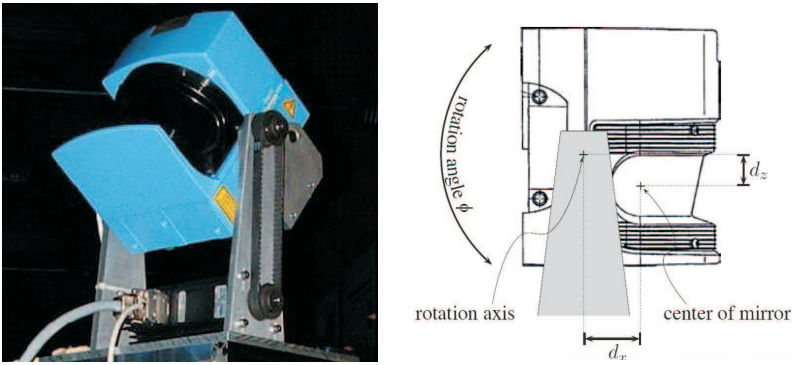
$$\mathbf{q} = F(\mathbf{u'}) \tag{6.1}$$

$$\mathbf{u'} = F^{-1}(\mathbf{q}) \tag{6.2}$$

where $\mathbf{q}$ is defined as in 2.2 and holds $\|\mathbf{q}\| = 1$. F is a function $\mathbb{R}^2 \rightarrow \mathbb{R}^3$ that depends on h, g as defined in Chapter 2.

6.2.2 Laser model

3D laser range finders are usually built by nodding or rotating a 2D scanner in a stepwise or continuous manner around its lateral or radial axis (Fig. 6.2). Combining the rotation of the mirror inside the 2D scanner with the external rotation of the scanner itself, spherical coordinates of the measured points



a b

Figure 6.2: (a) Our custom-built 3D scanner is composed of a SICK LMS 200 laser range finder mounted on a rotating support. (b) Schematic of the sensor used for calibration.

are obtained. However, since in reality it is impossible to adjust the two centers of rotation exactly on the same point, the measured parameters are not spherical coordinates and offset values exist. These offset values have to be estimated by calibrating the 3D sensor by considering its observation model.

The approach presented in this chapter for the extrinsic calibration of a camera with a 3D laser scanner is general and does not depend on the sensor model. Therefore, we assume the laser is already calibrated. Neverthless, we explain here the scanner model used in our experiments, but a different sensor setup could also be used along with its corresponding observation model.

The 3D range sensor used in this work is a custom-built 3D scanner (Fig. 6.2). It is composed of a two-dimensional SICK laser scanner mounted on a rotating support which is driven by a Nanotec stepping motor. The sensor model can be written as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c_i c_j & -c_i s_j & s_i & c_i d_x + s_i d_z \\ s_j & c_j & 0 & 0 \\ -s_i c_j & s_i s_j & c_i & -s_i d_x + c_i d_z \end{bmatrix} \cdot \begin{bmatrix} \rho_{ij} \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{6.3}$$

with $c_i = \cos(\varphi_i)$, $c_j = \cos(\theta_j)$, $s_i = \sin(\varphi_i)$, $s_j = \sin(\theta_j)$.
Furthermore, ρ_{ij} is the j-th measured distance with corresponding orientation

θ_j in the i-th scan plane, which makes the angle φ_j with the horizontal plane (Fig. 6.2(b)).
The offset of the external rotation axis from the center of the mirror in the laser frame has components d_x and d_z (as observed in Fig. 6.2(b)).
$[x, y, z]^T$ are the coordinates of each measured point relative to the global frame (with its origin at the center of the rotation axis, the x-axis pointing forward and the z-axis toward the top).
The sensor is calibrated as discussed in [99] based on a known ground truth.

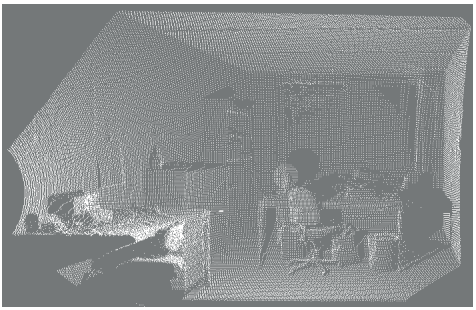
6.3 Bearing angle images

In this section, we describe how to highlight depth discontinuities and direction changes in the range image so that the user can easily find the corresponding points of the camera image points. Such features in the range image are called image details.

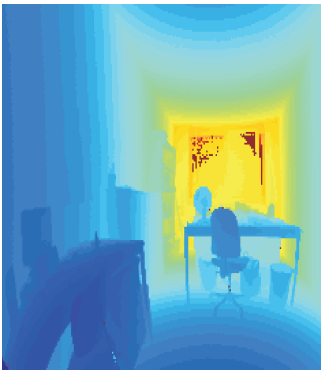
Fig. 6.3(b) shows the range image of an office-like environment extracted by our 3D scanner. In such an environment, we would like emphasizing key points like corners arising from the plane intersections of walls, tables, chairs, and other similar discontinuities.

Fig. 6.3(c) shows the result of directly applying a Sobel edge detector to the range image. As observed, edge detection does not directly help for our task, since edges are zones of the range image where the depth between two adjacent points significantly changes. In fact, many details in the range image do not create a big jump in the measured distance. Such edges are called "roof edges" and correspond to sharp direction changes (e.g. tetrahedron shaped corners). Therefore, a measure of direction should be used to highlight all the desired details in the scene. As representative of the surface direction, its corresponding normal vector is usually used (see [100, 101]). Surface normal vectors are estimated based on the neighborhood of each point. However, for our application, we avoid the use of surface normals as representatives of the direction. The reason is that we want to highlight the details of the surface along some specific directions (e.g vertical, horizontal, and diagonal). We will show that treating each dimension separately leads to enhanced estimation of the image details.

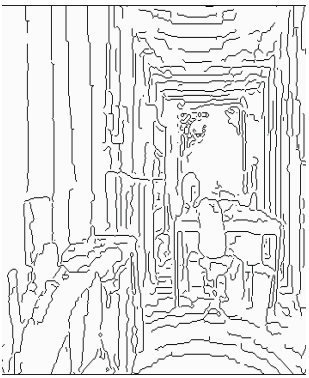
Let the range data coming from the 3D scanner be arranged in the form of a 2D matrix where its entries are ordered according to the direction of the laser beam. This matrix will be referred to as a depth matrix.



a



b



c

Figure 6.3: (a) 3D point cloud of an office. (b) Depth image of the same scene. Jet colormap has been used. The color shade (from blue to red) is proportional to the depth. (c) Sobel based edge map of the depth image.

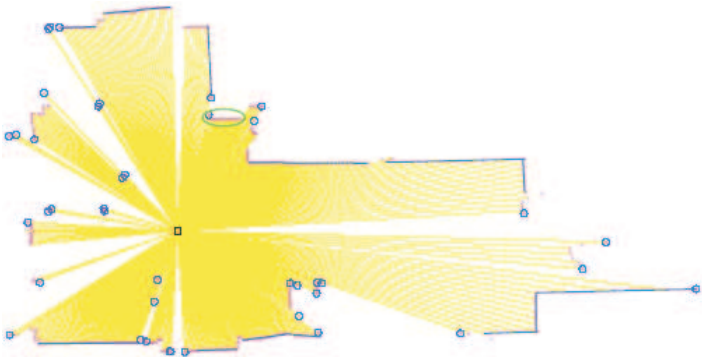
We compute the surface orientation along four separate directions of the depth matrix, namely the horizontal, vertical, and diagonal one (the latter having $+45^\circ$ and -45° orientation).

We define Bearing Angle (BA) the angle between the laser beam and the segment joining two consecutive measurement points (see Fig. 6.4(b)). This angle is computed for each point in the depth matrix along the four defined directions (that we call also "traces"). More formally:

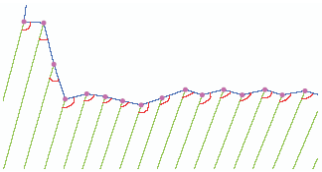
$$BA_i = \arccos \frac{\rho_i - \rho_{i-1} \cos d\varphi}{\sqrt{\rho_i^2 - \rho_{i-1}^2 - \rho_i \rho_{i-1} \cos d\varphi}} \tag{6.4}$$

where ρ_i is the i-th depth value in the selected trace of the depth matrix and $d\varphi$ is the corresponding angle increment (laser beam angular step in the direction of the trace).

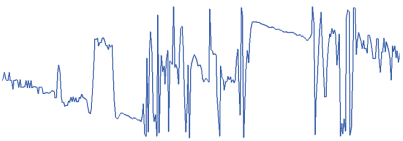
Performing this calculation for all points in the depth matrix will lead to an image that we call BA image. BA images can be computed from the depth matrix along any direction to highlight the details of the scene in the selected direction. In our application, horizontal, vertical, and diagonal traces suffice for a successful enhancement of the details of the scene (Fig. 6.5). However, any other direction could be also considered depending on the application. As observed in Fig. 6.5, these angular measures show the geometry of the scene by highlighting many details that were not distinguishable in the range image (Fig. 6.3(b)). Hence, these will be used in the next section for extracting corresponding features.



a



b



c

Figure 6.4: (a) A sample bird-eye-view of a laser scan taken in our laboratory. (b) Bearing Angles computed along the given scan plan. (c) Plot of a horizontal BA signal.

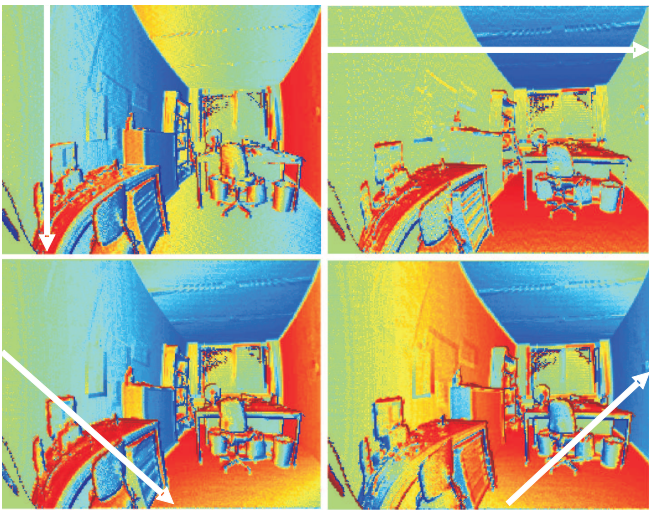


Figure 6.5: BA images for a real scan (top left: vertical, top right: horizontal, and bottom: two diagonal directions). Observe that in BA images the scene details are very highlighted (e.g. even the corners of the picture hanged on the left wall are now well distinguishable; in the range image of Fig. 6.3(a), they were not). In this pictures, jet colormap has been used. The color shade (from blue to red) is proportional to the BA value.

6.4 Laser-camera calibration

6.4.1 Data Collection

Our calibration technique needs a single acquisition of both laser and camera. The acquisition object can be any natural scene with a sufficient number of distinguishable key points (e.g. roof edges or depth discontinuities).

Our calibration procedure consists of three stages:

1. Compute the BA images of the acquired range image.
2. Hand-select several point correspondences (at least four) between the BA. image and intensity image (Fig. 6.6).
3. Perform calibration using a camera pose estimation algorithm followed by a non-linear refinement.

Observe that usually not all four BA images are needed. Depending on the scene and the orientation of the laser scanner to the scene, only the horizontal BA image could suffice. However, the remainder BA images can be used anyway to check whether there are further details that would be worth exploiting.

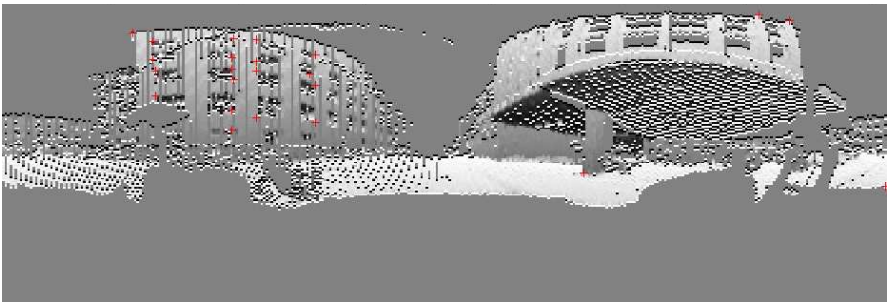
At the end of the visual correspondence pairing, we have n laser points in the laser frame and their correspondent points on the camera image plane. We write the laser and camera points in the following way:

$$\begin{aligned} \mathbf{q}_C &= [\mathbf{q}_{C,1}, \mathbf{q}_{C,2}, ..., \mathbf{q}_{C,n}], \\ \mathbf{q}_L &= [\mathbf{q}_{L,1}, \mathbf{q}_{L,2}, ..., \mathbf{q}_{L,n}], \\ d_L &= [d_{L,1}, d_{L,2}, ..., d_{L,n}], \end{aligned} \tag{6.5}$$

where $\mathbf{q}_C$ and $\mathbf{q}_L$ are the unit norm orientation vectors of the camera and laser points respectively in their reference frames. d_L are the point depths in the laser frame.

6.4.2 Calibration

Extrinsic calibration of a camera and a 3D laser range finder consists in finding the rotation $\mathbf{R} \in SO(3)$ and the translation $\mathbf{T} \in \mathbb{R}^3$ between the laser frame and the camera frame that minimizes a certain error function. In photogrammetry, the function to minimize is usually the reprojection error:



a

b

Figure 6.6: Thanks to BA images, visually identifying point correspondences between laser (b) and camera (a) becomes as easy as image pairing. Several point correspondences are highlighted in red. (b) BA image, (a) omnidirectional image after unwrapping

$$\min_{\mathbf{R},\mathbf{T}} \frac{1}{2} \sum_{i=1}^{n} \left\| \mathbf{u'}_i - \hat{\mathbf{u}}'(\mathbf{R}, \mathbf{T}, \mathbf{X}_{\mathbf{L}i}) \right\|^2 \tag{6.6}$$

where $\hat{\mathbf{u}}'(\mathbf{R}, \mathbf{T}, \mathbf{X}_{\mathbf{L}i})$ is the reprojection onto the image plane of the laser point $\mathbf{X}_{\mathbf{L}i}$ according to Equations (2.2) and (6.2).
However, the reprojection error is not theoretically optimal in our application because the resolution of the camera is not uniform. A better error function uses the Riemann metric associated to a sphere as it takes into account the spatial distribution (see [91,95]). This metric minimizes the difference of the bearing angles of the camera points and the bearing angles of the laser points after reprojection onto the image, that is:

$$\min_{\mathbf{R},\mathbf{T}} \frac{1}{2} \sum_{i=1}^{n} \left\| \arccos(\mathbf{q}_{C,i}^T \cdot \hat{\mathbf{q}}_{C,i}) \right\|^2 \tag{6.7}$$

where $\hat{\mathbf{q}}_C$ is the unit norm orientation vector corresponding to $\hat{\mathbf{u}}'(\mathbf{R}, \mathbf{T}, \mathbf{X}_{\mathbf{L}i})$.

According to equation (6.2), each correspondence pair contributes two equations. In total, there are $2n$ equations in 6 unknowns. Hence, at least 3 point associations are needed to solve for $\mathbf{R}$ and $\mathbf{T}$. However, 3 point associations yield up to four solutions and thus a fourth correspondence is needed to remove the ambiguity. This problem has already been theoretically investigated for a long time and is well known in the computer vision community as PnP problem (Perspective from n Points). Some solutions to this problem can be found in [102, 103].

We implemented the PnP algorithm described in [102]. The output of the PnP algorithm are the depth factors of the camera points in the camera reference frame, that is, $d_C = [d_{C,1}, d_{C,2}, ..., d_{C,n}]$. Finally, to recover $\mathbf{R}$ and $\mathbf{T}$ we used the motion estimation algorithm proposed by Zhang [104].

6.4.3 Non-linear optimization

The drawback of using the PnP algorithm is that the solution is quite sensitive to the position of the input points (that were hand-selected) and also to noisy range information. Furthermore, we have to take into account that the two sensors can have different resolution and that the rigid transformation is recovered by linear least-square estimation. Thus, the solution given in Section 6.4.2 is suboptimal.

Table 6.1: Results

$T(m)$	σ	R(deg)	σ	*Pixel error*	σ
0.207	0.05	0.64	0.21		
0.042	0.017	-1.24	0.85	1.6	1.2
0.139	0.005	166.95	1.08		

To refine the solution, we minimized (6.7) as a non-linear optimization problem by using the Levenberg-Marquardt algorithm [41, 42]. This requires an initial guess of $\mathbf{R}$ and $\mathbf{T}$ which is obtained using the method described in Section 6.4.2.

6.5 Results

The proposed method has been tested on the custom-built rotating scanner described in Section 6.2.2 and on the omnidirectional camera used in Section 3.3.2. The camera resolution was set equal to 640×480 pixels.
The rotating scanner provided 360° field of view range measurements with a vertical angular resolution of 1° and a horizontal resolution of 0.5°. The camera was calibrated as described in Chapter 3 while the laser was calibrated using a known ground truth as explained in [99].

Performance w.r.t. the number of points

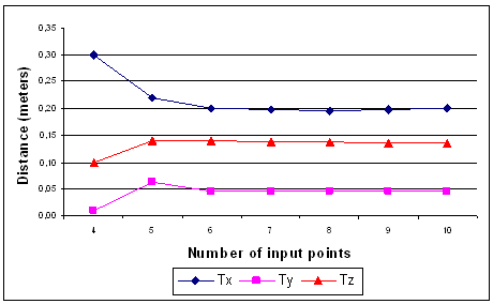


Figure 6.7: Estimation of the translation (meters) versus the number of selected points.

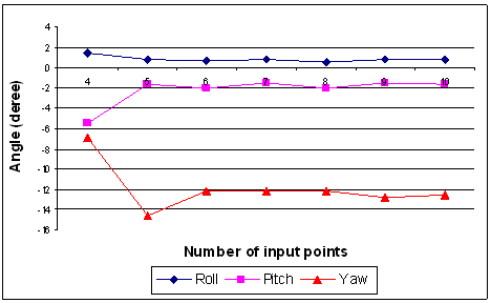


Figure 6.8: Estimation of the rotation (roll, pitch, and yaw angles) versus the number of selected points (the x-axis ranges from 4 to 10).

We evaluated the robustness of the proposed approach with respect to the number of manually selected points.
In particular, we varied the number of laser-camera correspondences from 4 to 10 and for each combination we did ten calibration trials using different input points. The results shown in Fig. 6.7, 6.8 are the average.

Observe that after selecting more than 5 points, the values of the estimated $\mathbf{R}$ and $\mathbf{T}$ become rather stable. This stability occurs when points are chosen uniformly from the entire scene viewed by the sensors. Conversely, when points are selected within local regions of the scene, the estimated extrinsic parameters are biased by the position of this region.
We also tried to use more than 10 points but the estimated parameters did not deviate from the average values that had been already estimated.

The estimated $\mathbf{R}$ and $\mathbf{T}$ were in agreement with the hand-measured values. Furthermore, the estimated parameters were stable against the position of the input points when these input points were picked uniformly from all around the scene.

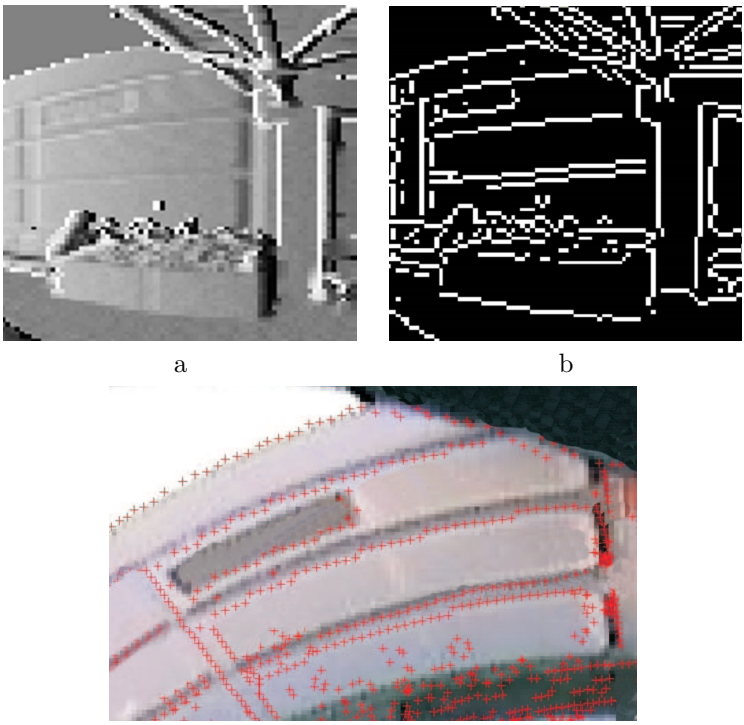
In Table 6.1, the mean and the standard deviation of $\mathbf{R}$ (expressed by Euler angles roll, pitch, and yaw) and $\mathbf{T}$ (with $\mathbf{T} = [T_x, T_y, T_z]$) are shown for the case of ten correspondence pairs. The results were averaged among ten different calibration trials.
Table 6.1, shows also the reprojection error (in pixel). For data fusion, this error is the most important. It measures the distance between the laser points reprojected onto the image using the estimated $\mathbf{R}$ and $\mathbf{T}$, and the image

points. In our experiments, the average reprojection error was 1.6 pixels and the standard deviation was 1.2 pixels.

Point cloud texturing

The reprojection of the laser points onto the image also offers an indirect way to evaluate the quality of the calibration. To do this, we chose not to reproject all the laser points onto the image. Rather, we reprojected only those laser points that represent discontinuities in the range image.
To select only depth discontinuities automatically, we applied an edge detector to the BA image. The edge points, as representative of depth discontinuities, were then reprojected onto the image.
The reprojection results are shown in Fig. 6.9. As observed, the laser edge points well reprojected onto the edges of the intensity image.

In the end, using the estimated $\mathbf{R}$ and $\mathbf{T}$, we colored an entire 3D scan by reprojecting the scan onto the corresponding image. The results of this color mapping are shown in Figures 6.10, 6.11. Figures 6.11(e),(f) are obtained using a Spherocam HDR camera [105] while the other figures are obtained using the same catadioptric camera mentioned in Section 3.3.2.

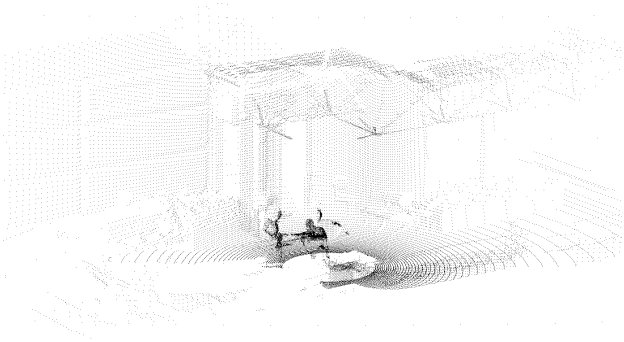


a b

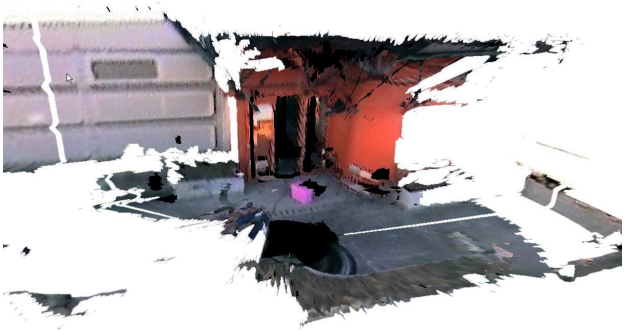
c

Figure 6.9: (a) A detail of the BA image. For visualization, we used a gray scale where the intensity is proportional to the BA value. (b) Result of a Sobel edge detector on the BA image. (c) The edges are reprojected onto the image using the computed **R** and **T**.

a



b



c

Figure 6.10: (a) An omnidirectional picture unwrapped into a rectangular image. The size of the original omnidirectional image was 640×480 pixels. (b) The 3D point cloud of the same scene extracted by our 3D laser range finder. (c) Result of texturing.

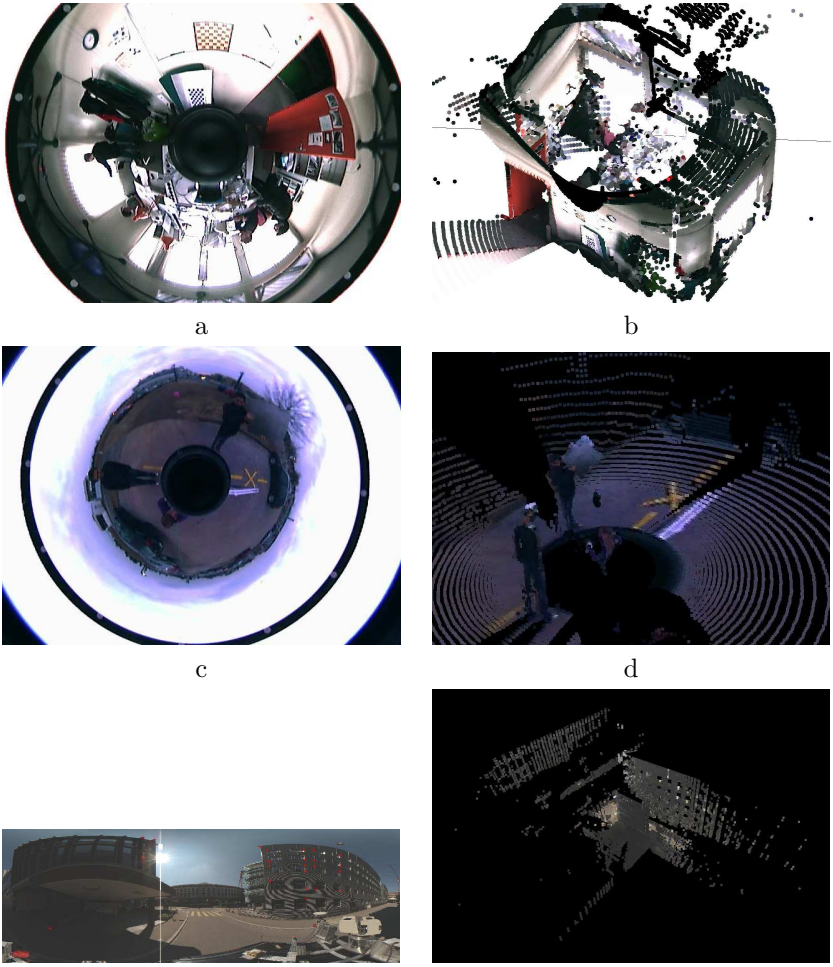


Figure 6.11: (b), (d), Results of texturing using images (a), (c). (e) A panoramic pictured captured with a Spherocam HDR [105] and (f) the textured 3D point cloud.

6.6 Conclusion

In this chapter, we presented a new approach for the extrinsic calibration of a camera with a 3D laser range finder, that can be done on the fly. The method uses only correspondent points that are manually selected by the user from a single laser-camera acquisition of a natural scene.

Our method relies on a novel technique to visualize the range information. This technique converts the visually ambiguous 3D range information into a 2D map (called BA image) where natural features of a scene are highlighted. In this way, finding laser-camera correspondences is facilitated. Once correspondence pairs have been given, calibration is done using the PnP algorithm followed by a non-linear refinement process.

Real experiments have been conducted using an omnidirectional camera and a rotating scanner, but the same approach can be also applied to any other type of camera (e.g. perspective) or laser range finder.
The results showed that by selecting the input points uniformly from the whole scene, robust calibration can be done by using only a few correspondences (at least eight or ten).

The BA images and the application of the method to an omnidirectional camera are the two main contributions of this chapter. Furthermore, the implication of the proposed calibration approach is important because it brings 3D computer vision systems out of the laboratory and into practical use. In fact, the proposed approach requires no special equipment and allows the user to quickly calibrate the system in those cases where special settings are difficult to be arranged (Section 6.1.2).

Chapter 7

Visual Odometry

In this chapter, we describe an algorithm for computing the ego-motion of a vehicle relative to the road. The algorithm uses, as only input, images provided by a single calibrated omnidirectional camera mounted on the roof of the vehicle. The front ends of the system are two different trackers. The first one is a homography-based tracker that detects and matches robust scale invariant features that most likely belong to the ground plane. The second one uses an appearance based approach and gives high resolution estimates of the rotation of the vehicle. This 2D pose estimation method has been successfully applied to videos from an automotive platform. We give an example of camera trajectory estimated purely from omnidirectional images over a distance of 400 meters. For performance evaluation, the estimated path is superimposed onto an aerial image. In the end, we use image mosaicing to obtain a textured 2D reconstruction of the estimated path.

7.1 Introduction

7.1.1 State of the art

ACCURATE estimation of the ego-motion of a vehicle relative to the road is a key component for autonomous driving and computer vision based driving assistance. Using cameras instead of other sensors for computing ego-motion allows for a simple integration of ego-motion data into other vision based algorithms, such as obstacle, pedestrian, and lane detection, without the need for calibration between sensors. This reduces maintenance

and cost. In the robotics community as well, effective use of video sensors for obstacle detection and outdoor navigation has been a goal for many years.

Most of the work in estimating robot motion has been produced using stereo cameras, and can be traced back to Moravec's work [106]. Similar work has been reported also elsewhere (see [107–109]). Furthermore, stereo visual odometry has also been successfully used on Mars by the NASA rovers since early 2004 [110]. Nevertheless, visual odometry methods for outdoor applications have been also produced, which use a single camera alone.

The problem of recovering relative camera poses and 3D structure from a set of 2D camera images has been largely studied for many years and is known in the computer vision community as "structure from motion" [11]. Very successful results have been obtained over long distances using either perspective or omnidirectional cameras (see [109, 111]). In [109], the authors deal with the case of a stereo camera but they also provide a monocular solution implementing a fully structure from motion algorithm that takes advantage of the 5-point algorithm and RANSAC robust estimation. In [111], the authors provide two approaches for monocular visual odometry based on omnidirectional imagery. In the first approach, they use optical flow computation, while in the second one full structure from motion.

Closely related to structure from motion is what is known in the robotics community as Simultaneous Localization and Mapping (SLAM), which aims at estimating the motion of the robot while simultaneously building and updating the environment map. SLAM has been most often performed with other sensors than regular cameras, however in the last years successful results have been obtained using single cameras alone (see [112–115]). Recently in [114], the authors presented a method for mapping large loops with a single hand-held camera. There, the authors extend Davison's work on visual 3D-SLAM [113] and build outdoor, closed-loop maps much larger than previously achieved with visual input alone.

7.1.2 Motivation and outline

In this chapter, we do not deal with visual SLAM, rather we concentrate on the development of a vision based method to estimate the motion of outdoor ground vehicles over long distances. In our approach, we used a single calibrated omnidirectional camera mounted on the roof of the car. We as-

sume that the vehicle undergoes a purely two-dimensional motion over a predominant flat ground. Furthermore, because we wanted to perform visual odometry in city streets, flat terrains, as in well as in motorways where buildings or 3D structure are not always present, we chose to estimate the motion of the vehicle by tracking the ground plane.

Ground plane tracking has been already exploited by the robotics community for indoor visual navigation and most works have been produced using standard perspective cameras [116–119]. In those works, the motion of the vehicle is estimated by using the property that the projection of the ground plane into two different camera views is related by a homography.

In this chapter, we propose a similar approach for central omnidirectional cameras, but our goal is to estimate the ego-motion of the vehicle in outdoor environments and over long distances. Thanks to the large field of view of the panoramic camera, interesting points from all around the car are extracted and matched from pairs of consecutive frames. Our key points are Scale Invariant Features (SIFT) [76], as they proved to work well also with omnidirectional pictures [120].

Furthermore, we want to extract those key points that only belong to the ground plane. To retain only these points and discard all the rest, we use a RANSAC based outlier removal, which uses the constraint that coplanar points seen from different views are related by a homographic transformation. The remaining inliers are then used to compute the rotation and translation matrices. To update the motion, we use only the magnitude of the translation because the rotation estimated from features alone gives rise to large drift errors after several hundreds of meters. Conversely, to estimate the rotation angle of the vehicle, we use an appearance based tracker. We show that by using this second tracker the drift error stays very low over several hundreds of meters.

The performance of our approach has been evaluated on a real platform. We will show an example of camera trajectory estimated purely from omnidirectional images over a distance of 400 meters. For performance evaluation, the estimated path is superimposed onto a satellite image of the same test environment. Furthermore, we use image mosaicing to obtain a textured 2D reconstruction of the estimated path.

The fusion of a feature based approach and an appearance based method

along with its application to an omnidirectional camera are the two main contributions of this chapter. This work led to a publication [10] and at the moment when this PhD thesis is being published, we have been notified that it was also conditionally accepted for the IEEE Transactions of Robotics.

This chapter is organized as follows. Section 7.2 describes our homography based ground plane navigation. Section 7.3 describes the appearance based tracker. Section 7.4 details the steps of the whole visual odometry algorithm. Finally, section 7.5 presents some experimental results.

7.2 Homography Based Ground Plane Navigation

The motion information that can be extracted by tracking 2D features is central to our vehicle navigation system. Therefore, we briefly review here a method that uses planar constraints and point tracking to compute the motion parameters.

7.2.1 Homography and Planar Motion Parameters

Early work on exploiting coplanar relations has been presented by Tsai and Huang [121], Longuet-Higgins [122] and Faugeras and Lustman [123]. The coplanar relation between two different views of the same plane can be summarized as follows. Consider two camera-centered coordinate systems, frame 1 and frame 2, which are related by a rigid body transformation:

$$\mathbf{X_2} = \mathbf{R}\mathbf{X_1} + \mathbf{T}, \tag{7.1}$$

where $\mathbf{X_1}, \mathbf{X_2} \in \mathbb{R}^\mathbf{3}$ are the coordinates of the same 3D point, respectively expressed in frame 1 and 2, and $\mathbf{R} \in \mathbf{SO(3)}$, $\mathbf{T} \in \mathbb{R}^\mathbf{3}$ are the rotation and the translation matrices encoding the relative position of the two coordinate systems. Now, assume that $\mathbf{X_1}$ lies on the plane defined by:

$$\mathbf{n^T X_1} = h, \tag{7.2}$$

where $\mathbf{n} \in \mathbb{R}^3$ is the plane normal and $h \in \mathbb{R}$ is the distance to the plane. Then, from (7.1) and (7.2), we have:

$$\mathbf{X_2} = (\mathbf{R} + \frac{\mathbf{T}\mathbf{n}^\mathbf{T}}{h})\mathbf{X_1}. \tag{7.3}$$

The images x_1, x_2 of the scene points can then be written as:

$$\lambda \mathbf{x_2} = \mathbf{K}(\mathbf{R} + \frac{\mathbf{T}\mathbf{n}^\mathbf{T}}{h})\mathbf{K}^{-1}\mathbf{x_1} = \mathbf{H}\mathbf{x_1} \tag{7.4}$$

where $\mathbf{x_1}$, $\mathbf{x_2}$ are expressed in homogeneous coordinates as $[x, y, 1]^T$; $\mathbf{K}$ is a 3×3 matrix describing the camera intrinsic parameters; λ is a scalar; $\mathbf{H}$ is a 3×3 matrix called homography that relates the two camera projections of the same plane points.

Since (7.4) is defined up to a scale factor, $\mathbf{H}$ has only eight degrees of freedom. This implies that four corresponding feature pairs (no three collinear) are required to linearly determine $\mathbf{H}$. If more than four points are available, then a least-square solution can be searched. The algorithm we used in our implementation to recover H from a set of consistent point correspondences uses the normalized Direct Linear Transformation (DLT) [11].

Observe that equation (7.4) suggests also a method to check whether a given set of points are coplanar. Namely, if we can select four coplanar corresponding point pairs which are in a sufficiently general configuration, then $\mathbf{H}$ can be computed and used to check whether the other points in the scene lie in the same plane. This is actually the principle of the RANSAC based outlier removal and will be detailed in Section 7.2.5.

Note that matrix $\mathbf{K}$ in equation (7.4) is defined only for perspective cameras. However, in this chapter we assume that our omnidirectional camera is already calibrated and that the image points $\mathbf{x_1}$ and $\mathbf{x_2}$ are already normalized to have the third component equal to 1. This allows us to write $\mathbf{K} = \mathbf{I}$. If not stated otherwise, in the remainder of this chapter we will assume that the image coordinates are always normalized. To calibrate our omnidirectional camera, we used the method described in Chapter 3.

Figure 7.1: The vehicle used in our experiments equipped with the omnidirectional camera (blue circle). The vertical field of view is indicated by the red lines.

7.2.2 Homography or Euclidean Transformation?

In our experiments, we mounted the omnidirectional camera on the roof of the car (Fig. 7.1) with the z-axis of the mirror perpendicular to the ground plane (Fig. 7.2). By fixing the origin of our coordinate system in the center of projection of the omnidirectional camera (Fig. 7.2), we have that $\mathbf{n} = [0, 0, -1]^T$. The distance h of the origin to the ground plane can be manually measured.

According to the last considerations, the homography $\mathbf{H}$ has the form:

$$\begin{aligned}
\mathbf{H} &= \mathbf{R} + \frac{\mathbf{T}\mathbf{n}^{\mathbf{T}}}{h} \\
&= \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} + \frac{1}{h} \begin{bmatrix} t_1 \\ t_2 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}^T \\
&= \begin{bmatrix} \cos\theta & -\sin\theta & -t_1/h \\ \sin\theta & \cos\theta & -t_2/h \\ 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{7.5}$$

where θ is the rotation angle of the camera about the z-axis and t_1 and t_2 are the elements of $\mathbf{T}$.

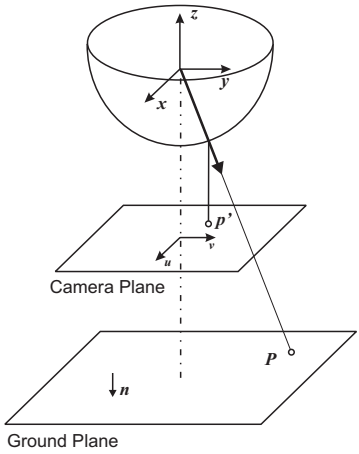


Figure 7.2: The omnidirectional camera model. The axis origin coincides with the single view point of the camera-mirror system. The camera axis is considered to be perpendicular to the ground plane.

Equation (7.5) describes a Euclidean transformation on the image plane, which is a particular case of homography. The Euclidean transformation has only three degrees of freedom and allows us a more stable estimation of the motion with respect to the homography (i.e. eight degrees of freedom) when the motion is constrained to be on the ground plane. However, because of the unavoidable vibrations the camera is subject to during the motion of the vehicle as well as the misalignments among the axes of the camera, mirror, and ground plane normal, the form of $\mathbf{H}$ may appear slightly different from (7.5). Therefore, a eight degrees-of-freedom homography is more appropriate than a Euclidean transformation to describe the relation between the two views.

In the next section, we will see how to decompose the homography to extract $\mathbf{R}$ and $\mathbf{T}$. We will assume that the image coordinates $\mathbf{x_1}$ and $\mathbf{x_2}$ are correctly matched and satisfy the homography constraint (i.e. coplanarity). In section 7.2.5, we will explain how to extract these points.

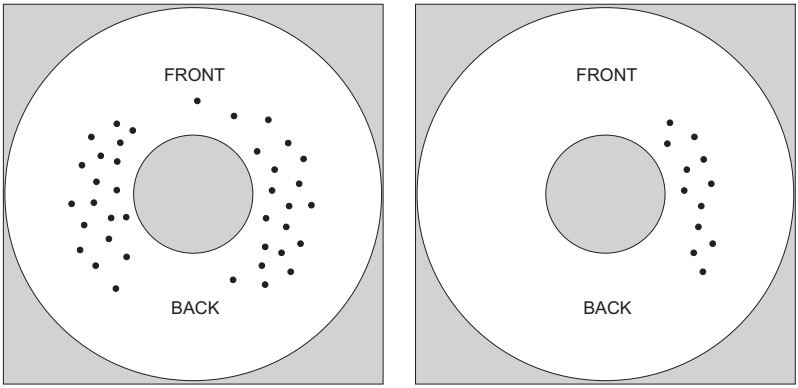


Figure 7.3: (a) Uniform distribution of features. (b) Non uniform distribution.

7.2.3 Decomposing H

If a camera is internally calibrated, it is possible to recover **R**, **T**, and **n** from **H** up to at most a two-fold ambiguity. A linear method for decomposing **H** was originally developed by Wunderlich [124] and later reformulated by Triggs [125]. The algorithm of Triggs is based on the singular value decomposition of **H**. The description of this method as well as its Matlab implementation can be found in [125]. This algorithm outputs two possible solutions for **R**, **T**, and **n** which are all internally self-consistent. In the general case, some false solutions can be eliminated by sign (visibility) tests or geometric constraints, while in our case we can disambiguate the solutions by choosing the one for which the computed plane normal **n** is closer to $[0, 0, 1]^T$. Once the two solutions are disambiguated, the rotation angle θ and the translation parameters t_1, t_2 with respect to the ground plane can be computed. In the remainder of this chapter, we will refer to this method as the "Triggs algorithm".

In our implementation, we used the Triggs algorithm but we combined it also with another method that we are now going to describe. Indeed, the Triggs algorithm works in general very well if the image points are spatially uniformly distributed on the camera image (see Fig. 7.3.a). If the image points are too close to a degenerate configuration or they are spatially distributed within one side of the whole omnidirectional image (Fig. 7.3.b),

then it is better to use the Euclidean approximation given in (7.5).

Here we describe how to use the Euclidean approximation to derive the rotation and translation parameters. From (7.5), we have:

$$\left\{ \begin{array}{rcl} x_2 & = & cx_1 - sy_1 - a \\ y_2 & = & sx_1 + cy_1 - b \end{array} \right. \tag{7.6}$$

with $c = \cos\theta$, $s = \sin\theta$, $a = t_1/h$, $b = t_2/h$, $\mathbf{x_1} = [x_1, y_1]$, and $\mathbf{x_2} = [x_2, y_2]$. Each point pair gives two equations, and hence given two point pairs we can linearly recover c, s, a, b. When more point correspondences are given (say n corresponding pairs) a linear least-squares solution can be found with the pseudo-inverse matrix method. To this end, observe that (7.6) can be rewritten as:

$$\begin{array}{rcl} \left[\begin{array}{cccc} x_1 & -y_1 & -1 & 0 \\ y_1 & x_1 & 0 & -1 \end{array} \right] \cdot \left[\begin{array}{c} c \\ s \\ a \\ b \end{array} \right] & = & \left[\begin{array}{c} x_2 \\ y_2 \end{array} \right] \\ \triangleq \mathbf{A} \cdot \left[\begin{array}{c} c \\ s \\ a \\ b \end{array} \right] & = & \mathbf{B} \end{array} \tag{7.7}$$

where $\mathbf{A}$ is a $2n \times 4$ matrix and $\mathbf{B}$ is a $2n \times 1$ vector. The linear least squares solution of (7.7) is $[c, s, a, b]^T = \mathbf{A}^{+}\mathbf{B}$, where $\mathbf{A}^{+} = (\mathbf{A^T A})^{-1}\mathbf{A^T}$ is the pseudoinverse of $\mathbf{A}$.

Observe that the matrix $\mathbf{Q} = [c, -s; s, c]$ may not be orthonormal because of the method used to compute its coefficients s and c. However, we can compute an orthonormal matrix that better approximate $\mathbf{Q}$.
The best rotation matrix $\mathbf{R_{2D}}$ to approximate $\mathbf{Q}$ in the Frobenius sense is $\mathbf{R_{2D}} = \mathbf{UV^T}$, where $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = SVD(\mathbf{Q})$ and $SVD(\mathbf{Q})$ is the singular value decomposition of $\mathbf{Q}$.
Here, "best" is in the sense of the smallest Frobenius norm of the difference $\mathbf{R_{2D}} - \mathbf{Q}$, which solves the problem:

$$\min_{\mathbf{R_{2D}}} \|\mathbf{R_{2D}} - \mathbf{Q}\|_F^2 \quad \textit{subject to} \quad \mathbf{R_{2D}} \cdot \mathbf{R_{2D}^T} = \mathbf{I} \tag{7.8}$$

Finally, from $\mathbf{R_{2D}}$ the rotation angle θ can be easily computed. At the same time, t_1, t_2 can be directly computed from a and b knowing h.
In the remainder of this chapter, we will refer to this last method as the "Euclidean method".

In the final implementation of our algorithm, we implemented both the Triggs algorithm and the Euclidean method. The trigger condition to use the one or the other is given by the spatial distributions of the image points. If the image points occupy both the left and the right half of the omnidirectional image (like in Fig. 7.3.a), then the Triggs algorithm is used. If the image points are only in one half (i.e. either the left or the right one) of the image (like if Fig. 7.3.b), then the Euclidean method is used.

7.2.4 Maximum likelihood estimation

In 7.2.3, we have described two different approaches to recover the translation parameters and the rotation angle of the vehicle given a set of image correspondences that are assumed to lie on a plane. However, the solution given by both approaches is obtained by a linear method that minimizes an algebraic distance which is not physically meaningful. We can refine it through maximum likelihood inference. The maximum likelihood estimate can be obtained by minimizing the following functional:

$$\min_{\theta,t_1,t_2} \sum_{i=1}^{n} \|\mathbf{x_1}^i - \hat{\mathbf{x}}_\mathbf{1}^i(\theta,t_1,t_2)\|^2 + \|\mathbf{x_2}^i - \hat{\mathbf{x}}_\mathbf{2}^i(\theta,t_1,t_2)\|^2, \tag{7.9}$$

with $\hat{\mathbf{x}}_\mathbf{1} = \mathbf{H^{-1}x_2}$, $\hat{\mathbf{x}}_\mathbf{2} = \mathbf{Hx_1}$. If the Triggs algorithm is used, $\mathbf{H}$ is defined as in [125], else it is defined as in (7.5).

To minimize (7.9), we used the Levenberg-Marquadt algorithm. This algorithm requires an initial guess for θ, t_1, t_2 . As an initial guess, we used the linear solutions provided either by the Triggs algorithm or the Euclidean method.

7.2.5 Coplanarity Check

The equations given in the previous sections assume that the corresponding feature pairs $\mathbf{x_1}$ and $\mathbf{x_2}$ are correctly matched and that the points lie on

the ground plane. Even though in omnidirectional images taken from the roof of the car the ground plane is predominant, there are also many feature points that come from other objects than just the road, like cars, buildings, trees, guardrails, etc. Furthermore, there are also many unavoidable false matches that are more numerous than those usually output by SIFT on standard perspective images (about 20-30% according to [76]) because of the large distortion introduced by the mirror. To discard the outliers, we used the Random Sample Consensus paradigm (RANSAC) [126]. The RANSAC steps in our case are formally the following:

1. Say $\mathbf{A}$ the set of all feature pairs output by SIFT from two consecutive frames. At each iteration, four putative corresponding pairs are randomly selected from $\mathbf{A}$ and a homography $\mathbf{H}$ is instantiated from these points (four is the minimum number of point pairs required to compute a eight degrees-of-freedom homography).
2. The instantiated $\mathbf{H}$ is used to determine the subset $\mathbf{S_1}$ of point pairs in $\mathbf{A}$ that are within some error tolerance d. This subset $\mathbf{S_1}$ is called consensus set.
3. If the number of members in $\mathbf{S_1}$ is greater than some threshold t, which is a function of the expected number of outliers in $\mathbf{A}$, then $\mathbf{S_1}$ is used to compute a new $\mathbf{H}^*$.
4. Otherwise, if the number of members in $\mathbf{S_1}$ is less than t, then a new subset $\mathbf{S_2}$ is randomly selected and the above process is repeated.

If, after some predetermined number of trials, no consensus set with t or more members is found, then the homography $\mathbf{H}^*$ with the largest consensus set is used. The estimation of t as a function of the number of outliers can be found in [126].

As an error measure to determine the subset $\mathbf{S_1}$ of pairs that are within the error tolerance d, we used the symmetric transfer error:

$$err^i = \|\mathbf{x_2^i} - \mathbf{H}\mathbf{x_1^i}\|^\mathbf{2} + \|\mathbf{x_1^i} - \mathbf{H^{-1}}\mathbf{x_2^i}\|^\mathbf{2} \tag{7.10}$$

We reject every pair for which $err^i > d$, where d is computed statistically according to the Huber-type skipped means rule [127], that is $d = 5.2MAD$. M.A.D. stands for Median Absolute Deviation and is defined as:

$$MAD(err) = median_i\{|err_i - median_j(err_j)|\} \tag{7.11}$$

7.3 Visual Compass

In the previous session, we described how to use point features to compute the rotation and translation matrices. Unfortunately, when using features to estimate the motion, the resulting rotation is extremely sensitive to systematic errors due to the intrinsic calibration of the camera or the extrinsic calibration between the camera and the ground plane. This effect is even more accentuated with omnidirectional cameras due to the large distortion introduced by the mirror. In addition to this, integrating rotational information over the time has the major drawback of generally becoming less and less accurate as integration introduces additive errors at each step. An example of camera trajectory recovered only using the feature based approach described in Section 7.2 is depicted in Fig. 7.8.

To improve the accuracy of the rotation estimation, we used an appearance based approach. This approach was inspired by a recent work on the use of omnidirectional cameras as visual compass [128]. Directly using the appearance of the world as opposed to extracting features or structure of the world is attractive because methods can be devised that do not need precise calibration steps. Here, we describe how we implemented our visual compass.

For ease of processing, every omnidirectional image is unwrapped into cylindrical panoramas (Fig. 7.5). The unwrapping considers only the white region of the omnidirectional image that is depicted in Fig 7.6. We call these unwrapped versions "appearances". If the camera is perfectly vertical to the ground, then a pure rotation about its vertical axis will result in a simple column-wise shift of the appearance in the opposite direction. The exact rotation angle could then be retrieved by simply finding the best match between a reference image (before rotation) and a column-wise shift of the successive image (after rotation). The best shift is directly related to the rotation angle undertaken by the camera. In the general motion, translational information is also present. This general case will be discussed later.

The input to our rotation estimation scheme is thus made of appearances that need to be compared. To compare them, we used different distance metrics. In particular, we have tried a cross correlation, a normalized cross correlation, a L_1 norm (Manhattan distance), and an L_2 norm (Euclidean distance). The best results were obtained by using the Euclidean distance. A performance comparison with the other metrics is not given in this chapter.

The Euclidean distance between two appearances I_i and I_j, with I_j being column-wise shifted (with column wrapping) by α pixels, is:

$$d(I_i, I_j, \alpha) = \sqrt{\sum_{k=1}^{h}\sum_{h=1}^{w}\sum_{l=1}^{c} |I_i(k,h,l) - I_j(k,h-\alpha,l)|^2} \tag{7.12}$$

where $h \times w$ is the image size, and c is the number of color components. In our experiments, we used the RGB color space, thus having three color components per pixel.

Being α_m the best shift that minimizes the distance

$$d(I_i, I_j, \alpha_m) \leq d(I_i, I_j, \alpha), \forall \alpha \in \mathbb{R},$$

then the rotation angle $\Delta\vartheta$ (in degrees) between I_i and I_j is:

$$\Delta\vartheta = \alpha_m \cdot \frac{360}{w} \tag{7.13}$$

The width w of the appearance is the width of the omnidirectional image after unwrapping and can be chosen arbitrarily. In our experiments, we used $w = 360$, that means the angular resolution was 1 pixel per degree. To increase the resolution to 0.1 *deg*, we used cubic spline interpolation with 0.1 pixel precision. We also tried larger image widths but we did not get any remarkable improvement in the final results. Thus, we used $w = 360$ as the unwrapping can be done in a negligible amount of time. The Euclidean distance between the two images in Fig. 7.5 as a function of the column-wise shift of the second image is shown in Fig. 7.4.

The distance minimization in (7.12) makes sense only when the camera undergoes a pure rotation about its vertical axis, as a rotation corresponds to a horizontal shift in the appearance. In the real case, the vehicle is moving and translational component is present. However, the "pure rotation" assumption still holds if the camera undergoes small displacements or the distance to the objects (buildings, tree, etc.) is big compared to the displacement. In the other cases, this assumption does not hold for the whole image but an improvement that can be done over the theoretical method is to only consider parts of the images, namely the front and back part (Fig. 7.11). Indeed, the contribution to the optic flow by the motion of the camera is not homogeneous in omnidirectional images; a forward/backward translation mostly contributes in the regions corresponding to the sides of the camera

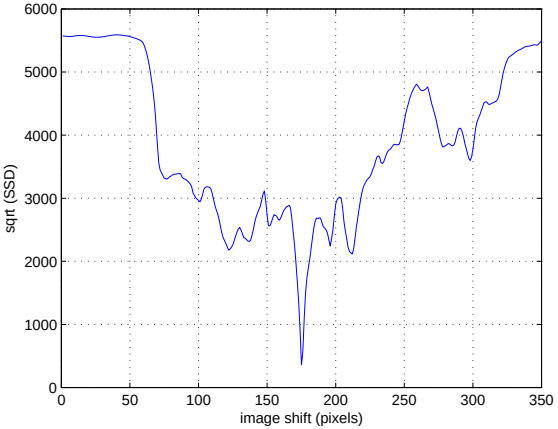


Figure 7.4: The Euclidean distance between the two images in Fig. 7.5 as a function of the column-wise shift of the second image. The distance is computed according to equation (7.12)

and very little in the parts corresponding to the front and back of the camera, while the rotation contributes equally everywhere.

Because we are interested in extracting the rotation information, only considering the regions of the images corresponding to the front and back of the camera allows us to reduce most of the problems introduced by the translation, in particular sudden changes in appearance (parallax).

According to the last considerations, in our experiments we used a reduced Field Of View (FOV) around the front and back of the camera (Fig. 7.6). A reduced field of view of about 30 *deg* around the front part is shown by the white window in Fig. 7.5. Observe that, besides reducing the FOV of the camera in the horizontal plane, we operated a reduction of the FOV in the vertical plane as well, in particular under the horizon line. The reason was to reduce the influence of the changes in appearance of the road. The resulting vertical FOV was 50 *deg* above and 10 *deg* below the horizon line (the horizon line is indicated in red in Fig. 7.5).

The fact of reducing the field of view provided an important improvement over using the whole field of view in terms of stability and sensitivity

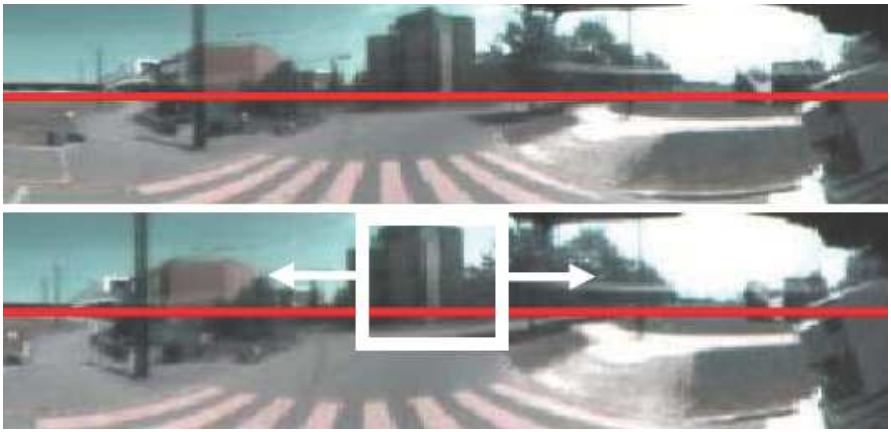


Figure 7.5: Two unwrapped omnidirectional images. For reasons of space, here only one half of the whole 360 *deg* is shown unwrapped. The central part of the image corresponds to the front view of the vehicle (see Fig.7.6). The two frames were taken at different times while the car was translating and turning right. The upper image is taken at time $t-1$, the lower image at time t. The red line is the horizon line. The white box is the search window used in our experiments.

to prominent features at the sides of the camera. The effect of the size of the FOV on the estimation of the camera trajectory is depicted in Fig. 7.10 and will be discussed in Section 7.5.

7.4 Motion Estimation Algorithm

As we already mentioned, the appearance based approach provides rotation angle estimates that are more reliable and stable than those output by the pure feature based approach. Here, we describe how we combined the rotation angle estimates of Section 7.3 with the camera translation estimates of Section 7.2.

In our experiments, the speed of the vehicle ranged between 10 and 20 Km/h while the images were constantly captured at 10 Hz. This means that the distance covered between two consecutive frames ranged between 0.3 and 0.6 meters. For this short distance, the camera configuration (x, y, θ), which

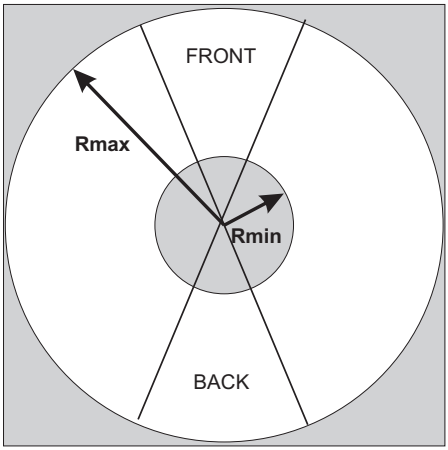


Figure 7.6: The cylindrical panorama is obtained by unwrapping the white region. The front view is the view pointing to the heading direction of the vehicle. The reduced FOV around the front and back of the camera is demarcated by the two lines.

contains its 2D position (x,y) and orientation θ, can be approximated in this way:

$$\begin{cases} x_{i+1} &= x_i &+ \delta\rho_i \cos\theta \\ y_{i+1} &= y_i &+ \delta\rho_i \sin\theta \\ \theta_{i+1} &= \theta_i &+ \delta\theta_i \end{cases} \tag{7.14}$$

where we use $\delta\rho = |\mathbf{T}|\ h$ and $\delta\theta = \Delta\vartheta$.
Observe that $\mathbf{T}$ is the same translation vector used in Section 7.2 and thus $|\mathbf{T}| = norm([t_1, t_2])$, where t_1 and t_2 are computed as described in sections 7.2.3 and 7.2.4. Parameter h is the scale factor (i.e. in our case this is the height of the camera to the ground plane). The camera rotation angle $\Delta\vartheta$ is computed as described in section 7.3. Observe that we did not use at all the rotation estimates provided by the feature based method of section 7.2.

Now, let us resume the steps of our motion estimation scheme, which have been detailed in section 7.2 and 7.3. Our omnidirectional visual odometry operates as follows:

1. Acquire two consecutive frames. Consider only the region of the omnidirectional image, which is between *Rmin* and *Rmax* (Fig. 7.6).

2. Extract and match SIFT features between the two frames. Use the double consistency check to reduce the number of outliers. Then, use the calibrated camera model to normalize the feature coordinates to have the third homogeneous component equal to 1.
3. Use RANSAC to reject points that are not coplanar (section 7.2.5).
4. Apply the linear algorithm described in section 7.2.3 to estimate $\mathbf{R}$ and $\mathbf{T}$ from the remaining inliers. In doing this, switch between the Triggs algorithm and the Euclidean method as described in Section 7.2.3. Then, refine $\mathbf{R}$ and $\mathbf{T}$ using maximum likelihood estimation (section 7.2.4).
5. Unwrap the two images and compare them using the appearance method described in section 7.3. In particular, minimize (7.12), with reduced field of view, to compute the column-wise shift between the appearances and use (7.13) to compute the rotation angle $\Delta\vartheta$.
6. Use $\delta\rho = |\mathbf{T}|\ h$ and $\delta\theta = \Delta\vartheta$ and integrate the motion using (7.14).
7. Repeat from step 1.

7.5 Results

The approach proposed in this chapter has been successfully tested on a real vehicle equipped with a central omnidirectional camera. A picture of our vehicle (a Smart) is shown in Fig 7.1.

Our omnidirectional camera, composed of a hyperbolic mirror (KAIDAN 360 One VR, the same used in the previous chapters) and a digital color camera (SONY XCD-SX910, image size 640×480 pixels), was installed on the front part of the roof of the vehicle. The frames were grabbed at 10 Hz and the vehicle speed ranged between 10 and 20 Km/h.

The resulting path estimated by our visual odometry algorithm using a 10 *deg* FOV is shown in the figures 7.8, 7.11, and 7.12. Our ground truth is an aerial image of the same test environment provided by Google Earth (Fig. 7.11). The units used in the three figures are meters.

In this experiment, the vehicle was driven along a 400 meter long loop and returned to its starting position (pointed to by the yellow arrow in Fig. 7.11). The estimated path is indicated with red dots in Fig. 7.11 and is shown superimposed on a satellite image for comparison. The final error at the loop closure is about 6.5 meters. This error is due to the unavoidable visual odometry drift; however, observe that the trajectory is very well es-

timated until the third 90-degree turn. After this turn, the estimated path deviates smoothly from the expected path instead of continuing straight. After road inspection, we found that this deviation was due to three 0.3 meter tall road humps (pointed to by the cyan arrow in Fig. 7.11) that violate the planar motion assumption.

The content of Fig. 7.12 is very important as it allows us to evaluate the quality of motion estimation. In this figure, we show a textured top viewed 2D reconstruction of the whole path. Observe that this image is not a satellite image but is an image mosaicing. Every input image of this mosaic was obtained by an Inverse Perspective Mapping (IPM) of the original omnidirectional image onto an horizontal plane. This inverse mapping is always possible for central cameras, that is, when a camera has a single effective viewpoint. After being undistorted through IPM, these images have been merged together using the 2D poses estimated by our visual odometry algorithm. The estimated trajectory of the camera is shown superimposed with red dots. If one visually and carefully compares the mosaic (Fig. 7.12) with the corresponding satellite image (Fig. 7.11), it would be possible to recognize in the mosaic the same elements that are present in the satellite image, that is, trees, white footpaths, pedestrian crossings, roads' placement, etc. Furthermore, can be verified that the location of these elements in the mosaic fits well the location of the same elements in the satellite image.

As we mentioned in Section 7.3, we also evaluated the effect of the FOV on the final motion estimation. Fig. 7.10 shows the recovered estimated trajectory respectively using FOV=10 *deg*, FOV=20 *deg*, FOV=30 *deg*, and FOV=60 *deg*. Observe that the estimation of the trajectory improves as the FOV decreases. Indeed, as we mentioned already in Section 7.3, the fact of reducing the field of view allows us to reduce most of the problems introduced by the translation, like sudden changes in parallax. The best performance in terms of closeness to the ground truth of Fig. 7.11 is obtained when FOV=10 *deg*.

In Fig. 7.9, the effect of the FOV on the estimation of the heading direction θ is also shown. Also here, the best performance is when FOV=10 *deg*. In this case in fact 90-degree turns are very well estimated. Furthermore, when FOV=10 *deg* the heading direction stays quite constant after each turn, that is when the vehicle covers a straight path. Note that when the vehicle returns to its start position, the estimated heading direction is equal to 355 *deg*, that means the orientation error at the loop closure is 5 *deg*.

Figure 7.7 shows the rotation angle $\delta\theta$ estimated for FOV=10 *deg* as a function of the traveled distance.
Finally, a comparison of the proposed algorithm with the only feature based method of Section 7.2 is shown in Fig. 7.8.

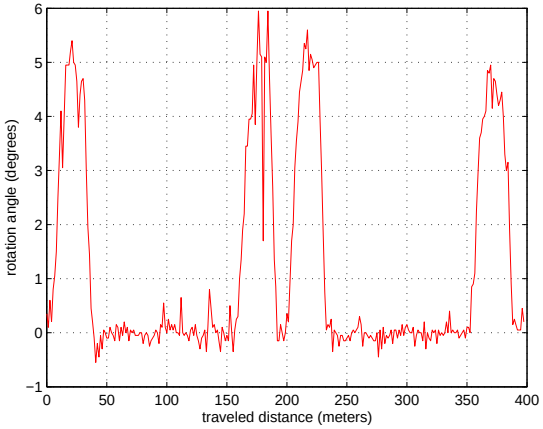


Figure 7.7: The rotation angle $\delta\theta$ (degrees) estimated by the appearance based method vs. the traveled distance (meters)

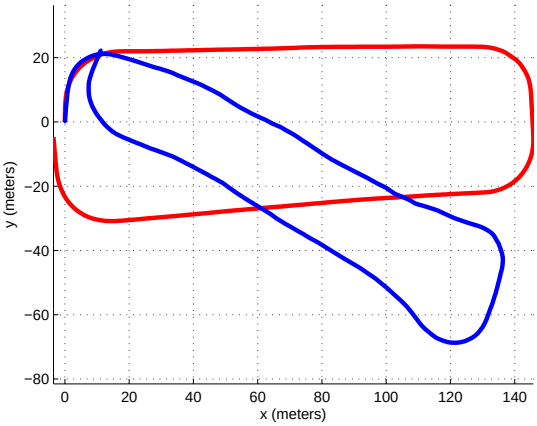


Figure 7.8: A comparison between the camera trajectory recovered through two distinct approaches: in red, the trajectory recovered using the whole algorithm described in this chapter (feature and appearance based); in blue, the trajectory recovered using only the feature based approach described in Section 7.2

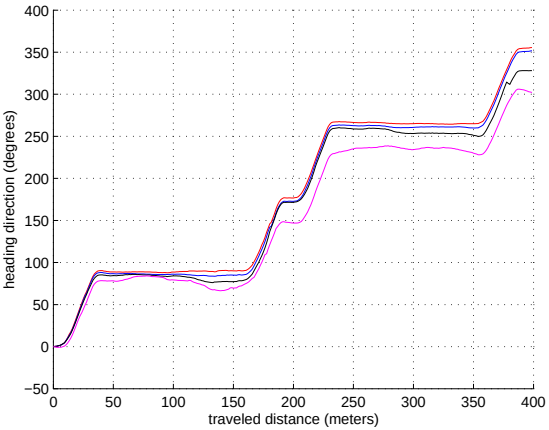


Figure 7.9: The heading direction θ (degrees) vs. the traveled distance (meters). The results are shown for different FOVs: FOV = 10 *deg* (red), FOV = 20 *deg* (blue), FOV = 30 *deg* (black), FOV = 60 *deg* (magenta).

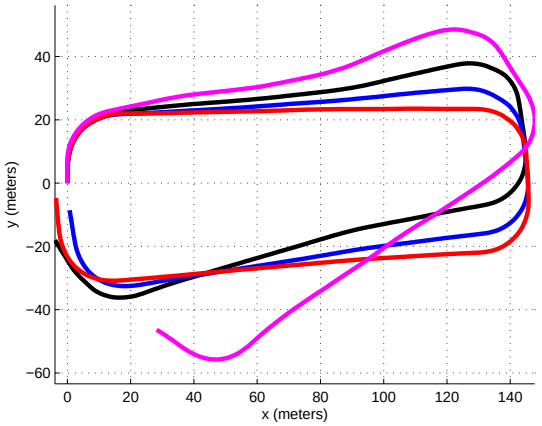


Figure 7.10: A comparison of camera trajectory recovered by using different FOVs: FOV = 10 *deg* (red), FOV = 20 *deg* (blue), FOV = 30 *deg* (black), FOV = 60 *deg* (magenta).

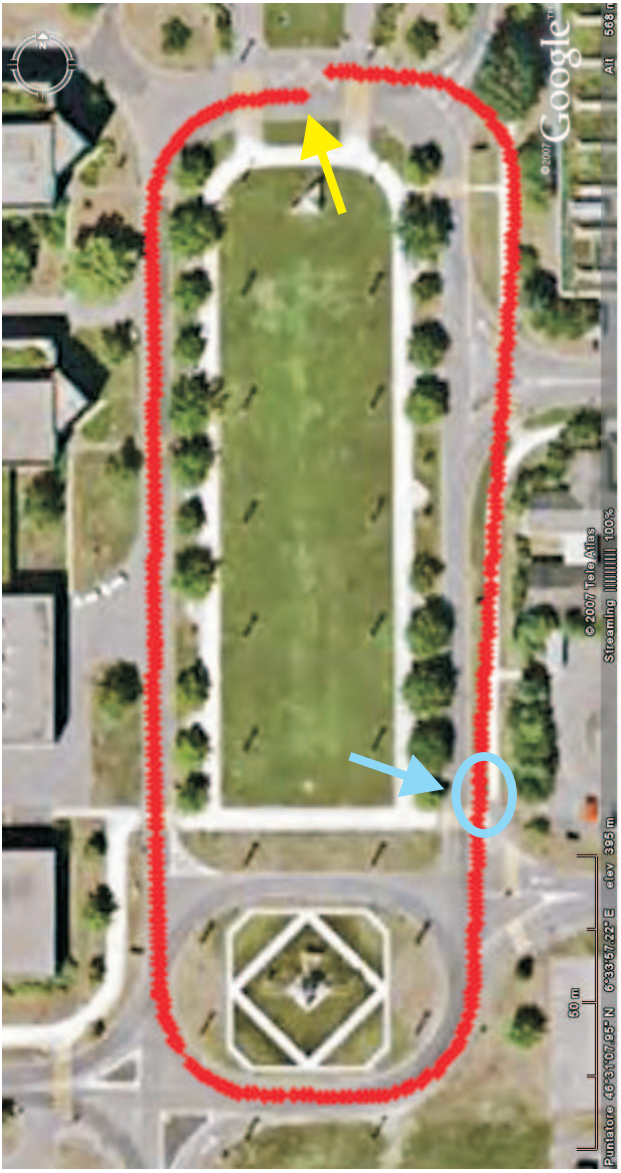


Figure 7.11: The estimated path superimposed onto a Google Earth image of the test environment. The scale is shown at the lower right corner.

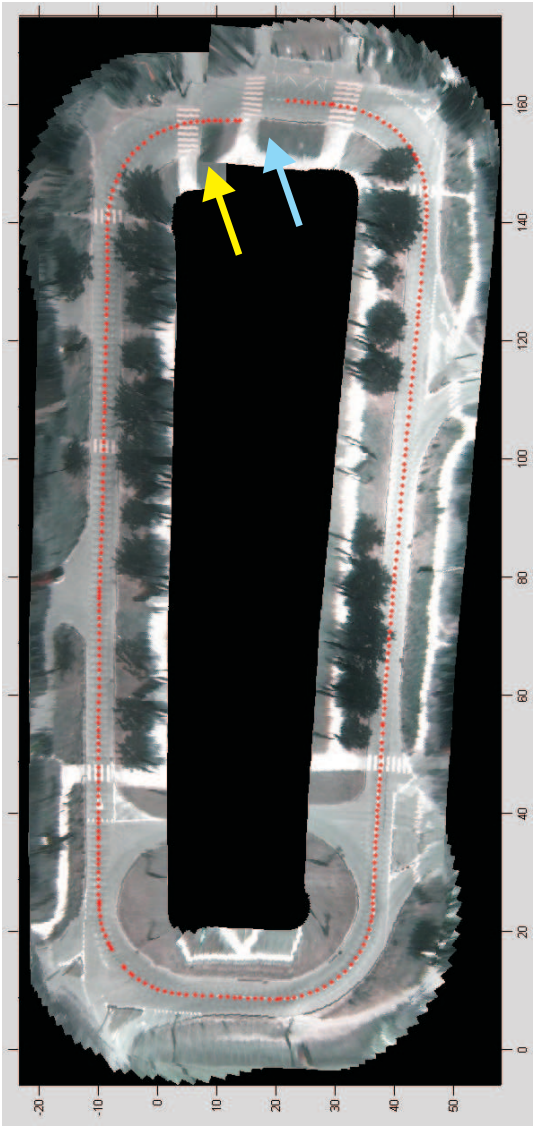


Figure 7.12: Image mosaicing that shows a textured 2D reconstruction of the estimated path. The two arrows point out the final error at the loop closure (the pedestrian crossing pointed to by the cyan arrow should theoretically coincide with that pointed to by the yellow arrow).

7.6 Conclusion

In this chapter, we described an algorithm for computing the ego-motion of a vehicle relative to the road. The algorithm uses as only input images provided by a single omnidirectional camera. The front ends of the system are two different trackers. The first one is a feature based tracker that uses SIFT features and a RANSAC based outlier removal to track the key points that most likely belong to the ground plane. The second one uses an appearance based approach to give high resolution estimates of the rotation angle of the vehicle. Using the first tracker to compute the vehicle displacement in the heading direction and the second tracker to compute the vehicle rotation has proved to give very good visual odometry estimates under planar motion assumption. Furthermore, the performance of the motion estimation given by the proposed method is better than the pure feature based approach (Fig. 7.8).

The proposed algorithm has been successfully applied to videos from an automotive platform. We gave an example of camera trajectory estimated purely from omnidirectional images over a distance of 400 meters. For performance evaluation, the estimated path has been superimposed onto a satellite image of the same test environment and a textured 2D reconstruction of the path has been done.

The appearance based method, with reduced field of view, has proved to give good estimates of the heading direction of the vehicle even in presence of close objects like trees or buildings. Furthermore, straight segments of roads as well as 90 degree turns were very well estimated. The accumulated error after 400 meters (i.e. error at the loop closure) was about 6.5 meters for the distance and 5 degrees for the orientation. This error includes also a smooth deviation caused by some road humps present along the path. Other sources of error are unavoidably due to the integration of the translation and rotation over the time, camera vibrations, or camera calibration. Calibration errors can be intrinsic (e.g. position of the image center, camera-mirror model) or extrinsic (e.g. perpendicularity of the camera axis to the ground plane). Appearance based methods are less sensitive to calibration errors than pure feature based approaches but the sensitivity increases when reducing the field of view (see [128] for a performance analysis). In our implementation, the chosen reduced field of view was a good compromise between accuracy and sensitivity.

However, many improvements can be still implemented. One of the improvements under work is the automatic calibration of the camera during motion and a relaxation of the planar motion assumption.

Chapter 8

Conclusion

This chapter summarizes the contributions of this thesis and future directions of research.

8.1 Summary

For mobile robots to be able to work with and for people and thus operate in our everyday environments, they need to be able to acquire knowledge through perception. In other words they need to collect sensor measurements from which they extract meaningful information. This thesis covered some of the essential components of a robot perception system combining omnidirectional vision, odometry, and 3D laser range finders:

- omnidirectional camera modeling: how to link 3D scene points to their 2D reprojections on the camera image plane,
- omnidirectional camera calibration: how to determine this model for a given camera,
- feature extraction and matching: how to extract and robustly match distinctive line features that are suitable for self-calibration and indoor robot motion estimation,
- calibration between camera and odometry: how to establish the relative pose between the two reference systems,
- calibration between camera and 3D scanner,
- visual odometry: how to recover the trajectory of a car using the visual input alone.

We started by reviewing some of the most used imaging model used to describe central omnidirectional cameras (Chapter 2). We saw that Geyer and Daniilidis proposed a unified imaging model for catadioptric cameras (hyperbolic, parabolic, and elliptic), while Micusik devised several models for dioptric cameras. Using the Micusik's formalism, we reviewed all these imaging models (Section 2.2) and proposed our new unified model (the Taylor model) which encompasses both dioptric and catadioptric cameras (Section 2.3). This model uses a simplified Taylor series expansion whose coefficients need to be estimated by calibration. Camera-mirror-lens misalignments are taken into account by considering an Affine relation between the sensor plane and the camera image plane.

We also devised a flexible and practical methodology to calibrate our unified model (Chapter 3). Our approach takes advantage of a checkerboard-like pattern that is shown by the user at different positions and orientations. The pattern can be freely moved and no a priori knowledge is used. The calibration parameters (that are the Taylor coefficients and the Affine parameters) are estimated using a four-step linear minimization followed by a non-linear refinement which is based on the maximum likelihood criterion. Performance evaluations were done on both simulated and real omnidirectional cameras (Section 3.3). We showed that calibration is robust against image noise and can be improved by increasing the polynomial degree and the number of images (Section 3.3.2). We also applied our calibration procedure to different cameras (dioptric and catadioptric, Section 3.3.2) and showed the accuracy of calibration in a "structure from motion" experiment (Section 3.3.2). We also devised a procedure to detect the center of distortion of the image, which, unlike all previous methods, does not requires the visibility of the mirror's external boundary. This calibration method, led to an opensource toolbox for Matlab, called OCamCalib ((Section 3.4)), made available on the author's web page.

In Chapter 4, we described a method to extract and robustly match vertical lines among omnidirectional images. Matching robustness was achieved by devising a descriptor that is very distinctive for each feature and is invariant to rotation and slight changes of illumination. The performance of the descriptor, which is based on gradient orientation histograms, was shown by tracking features over large camera displacements in several real experiments (Sections 4.7, 5.5.2, 5.6.1, and 5.6.2). Furthermore, this method proved to be very useful and robust for other robotic applications, like camera self-calibration and indoor robot motion estimation (Chapter 5).

In Chapter 5, we faced the problem of extrinsically calibrating an omnidirectional camera with the robot's encoder reference system. Unlike previous methods, we proposed a self-calibration procedure which is based on an Extended Kalman Filter (EKF). The inputs to the EKF are the encoder readings and the bearing angles of line features tracked over the time. By means of the EKF, the parameters characterizing the pose between the two reference systems are estimated while the robot is moving. The proposed strategy was deeply validated through a theoretical analysis, starting from the simpler case of a single feature and extended then to multiple features. In particular, an observability analysis, which takes into account the system nonlinearities, was carried out and clearly showed that the system contains whole the necessary information to estimate the calibration parameters. Furthermore, many accurate simulations and experiments fully validated this strategy (Sections 5.5, 5.6.1). In particular, they showed that by choosing suitable trajectories (alternating straight paths with pure rotations) it is possible to estimate the parameters with high accuracy by moving the robot along very short paths (few meters) (Section 5.5.1).

In Chapter 6, we faced the problem of extrinsically calibrating an omnidirectional camera with a 3D laser range finder. Unlike other previous methods, that use several laser-camera acquisitions, calibration patterns and/or visible laser trace, we devised a method that can be done on the fly. Our method needs a single laser-camera acquisition of a natural scene (either indoor or outdoor) and no calibration pattern. The input points are correspondences that are hand-selected by the user. As we pointed already out, the difficulty resides in visually identifying point correspondences with the laser scan because of the lack of contrast in laser's range images. Our technique consisted in converting the visually ambiguous 3D range information into a 2D map (called Bearing Angle (BA) image) where natural features of a scene are highlighted (Section 6.3). In this way, finding laser-camera correspondences becomes as easy as image pairing. Once correspondence pairs were given, calibration was done using standard methods (Section 6.4.2). Real experiments have been conducted using an omnidirectional camera and a rotating scanner (Section 6.5), but the same approach can be also applied to any other type of camera (both omnidirectional and perspective) or 3D scanner. The results showed that by uniformly selecting the input points from the whole scene, calibration can be done robustly using only a few correspondences.

Finally, in Chapter 7 we faced the problem of visual odometry for outdoor ground vehicles. We devised a method which uses ground point features and

an appearance based method (used as a visual compass) to estimate the ego-motion of an omnidirectional camera mounted on a car under the assumption of planar motion. Experiments were done with a real car. We gave an example of camera trajectory estimated purely from omnidirectional images over a distance of 400 meters. For performance evaluation, the estimated path was superimposed onto a satellite image of the same test environment and a textured 2D reconstruction of the path was also given. The combination of both feature and appearance based methods proved to outperform the standard feature based algorithm by Triggs (Section 7.5, Fig. 7.8). The appearance based method, with reduced field of view, has proved to give very good estimates of the heading direction of the vehicle even in presence of close objects (e.g. trees, buildings, etc). Furthermore, straight segments of roads as well as 90-degree turns were very well estimated. The accumulated error after 400 meters (error at the loop closure) was about 6.5 meters for the distance and 5 degrees for the orientation.

8.2 Outlook

Even if the experiments provided promising results, there are still some aspects that can be improved to provide better performance.

- Our unified Taylor model assumes that camera-mirror (or lens) misalignments are small and thus the relation between the sensor plane and the camera plane can be modeled by an Affine transformation (Section 2.2.3). This assumption is commonly accepted [20, 24] and has proved to be a good approximation of the most complete perspective projection model [26]. However, when misalignments are larger, an homography should be better considered in place of the affinity to take into account the real pose of the two reference systems.

- Our calibration approach uses as a calibration pattern a planar grid that is automatically detected using a chessboard detection algorithm realized on purpose in our lab [39]. However, auto-calibration could be considered by visually tracking several natural keypoints from the environment. The intrinsic parameters could then be estimated using epipolar geometry and RANSAC for outlier removal as in [129].

- Our vertical line tracking algorithm proved to perform very well over large camera displacements (up to two meters, Section 4.7) thanks to the robustness of the descriptor. As we mentioned, our descriptor is

invariant to rotation (line orientation) and slight changes of illumination. Further developments could be considered in the direction of scale invariance. This could be done by selecting the circular areas along the line at different scales. A descriptor could then be assigned for each scale.

- In the contest of camera-odometry extrinsic self-calibration, the following areas deserve further investigations: apply optimal control methods in order to find the best robot trajectory which minimizes the error of the estimated parameters; consider the effect of a systematic component on the odometry; for the omnidirectional camera, introduce other two parameters (e.g. two angles) to take into account the orientation of the mirror axis (that we assumed to be perpendicular to the plane of motion).

- In the area of camera-laser extrinsic calibration, further research should be done to take into account the different resolutions of the laser and of the omnidirectional camera.
  Furthermore, the pose between the two sensors was estimated using standard camera-pose-estimation algorithms which take advantage of point correspondences; further improvements in this area could be obtained using instead line features (although the application of the method would then be limited to structured environments).
  In our method, we considered the camera already calibrated; the method could be extended by refining also the camera intrinsic parameters while estimating the extrinsic ones.

- Our visual odometry algorithm is based on planar motion assumption. To overcome this assumption, one should adopt the full projection model and add the information of 3D scene points instead of the only ground points. Then, using the epipolar geometry constraint (defined by the essential matrix) would be possible to recover the structure and motion using the well known 5 or 8-point algorithm. However, this method does not work when only the ground plane is present because coplanar points are a degenerate configuration for the above algorithms; conversely, a combination of these methods with our approach would certainly overcome this situation.
  Another improvement that could be done is camera self-calibration during the motion of the vehicle; the calibration parameters could be intrinsic (i.e. camera model) and/or extrinsic (e.g. the two angles

defining the orientation of the camera to the ground plane). For this purpose, the solution could be similar to that we used in Chapter 5 where we calibrated the camera with the odometry using an EKF; in this case, the state of the system would consist of the vehicle configuration (i.e. x_R, y_R, θ_R) plus the intrinsic and extrinsic parameters. As a motion model, a constant velocity model could be used.
We have also shown that appearance based methods are less sensitive to calibration errors than pure feature based approaches but the sensitivity increases when reducing the field of view (see [128] for a performance analysis). Further improvements could be obtained by making the size of the field of view adaptive with the observed scene.

Appendix A

Appendix

A.1 Observability analysis (Chapter 5)

We want to show that the gradients $dL^0\beta$, $dL^1_{f_1}\beta$, $dL^1_{f_2}\beta$, $dL^2_{f_1f_2}\beta$ and $dL^2_{f_2f_2}\beta$ are independent. f_1, f_2 and β are defined by the equations (5.24) and (5.11).

We start our proof by computing the five Lie derivatives $L^0\beta$, $L^1_{f_1}\beta$, $L^1_{f_2}\beta$, $L^2_{f_1f_2}\beta$ and $L^2_{f_2f_2}\beta$.

$$L^0\beta = \beta \tag{A.1}$$

$$L^1_{f_1}\beta = \frac{-\rho sin\phi + Dsin\theta}{\gamma} \equiv \frac{a}{\gamma} \tag{A.2}$$

$$L^1_{f_2}\beta = \frac{D\rho cos\theta + D^2}{\gamma} \equiv \frac{b}{\gamma} \tag{A.3}$$

$$L^2_{f_1f_2}\beta = \tag{A.4}$$
$$\frac{Dcos\theta(D^2+\rho^2) - 2\rho^2 Dsin\theta sin\phi + 2\rho D^2 cos\phi}{\gamma^2} \equiv \frac{c}{\gamma}$$

$$L^2_{f_2f_2}\beta = \frac{D\rho sin\theta(D^2-\rho^2)}{\gamma^2} \equiv \frac{d}{\gamma} \tag{A.5}$$

where $\gamma = D^2 + \rho^2 + 2\rho D cos\theta$.

To prove that the gradients of the previous functions are independent we show that the determinant of the matrix whose rows are these gradients is different from zero.

First of all, we remark that only $L^0\beta$ depends on ψ (see equation (5.11)). Therefore this matrix has the following structure:

$$\begin{bmatrix} * & * & * & * & -1 \\ * & * & * & * & 0 \\ * & * & * & * & 0 \\ * & * & * & * & 0 \\ * & * & * & * & 0 \end{bmatrix}$$

Hence, we have to prove that the bottom left submatrix 4×4 has the determinant different from zero. In other words, we can consider only the gradients of the last four functions with respect to D, θ, ϕ, and ρ. Now let us define the following two vectors:

$$\overrightarrow{w} = [a, b, c, d]^T \qquad \overrightarrow{v} = [a, b, 2c, 2d]^T$$

Since $\gamma \neq 0$, the determinant of the previous submatrix is different from zero if and only if is different from zero the following determinant:

$$det = \tag{A.6}$$

$$\left| \frac{\partial \overrightarrow{w}}{\partial D} - \frac{\partial \gamma}{\partial D}\frac{\overrightarrow{v}}{\gamma} \quad \frac{\partial \overrightarrow{w}}{\partial \theta} - \frac{\partial \gamma}{\partial \theta}\frac{\overrightarrow{v}}{\gamma} \quad \frac{\partial \overrightarrow{w}}{\partial \phi} - \frac{\partial \gamma}{\partial \phi}\frac{\overrightarrow{v}}{\gamma} \quad \frac{\partial \overrightarrow{w}}{\partial \rho} - \frac{\partial \gamma}{\partial \rho}\frac{\overrightarrow{v}}{\gamma} \right|$$

On the other hand, through a direct computation it is possible to prove that

$$\left| \frac{\partial \overrightarrow{w}}{\partial D}, \quad \frac{\partial \overrightarrow{w}}{\partial \theta}, \quad \frac{\partial \overrightarrow{w}}{\partial \phi}, \quad \frac{\partial \overrightarrow{w}}{\partial \rho} \right| \neq 0$$

and therefore it is possible to express the vector $\overrightarrow{v}$ as a combination of the four vectors: $\frac{\partial \overrightarrow{w}}{\partial D}$, $\frac{\partial \overrightarrow{w}}{\partial \theta}$, $\frac{\partial \overrightarrow{w}}{\partial \phi}$ and $\frac{\partial \overrightarrow{w}}{\partial \rho}$.

By using this combination for $\overrightarrow{v}$ in (A.6) it is possible to show that also the determinant in (A.6) is different from zero.

Bibliography

[1] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A flexible technique for accurate omnidirectional camera calibration and structure from motion," in *IEEE International Conference on Computer Vision Systems (ICVS 2006)*, jan 2006.

[2] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easy calibrating omnidirectional cameras," in *IEEE International Conference on Intelligent Robots and Systems (IROS 2006)*, oct 2006.

[3] D. Scaramuzza and R. Siegwart, *Vision Systems and Applications, book chapter "A Practical Toolbox for Calibrating Omnidirectional Cameras"*. I-Tech Education and Publishing, Vienna, June 2007.

[4] D. Scaramuzza and R. Siegwart, "A new method and toolbox for easily calibrating omnidirectional cameras," in *International Conference on Computer Vision Systems (ICVS 2007), Workshop on Camera Calibration Methods for Computer Vision Systems (CCMCVS'07)*, march 2007.

[5] D. Scaramuzza, "Ocamcalib toolbox: a matlab calibration toolbox omnidirectional cameras." Google for "*ocamcalib*".

[6] D. Scaramuzza, A. Martinelli, and R. Siegwart, "Precise bearing angle measurement based on omnidirectional conic sensor and defocusing," in *European Conference on Mobile Robots (ECMR 2005)*, sep 2005.

[7] A. Martinelli, D. Scaramuzza, and R. Siegwart, "Automatic self-calibration of a vision system during robot motion," in *IEEE International Conference on Robotics and Automation (ICRA 2006)*, May 2006.

[8] D. Scaramuzza, N. Criblez, A. Martinelli, and R. Siegwart, *Field and Service Robotics: results of the 6th international conference (FSR*

2007), title of the chapter: Robust Feature Extraction and Matching for Omnidirectional Images. Springer-Verlag, STAR, 2008.

[9] D. Scaramuzza, A. Harati, and R. Siegwart, "Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes," in *IEEE International Conference on Intelligent Robots and Systems (IROS 2007)*, 2007.

[10] D. Scaramuzza and R. Siegwart, "Monocular omnidirectional visual odometry for outdoor ground vehicles," in *International Conference on Computer Vision Systems (ICVS 2008), Ed. Springer*, May 2008.

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

[12] R. Descartes and D. Smith, *The geometry of Renè Descartes*. Dover, New York, originally published in Discours de la Methode, 1637.

[13] R. P. Feynman, R. Leighton, and M. Sands, *The Feynman Lectures on Physics*. Addison-Wesley, 1 ed., 1963.

[14] E. Hecht and A. Zajac, *Optics*. Addison Wesley, 1974.

[15] S. Nayar, "Catadioptric omnidirectional camera," in *IEEE CVPR'97*, pp. 482–488, jun 1997.

[16] S. Baker and S. Nayar, "A theory of catadioptric image formation," in *ICCV'98*, pp. 35–42, 1998.

[17] S. Baker and S. Nayar, "A theory of single-viewpoint catadioptric image formation," *International Journal of Computer Vision (IJCV)*, vol. 35, no. 2, pp. 175–196, 1999.

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

[19] X. Ying and Z. Hu, "Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model?," in *European Conference on Computer Vision (ECCV)*, pp. 442–355, Lecture Notes in Computer Science, Springer Verlag, may 2004.

[20] B. Micusik, *Two View Geometry of Omnidirectional Cameras.* PhD thesis, Center for Machine Perception, Czech Technical University in Prague, 2004.

[21] T. Svoboda, T. Pajdla, and V. Hlavac, "Epipolar geometry for panoramic cameras," in *European Conference on Computer Vision (ECCV)*, pp. 218–232, Lecture Notes in Computer Science, Springer Verlag, 1998.

[22] T. Svoboda and T. Pajdla, "Epipolar geometry for central catadioptric cameras," *International Journal of Computer Vision (IJCV)*, vol. 49, pp. 23–37, aug 2002.

[23] P. Sturm, "Mixing catadioptric and perspective cameras," in *IEEE Workshop on Omnidirectional Vision*, pp. 60–67, 2002.

[24] B. Micusik and T. Pajdla, "Estimation of omnidirectional camera model from epipolar geometry," in *CVPR*, jun 2003.

[25] J. Kumler and M. Bauer, "Fish-eye lens designs and their relative performance," in *International Society for Optical Engineering (SPIE) - Current Developments in Lens Design and Optical Systems Engineering*, vol. 4093, pp. 360–369, oct 2000.

[26] O. Frank, R. Katz, C. Tisse, and H. Durrant-Whyte, "Camera calibration for miniature, low-cost, wide-angle imaging systems," in *British Machine Vision Conference (BMVC07)*, 2007.

[27] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on PAMI*, vol. 22, no. 11, 2000.

[28] C. Cauchois, E. Brassart, L. Delahoche, and T. Delhommelle, "Reconstruction with the calibrated syclop sensor," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS2000)*, pp. 1493–1498, 2000.

[29] H. Bakstein and T. Pajdla, "Panoramic mosaicing with a 180° field of view lens," in *IEEE Workshop on Omnidirectional Vision*, pp. 60–67, 2002.

[30] C. Geyer and K. Daniilidis, "Paracatadioptric camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 687–695, may 2002.

[31] J. Gluckman and S. Nayar, "Ego-motion and omnidirectional cameras," in *6th International Conference on Computer Vision*, pp. 999–1005, 1998.

[32] S. Kang, "Catadioptric self-calibration," in *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 201–207, 2000.

[33] B. Micusik and T. Pajdla, "Para-catadioptric camera auto-calibration from epipolar geometry," in *Asian Conference on Computer Vision*, 2004.

[34] S. Bougnoux, "From projective to euclidean space under any practical situation, a criticism of self-calibration," in *6th International Conference on Computer Vision*, pp. 790–796, 1998.

[35] J. Barreto and H. Araujo, "Geometric properties of central catadioptric line images and their application in calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1327–1333, aug 2005.

[36] P. Sturm and S. Ramaligam, "A generic concept for camera calibration," in *European Conference on Computer Vision (ECCV2004)*, 2004.

[37] C. Mei and P. Rives, "Calibrage non biaise d'un capteur central catadioptrique," in *RFIA*, January 2006.

[38] C. Mei and P. Rives, "Single view point omnidirectional camera calibration from planar grids," in *IEEE International Conference on Robotics and Automation*, April 2007.

[39] M. Rufli, "Automatic chessboard detection in blurred and very distorted images," tech. rep., ETH, Master semeter project, dec 2007.

[40] C. Mei, "Matlab calibration toolbox for omnidirectional cameras." *http : //www.robots.ox.ac.uk/ cmei/Toolbox.html*.

[41] K. Levenberg, "A method for the solution of certain problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[42] D. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *SIAM Journal on Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[43] H. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, pp. 133–135, sep 1981.

[44] R. I. Hartley, "In defence of the 8-point algorithm," in *IEEE International Conference on Computer Vision (ICCV'95)*, 1995.

[45] C. Harris and M. Stephens, "A combined corner and edge detector," in *Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[46] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *In Proc. 13th British Machine Vision Conference*, pp. 384–393, 2002.

[47] T. Kadir, A. Zisserman, and M. Brady, "An affine invariant salient region detector," in *7th European Conference on Computer Vision*, 2004.

[48] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *8th International Conference on Computer Vision*, pp. 525–531, 2001.

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

[50] T. Lindeberg, "Feature detection with automatic scale selection," *International Journal of Computer Vision*, no. 2, pp. 79–116, 1998.

[51] K. Mikolajczyk and C. Schmid, "An affine invariant interest point detector," in *7th European Conference on Computer Vision*, 2002.

[52] A. Baumberg, "Reliable feature matching across widely separated views," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[53] T. Tuytelaars and L. Van Gool, "Matching widely separated views based on affine invariant regions," *International Journal of Computer Vision*, vol. 1, no. 59, pp. 61–85, 2004.

[54] T. Mauthner, F. Fraundorfer, and H. Bischof, "Region matching for omnidirectional images using virtual camera planes," in *Computer Vision Winter Workshop 2006*, 2006.

[55] B. Micusik and T. Pajdla, "Structure from motion with wide circular field of view cameras," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 1135–1149, July 2006.

[56] A. Briggs, Y. Li, D. Scharstein, and M. Wilder, "Robot navigation using 1d panoramic images," in *IEEE International Conference on Robotics and Automation (ICRA 2006)*, 2006.

[57] G. Medioni and R. Nevatia, "Segment-based stereo matching," *Computer Vision, Graphics and Image Processing*, vol. 31, pp. 2–18, 1985.

[58] N. Ayache, *Stereovision and Sensor Fusion*. MITPress, 1990.

[59] Z. Zhang, "Token tracking in a cluttered scene," *Image and Vision Computing*, vol. 12, no. 2, pp. 110–120, 1994.

[60] J. Crowley and P. Stelmazyk, "Measurement and integration of 3d structures by tracking edge lines," in *ECCV'90*, 1990.

[61] R. Deriche and O. Faugeras, "Tracking line segments," in *ECCV*, pp. 259–267, 1990.

[62] D. Huttenlocher, G. Klanderman, and W. Rucklidge, "Comparing images using the hausdorff distance," *IEEE Transactions on Pattern analysis and Machine Intelligence*, 1993.

[63] N. Ayache and O. Faugeras, "Building a consistent 3d representation of a mobile robot environment by combining multiple stereo views," in *IJCAI*, 1987.

[64] R. Horaud and T. Skordas, "Stereo correspondence through feature grouping and maximal cliques," *IEEE Transactions on Pattern analysis and Machine Intelligence*, vol. 11, no. 11, pp. 1168–1180, 1989.

[65] P. Gros, "Matching and clustering: Two steps towards object modelling in computer vision," *International Journal of Robotics Research*, vol. 14, no. 6, pp. 633–642, 1995.

[66] V. Venkateswar and R. Chellappa, "Hierarchical stereo and motion correspondence using feature groupings," *International Journal of Computer Vision*, pp. 245–269, 1995.

[67] R. Gonzalez and R. Woods, *Digital Image Processing*. Prentice Hall, 2 ed., 2002.

[68] C. Baillard, C. Schmid, A. Zisserman, and A. Fitzgibbon, "Automatic line matching and 3d reconstruction of buildings from multiple views," in *IAPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery*, 1999.

[69] D. Tell and S. Carlsson, "Wide baseline point matching using affine invariants computed from intensity profiles," in *European Conference Computer Vision*, pp. 814–828, 2000.

[70] Y. Yagi and M. Yachida, "Real-time generation of environmental map and obstacle avoidance using omnidirectional image sensor with conic mirror," in *CVPR'91*, 1991.

[71] E. Brassart, L. Delahoche, C. Cauchois, C. Drocourt, C. Pegard, and E. Mouaddib, "Experimental results got with the omnidirectional vision sensor: Syclop," in *International workshop on omnidirectional vision (OMNIVIS 2000)*, 2000.

[72] D. Prasser, G. Wyeth, and M. Milford, "Experiments in outdoor operation of ratslam," in *Australian Conference on Robotics and Automation*, 2004.

[73] M. Rahman, P. Bhattacharya, and B. Desai, "Similarity searching in image retrieval with statistical distance measure and supervised learning," in *International Conference on Advances in Pattern Recognition (ICAPR)*, 2005.

[74] Y. Rubner, C. Tomasi, and L. Guibas, "The earth mover's distance as a metric for image retrieval," *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.

[75] Y. Rubner, J. Puzicha, C. Tomasi, and J. Buhmann, "Empirical evaluation of dissimilarity measures for color and texture," *Computer Vision and Image Understanding*, vol. 84, no. 1, pp. 25–43, 2001.

[76] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 20, pp. 91–110, 2003.

[77] A. Martinelli, F. Pont, and S. R., "Multi-robot localization using relative observations," in *International Conference on Robotics and Automation*, apr 2005.

[78] A. Mourikis and S. Roumeliotis, "Optimal sensing strategies for mobile robot formations: Resource-constrained localization," in *Robotics: Science and Systems*, jun 2005.

[79] I. A., *Nonlinear Control Systems*. Springer Verlag, 3 ed., 1995.

[80] Y. Bar-Shalom and T. Fortmann, "Tracking and data association," *mathematics in science and engineering*, vol. 179, 1988.

[81] R. Hermann and A. Krener, "Nonlinear controllability and observability," *IEEE Transaction On Automatic Control*, vol. 22, no. 5, pp. 728–740, 1977.

[82] A. Bicchi, D. Pratichizzo, A. Marigo, and A. Balestrino, "On the observability of mobile vehicles localization," in *IEEE Mediterranean Conference on Control and Systems*, 1998.

[83] P. Bonnifait and G. Garcia, "Design and experimental validation of an odometric and goniometric localization system for outdoor robot vehicles," *IEEE Transactions On Robotics and Automation*, vol. 14, aug 1998.

[84] K. Chong and L. Kleeman, "Accurate odometry and error modelling for a mobile robot," in *International Conference on Robotics and Automation*, vol. 4, pp. 2783–2788, 1997.

[85] A. Martinelli, N. Tomatis, A. Tapus, and R. Siegwart, "Simultaneous localization and odometry calibration," in *International Conference on Inteligent Robot and Systems (IROS03)*, 2003.

[86] D. Haehnel, W. Burgard, and S. Thrun, "Learning compact 3d models of indoor and outdoor environments with a mobile robot," *Robotics and Autonomous Systems*, vol. 44, no. 1, pp. 15–27, 2003.

[87] J. Weingarten and R. Siegwart, "3d slam using planar segments," in *IROS06*, pp. 9–15, October 2006.

[88] O. Wulf, K. Arras, H. Christensen, and B. Wagner, "2d mapping of cluttered indoor environments by means of 3d perception," in *ICRA04*, pp. 4204–4209, April 2004.

[89] K. Pervoelz, A. Nuechter, H. Surmann, and J. Hertzberg, "Automatic reconstruction of colored 3d models," in *Robotik*, 2004.

[90] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder *improvescameraintrinsiccalibration*," in *IEEE IROS04*, 2004.

[91] C. Mei and P. Rives, "Calibration between a central catadioptric camera and a laser range finder for robotic applications," in *ICRA06*, May 2006.

[92] S. Wasielewski and O. Strauss, "Calibration of a multi-sensor system laser rangefinder/camera," in *Intelligent Vehicles' Symposium*, pp. 472–477, 1995.

[93] O. Jokinen, "Self-calibration of a light striping system by matching multiple 3-d profile maps," in *2nd International Conference on 3D Digital Imaging and Modeling*, pp. 180–190, 1999.

[94] D. Cobzas, H. Zhang, and M. Jaegersand, "A comparative analysis of geometric and image-based volumetric and intensity data registration algorithms," in *IEEE ICRA02*, 2002.

[95] R. Unnikrishnan and M. Hebert, "Fast extrinsic calibration of a laser rangefinder to a camera," tech. rep., Robotics Institute, Carnegie Mellon University, July 2005.

[96] R. Unnikrishnan, "Laser-camera calibration toolbox for matlab." *http : //www.cs.cmu.edu/ ranjith/lcct.html*.

[97] *http : //www.m − elrob.eu/*.

[98] P. Lamon, C. Stachniss, R. Triebel, P. Pfaff, C. Plagemann, G. Grisetti, S. Kolsky, W. Burgard, and R. Siegwart, "Mapping with an autonomous car," in *IEEE/RSJ IROS'06, Workshop: Safe Navigation in Open and Dynamic Environments*, 2006.

[99] J. Weingarten, *Feature-based 3D SLAM*. PhD thesis, Swiss Federal Institute of Technology Lausanne, EPFL, 2006.

[100] K. Pulli, "Vision methods for an autonomous machine based on range imaging," 1993.

[101] O. Bellon and L. Silva, "New improvements to range image segmentation by edge detection," *IEEE Signal Processing Letters*, vol. 9, no. 2, pp. 43–45, 2002.

[102] L. Quan and Z. Lan, "Linear n-point camera pose determination," *IEEE Transactions on PAMI*, vol. 21, 1999.

[103] X. Gao, X. Hou, J. Tang, and H. Chen, "Complete solution classification for the perspective-three-point problem," *IEEE Transactions on PAMI*, vol. 25, no. 8, pp. 930–943, 2003.

[104] Z. Zhang, "Iterative point matching for registration of free-form curves," Tech. Rep. 1658, INRIA-Sophia Antipolis, 1992.

[105] *http : //www.spheron.com.*

[106] H. Moravec, *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover.* PhD thesis, Stanford University, 1980.

[107] S. Lacroix, A. Mallet, R. Chatila, , and L. Gallo, "Rover self localization in planetary-like environments," in *International Symposium on Articial Intelligence, Robotics, and Automation for Space (i-SAIRAS)*, pp. 433–440, 1999.

[108] I. Jung and S. Lacroix, "Simultaneous localization and mapping with stereovision," in *Robotics Research: the 11th International Symposium*, 2005.

[109] D. Nister, O. Naroditsky, , and B. J., "Visual odometry for ground vehicle applications," *Journal of Field Robotics*, 2006.

[110] M. Maimone, Y. Cheng, and L. Matthies, "Two years of visual odometry on the mars exploration rovers: Field reports," *Journal of Field Robotics*, vol. 24, no. 3, pp. 169–186, 2007.

[111] P. I. Corke, D. Strelow, , and S. Singh, "Omnidirectional visual odometry for a planetary rover," in *IROS*, 2004.

[112] M. C. Deans, *Bearing-Only Localization and Mapping.* PhD thesis, Carnegie Mellon University, 2002.

[113] A. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *International Conference on Computer Vision*, 2003.

[114] L. A. Clemente, A. J. Davison, I. Reid, J. Neira, and J. D. Tardos, "Mapping large loops with a single hand-held camera," in *Robotics Science and Systems*, 2007.

[115] T. Lemaire and S. Lacroix, "Slam with panoramic vision," *Journal of Field Robotics*, vol. 24, no. 1-2, pp. 91–111, 2007.

[116] H. Wang, K. Yuan, W. Zou, and Q. Zhou, "Visual odometry based on locally planar ground assumption," in *IEEE International Conference on Information Acquisition*, pp. 59–64, 2005.

[117] Q. Ke and T. Kanade, "Transforming camera geometry to a virtual downward-looking camera: Robust ego-motion estimation and ground-layer detection," in *CVPR 2003*, jun 2003.

[118] J. Guerrero, R. Martinez-Cantin, and C. Sagues, "Visual map-less navigation based on homographies," *Journal of Robotic Systems*, vol. 22, no. 10, pp. 569–581, 2005.

[119] B. Liang and N. Pears, "Visual navigation using planar homographies," in *IEEE ICRA*, pp. 205–210, 2002.

[120] P. Hansen, P. Corke, W. Boles, and K. Daniilidis, "Scale invariant feature matching with wide angle images," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*, November 2007.

[121] R. Tsai and T. Huang, "Estimating three-dimensional motion parameters of a rigid planar patch," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 29, no. 6, pp. 1147–1152, 1981.

[122] H. C. Longuet-Higgins, "The reconstruction of a plane surface from two perspective projections," in *Royal Society London*, vol. 277, pp. 399–410, 1986.

[123] O. D. Faugeras and F. Lustman, "Motion and structure from motion in a piecewise planar environment," *International Journal of Pattern Recognition and Artificial Inteligence*, no. 3, pp. 485–508, 1988.

[124] W. Wunderlich, "Rechnerische rekonstruktion eines ebenen objekts aus zwei photographien," tech. rep., Mitteilungen der geodaetischen Institute, TU Graz, 1982.

[125] B. Triggs, "Autocalibration from planar scenes," in *ECCV98*, 1998.

[126] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[127] F. Hampel, "The influence curve and its role in robust estimation," *Journal of American Statistics Association*, vol. 69, pp. 383–393, 1974.

[128] F. Labrosse, "The visual compass: performance and limitations of an appearance-based method," *Journal of Field Robotics*, vol. 23, no. 10, pp. 913–941, 2006.

[129] B. Micusik and T. Pajdla, "Autocalibration and 3d reconstruction with non-central catadioptric cameras," in *IEEE Internation Conference on Computer Vision and Pattern Recognition*, 2004.

Curriculum Vitæ

Biography

Davide Scaramuzza was born April 2 1980 in Terni, Italy.

In July 2004, he received his Laurea degree (equivalent to a M.S.) in Electronic Engineering at the University of Perugia.
He got the final grade of 110/110 Summa cum Laude and "dignity of printing".
The title of his Master dissertation was: *Design and realization of a Stereoscopic Vision System for Robotics, with applications to tracking of moving objects and self-localization.* His Master thesis supervisor was Prof. Dr. Paolo Valigi.

From October 2004 to February 2005, he did an internship at the Roland Siegwart's Autonomous Systems Lab (ASL) at the Swiss Federal Institute of Technology of Lausanne (EPFL).

From March 2005 to July 2006, he was PhD candidate at the EPFL and he finally moved to the ETH of Zurich to follow his supervisor Prof. Dr. Roland Siegwart.

From July 2006 to present, he has been PhD candidate at the D-MAVT department of the ETH under the supervision of Prof. Siegwart.

Since 2004 he has been member of the IEEE community.

Awards

1. "AICA-FEDERCOMIN prize": In June 2005, he received the first Italian national prize for the best Master Thesis of the year 2004 in the field of Information Communication and Technology (ICT). The prize was offered by AICA-FEDERCOMIN, the co-association of all Italian companies in the field of ICT. The prize was given by the Italian Minister of Innovation and Technology Luciano Stanca. The prize amount was 5,000 Euros.

2. "TERNANA OPERA EDUCATRICE (TOE) prize": In February 2005, he received the first prize as the best graduate student of his home town Terni in the year 2004. The prize was offered by TOE that is a foundation giving grants for education. TOE was founded in memory of Elia Rossi Passavanti, Italian patriot of the second world war and lover of education. The prize amount was 800 Euros.

List of Publications

Book Chapters

- **Scaramuzza, D.**, Criblez, N., Martinelli, A. and Siegwart, R., *Robust Feature Extraction and Matching for Omnidirectional Images*, in Field and Service Robotics: results of the 6th international conference, Springer-Verlag, STAR, ISBN: 978-3-540-75403-9, 2008.
- **Scaramuzza, D.** and Siegwart, R. (2007), *A Practical Toolbox for Calibrating Omnidirectional Cameras*, in Vision Systems Applications, ISBN: 978-3-902613-01-1, edited by Prof. Goro Obinata and Ashish Dutta, publisher I-Tech Education and Publishing, Vienna, Austria, June, 2007.

Peer-reviewed Proceedings

- **Scaramuzza, D.** ad Siegwart R., Monocular Omnidirectional Visual Odometry for Outdoor Ground Vehicles, in International Conference on Computer Vision Systems (ICVS 2008), Santorini, Greece, 2008.

- Ramisa, A., Vasudevan S., **Scaramuzza, D.**, De Mantaras, R.L., and Siegwar R., A tale of two object recognition methods for mobile robots, in International Conference on Computer Vision Systems (ICVS 2008), Santorini, Greece, 2008.
- **Scaramuzza, D.**, Harati, A., and Siegwart, R., Extrinsic Self Calibration of a Camera and a 3D Laser Range Finder from Natural Scenes, in IEEE International Conference on Intelligent Robots and Systems (IROS 2007), San Diego, USA, October, 2007.
- **Scaramuzza, D.**, Martinelli, A. and Siegwart, R., A Toolbox for Easy Calibrating Omnidirectional Cameras, in IEEE International Conference on Intelligent Robots and Systems (IROS 2006), Beijing, China, October, 2006.
- Martinelli, A., **Scaramuzza, D.**, Siegwart, R., Automatic Self-Calibration of a Vision System during Robot Motion, in IEEE International Conference on Robotics and Automation (ICRA 2006), Orlando, USA, May, 2006.
- **Scaramuzza, D.**, Martinelli, A. and Siegwart, R., A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion, in IEEE International Conference on Computer Vision Systems (ICVS 2006), New York, USA, January 2006.
- **Scaramuzza, D.**, Martinelli, A. and Siegwart, R., Precise Bearing Angle Measurement Based on Omnidirectional Conic Sensor and Defocusing, in European Conference on Mobile Robots (ECMR 2005), Ancora, Italy, September, 2005.
- **Scaramuzza, D.**, Pagnottelli S. and Valigi P., Ball Detection and Predictive Ball Following Based on a Stereoscopic Vision System, in IEEE International Conference on Robotics and Automation (ICRA 2005), Barcelona, Spain, April, 2005.

Workshops

- **Scaramuzza, D.** and Siegwart, R., A New Method and Toolbox for Easily Calibrating Omnidirectional Cameras, in International Conference on Computer Vision Systems (ICVS 2007), Workshop on Camera Calibration Methods for Computer Vision Systems (CCMCVS'07), Bielefeld, Germany, March, 2007.