UNIVERSIDADE DE SÃO PAULO—USP
ESCOLA DE ENGENHARIA DE SÃO CARLOS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Ana Rita Pereira

**Odometria visual: comparando métodos
estéreo e multi-câmera**

São Carlos
2017

Ana Rita Pereira

**Visual odometry: comparing a stereo
and a multi-camera approach**

Dissertação apresentada à Escola de Engenharia de São Carlos da Universidade de São Paulo como parte dos requisitos para a obtenção do título de Mestre em Ciências, Programa de Engenharia Elétrica.

Área de concentração: Sistemas Dinâmicos

Orientador: Prof. Dr. Valdir Grassi Júnior
São Carlos

2017

Trata-se da versão corrigida da Dissertação. A versão original se encontra disponível na EESC/USP que aloja o Programa de Pós-Graduação de Engenharia Elétrica.

AUTORIZO A REPRODUÇÃO TOTAL OU PARCIAL DESTE TRABALHO,
POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS
DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

P532o Pereira, Ana Rita Fernandes Lobo
 Odometria visual: comparando métodos estéreo e
 multi-câmera / Ana Rita Fernandes Lobo Pereira;
 orientador Valdir Grassi Júnior. São Carlos, 2017.

 Dissertação (Mestrado) - Programa de Pós-Graduação
 em Engenharia Elétrica e Área de Concentração em
 Sistemas Dinâmicos -- Escola de Engenharia de São
 Carlos da Universidade de São Paulo, 2017.

 1. visão computacional. 2. odometria visual. 3.
 visão estéreo. 4. visão omnidirecional. 5. múltiplas
 câmeras. I. Título.

FOLHA DE JULGAMENTO

Candidata: **ANA RITA FERNANDES LOBO PEREIRA**

Título da dissertação: "Odometria visual: comparando métodos estéreo e multi-câmera"

Data da defesa: 25.07.2017

Comissão Julgadora:

Resultado:

Prof. Dr. **Valdir Grassi Junior**
(Orientador)
(Escola de Engenharia de São Carlos/EESC)

Aprovada

Prof. Associado **Jun Okamoto Junior**
(Escola Politécnica/EP-USP)

APROVADO

Profª. Drª. **Raquel Frizera Vassallo**
(Universidade Federal do Espírito Santo/UFES)

Aprovada

Coordenador do Programa de Pós-Graduação em Engenharia Elétrica:
Prof. Associado **Luis Fernando Costa Alberto**

Presidente da Comissão de Pós-Graduação:
Prof. Associado **Luis Fernando Costa Alberto**

Acknowledgements

Primarily, I would like to thank my supervisor, Professor Valdir Grassi Jr., and my unofficial supervisor, Professor Helder Araújo, for tracing this exciting 2-year journey, where I had the opportunity to learn so much. Thank you also for all the time and devotion put into this work, through several suggestions and explanations.

I'm also grateful to all the teachers who helped me, either through lessons or more general support. Thanks to CNPq for financing my stay in São Carlos, and USP/ICMC for providing all the necessary means for the fulfilling of this work.

Special thanks go to all the friends I made while being in São Carlos: A big "thank you" to everyone at LASI, for all the shared moments of support and good humor. Also, to all my friends at LRM and LabRoM, as well as all the friends I made outside the university.

I'm grateful to everyone at Intermodalics/Pick-it, for introducing me in my first professional experience in such a kind way.

Finally, immeasurable gratitude goes to all the people who have been always there for me, namely my family and my boyfriend, as well as his family.

Resumo

PEREIRA, Ana Rita Fernandes Lobo. **Odometria visual: comparando métodos estéreo e multi-câmera.** 2017. 133 p. Dissertação (Mestrado em Ciências) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2017.

O objetivo deste mestrado é implementar, analisar e comparar abordagens de odometria visual, de forma a contribuir para a localização de um veículo autônomo. O algoritmo de odometria visual estéreo Libviso2 é comparado com um método proposto, que usa um sistema multi-câmera omnidirecional. De acordo com este método, odometria visual monocular é calculada para cada câmera individualmente e, seguidamente, a melhor estimativa é selecionada através de um processo de votação que envolve todas as câmeras. O fato de o sistema de visão ser omnidirecional faz com que a parte dos arredores mais rica em características possa sempre ser usada para estimar a pose relativa do veículo. Nas experiências são utilizadas as câmeras Bumblebee XB3 e Ladybug 2, fixadas no teto de um veículo. O processo de votação do método multi-câmera omnidirecional proposto apresenta melhorias relativamente às estimativas monoculares individuais. No entanto, a odometria visual estéreo fornece resultados mais precisos.

Keywords. visão computacional, odometria visual, visão estéreo, visão omnidirecional, múltiplas câmeras

Abstract

PEREIRA, Ana Rita Fernandes Lobo. **Visual odometry: comparing a stereo and a multi-camera approach.** 2017. 133 p. Dissertation (Masters in Science) - São Carlos School of Engineering, University of São Paulo, 2017.

The purpose of this project is to implement, analyze and compare visual odometry approaches to help the localization task in autonomous vehicles. The stereo visual odometry algorithm Libviso2 is compared with a proposed omnidirectional multi-camera approach. The proposed method consists of performing monocular visual odometry on all cameras individually and selecting the best estimate through a voting scheme involving all cameras. The omnidirectionality of the vision system allows the part of the surroundings richest in features to be used in the relative pose estimation. Experiments are carried out using cameras Bumblebee XB3 and Ladybug 2, fixed on the roof of a vehicle. The voting process of the proposed omnidirectional multi-camera method leads to some improvements relatively to the individual monocular estimates. However, stereo visual odometry provides considerably more accurate results.

Keywords. computer vision, visual odometry, stereo vision, omnidirectional vision, multiple cameras

List of Figures

Figure 1	<i>Camera obscura</i>	26
Figure 2	Perspective camera model	27
Figure 3	Perspective camera model in detail	28
Figure 4	Appearance-based corresponding methods	31
Figure 5	Feature-based corresponding methods	32
Figure 6	Epipolar geometry	34
Figure 7	3D reconstruction	38
Figure 8	Homography	39
Figure 9	Panorama built from several images	40
Figure 10	Examples of omnidirectional vision sensors	42
Figure 11	Examples of central systems	43
Figure 12	Caustics	45
Figure 13	Examples of non-central systems	46
Figure 14	General Imaging Model	46
Figure 15	Epipolar geometry of general imaging systems	47
Figure 16	3D-to-3D point alignment problem	51
Figure 17	3D-to-2D point alignment problem	52
Figure 18	2D-to-2D point alignment problem	53
Figure 19	Planar motion assumption	65
Figure 20	Visual compass	66
Figure 21	Flowchart with overview of Libviso2 feature matching steps . . .	76
Figure 22	Libviso2 "circle" feature matching procedure	76
Figure 23	FAST feature detection	80

Figure 24	Sampson distance	82
Figure 25	Flowchart of the voting scheme	85
Figure 26	Carina2 with mounted cameras	88
Figure 27	Vehicle, Ladybug 2 and Bumblebee XB3 reference frames	88
Figure 28	Experimental trajectories	89
Figure 29	Example pictures of the environment	90
Figure 30	Bumblebee camera	92
Figure 31	Geometry of the Bumblebee camera	92
Figure 32	Ladybug 2 camera	93
Figure 33	Geometry of the Ladybug 2 camera	94
Figure 34	ROS Logo	96
Figure 35	OpenCV Logo	96
Figure 36	Eigen Logo	97
Figure 37	Stereo V.O., scenario 1: Estimated trajectory	98
Figure 38	Stereo V.O., scenario 1: Relative motion errors	98
Figure 39	Stereo V.O., scenario 2: Estimated trajectory	99
Figure 40	Stereo V.O., scenario 2: Relative motion errors	99
Figure 41	Omnidirectional multi-camera V.O., scenario 1: Estimated tra- jectory	100
Figure 42	Omnidirectional multi-camera V.O., scenario 1: Relative mo- tion errors	100
Figure 43	Omnidirectional multi-camera V.O., scenario 1: Estimated tra- jectories of all monocular and omnidirectional multi-camera V.O., XY-plane	101
Figure 44	Omnidirectional multi-camera V.O., scenario 1: Estimated tra- jectories of all monocular and omnidirectional multi-camera V.O., XZ-plane	102
Figure 45	Omnidirectional multi-camera V.O., scenario 1: Voted camera at each frame	102
Figure 46	Omnidirectional multi-camera V.O., scenario 1: Relative trans- lation errors of omnidirectional multi-camera (red) and monoc- ular V.O.	104

Figure 47	Omnidirectional multi-camera V.O., scenario 2: Estimated trajectory	105
Figure 48	Omnidirectional multi-camera V.O., scenario 2: Relative motion errors	106
Figure 49	Omnidirectional multi-camera V.O., scenario 2: Estimated trajectories of all monocular and omnidirectional multi-camera V.O., XY-plane	107
Figure 50	Omnidirectional multi-camera V.O., scenario 2: Estimated trajectories of all monocular and omnidirectional multi-camera V.O., XZ-plane	107
Figure 51	Omnidirectional multi-camera V.O., scenario 2: Voted camera at each frame	108
Figure 52	Omnidirectional multi-camera V.O., scenario 2: Relative translation errors of omnidirectional multi-camera (red) and monocular V.O.	109
Figure 53	Stereo and omnidirectional multi-camera V.O., scenario 1: Estimated trajectory	111
Figure 54	Stereo and omnidirectional multi-camera V.O., scenario 2: Estimated trajectory	111
Figure 55	Stereo and omnidirectional multi-camera V.O., scenario 1: Relative motion errors	112
Figure 56	Stereo and omnidirectional multi-camera V.O., scenario 2: Relative motion errors	112
Figure 57	Multiple coordinate frames	130
Figure 58	Transformations between coordinate frames	132

List of Tables

Table 1	Stereo visual odometry algorithms.	59
Table 2	Conventional monocular visual odometry algorithms.	64
Table 3	Omnidirectional monocular visual odometry algorithms.	72
Table 4	Omnidirectional vision specific characteristics of algorithms. . . .	73

List of Abbreviations

USP	Universidade de São Paulo
EESC	Escola de Engenharia de São Carlos
ICMC	Instituto de Ciências Matemáticas e de Computação
LRM	Laboratório de Robótica Móvel
GPS	Global Positioning System
IMU	Inertial Measurement Unit
ROS	Robot Operating System
API	Application Programming Interface
CPU	Central Processing Unit
MP	Mega Pixels
RGB	Red Green Blue
VO	Visual Odometry
FoV	Field of View
CoP	Center of Projection
DoF	Degrees of Freedom
nD	n-Dimensional
FAST	Features from Accelerated Segment Test
SIFT	Scale-Invariant Feature Transform
SURF	Speeded-Up Robust Features
CENSURE	Center Surround Extremas
BRIEF	Binary Robust Independent Elementary Features
ORB	Oriented BRIEF
BRISK	Binary Robust Invariant Scalable Keypoints
SLAM	Simultaneous Localization And Mapping
AR	Augmented Reality
MAV	Micro Air Vehicle
RANSAC	Random Sample Consensus
IEKF	Extended Kalman Filter
EKF	Invariant Extended Kalman Filter
ISPKF	Iterated Sigma Point Kalman Filter
FMT	Fourier-Mellin Transform
FFT	Fast Fourier Transform
SVO	Semi-direct Visual Odometry
ICP	Iterative Closest Point
ELAS	Efficient Large-scale Stereo Matching

Contents

1	Introduction	21
1.1	Visual odometry overview	21
1.2	Proposed work	23
2	Background	25
2.1	The Perspective Camera Model	25
2.1.1	Description	26
2.1.2	Calibration	30
2.1.3	Two views	30
2.1.4	The Correspondence Problem	31
2.1.5	Epipolar Geometry	34
2.2	Omnidirectional Vision	41
2.2.1	Examples of Omnidirectional Systems	41
2.2.2	Central Configurations	43
2.2.3	Non-central Configurations	44
3	Visual odometry	49
3.1	Definition	49
3.1.1	Point alignment	50
3.1.2	Simplification of the model	53
3.1.3	Overview	53
3.2	State of art	54
3.2.1	Stereo Visual Odometry	56
3.2.2	Conventional Monocular Visual Odometry	60
3.2.3	Omnidirectional Monocular Visual Odometry	65

4	Stereo Visual Odometry	75
4.1	Feature Extraction and Matching	75
4.2	Visual Odometry	77
5	Visual odometry for an omnidirectional multi-camera system	79
5.1	Feature extraction and matching	79
5.2	Visual odometry	80
5.2.1	Monocular visual odometry algorithm	81
5.2.2	Voting scheme	83
6	Experiments and results	87
6.1	Experimental procedure	87
6.1.1	Scenarios	89
6.1.2	Analysis	89
6.2	Equipment	91
6.2.1	Hardware	91
6.2.2	Middleware	95
6.2.3	Software	96
6.3	Results	97
6.3.1	Stereo visual odometry	97
6.3.2	Omnidirectional multi-camera visual odometry	99
6.3.3	Comparing stereo and omnidirectional multi-camera ap- proaches	110
7	Conclusion	115
7.1	Summary of results	115
7.2	Future work	116
	Bibliography	119
	Appendices	127
	APPENDIX A Reference frames	129
A.1	Definition	129

A.2	Transformations between Coordinate Frames	130
A.3	Homogeneous Coordinates	132

Chapter 1

Introduction

1.1 Visual odometry overview

In robotic applications the task of localization can be performed in several ways. One can use sensors that provide the absolute position of the robot, like the Global Positioning System (GPS), or sensors that provide information of motion relatively to the previous sampling instant. Such incremental deduction of motion is known as odometry. For example, if the robot is a wheeled vehicle, it is possible to estimate its motion by counting the spins of the left and right wheels (wheel odometry). However, not only does wheel odometry fail at slippery terrain, but it also does not distinguish slopes, and estimates motion as if the ground was plain. An Inertial Measurement Unit (IMU) provides information about its own acceleration and orientation in the x , y and z directions, and one can integrate this information to retrieve first the velocity and then the position. The data obtained by an average priced IMU contains, however, some noise, which grows considerably with each integration. Another way of performing odometry is by means of visual information, i.e., cameras that capture the outside world. Such techniques belong to the field of visual odometry and are the main subject of this work.

Visual odometry (VO) approaches can differ in several aspects and there are many possibilities for estimating the egomotion. For example, methods can be classified by the type and number of cameras being used, or the way they are mounted on the robot, or whether or not they all observe the same elements in the

scene. Finally, the method is classified by the algorithm that receives the visual information and returns the motion estimation.

Independently of the number of cameras used, one can divide the techniques of visual odometry in two big groups, namely monocular and stereo visual approaches. Monocular visual odometry uses only one camera, or several cameras whose field of views (FoVs) contain little or no overlapping. Stereo visual odometry requires two or more cameras observing the same points in the world. Having at least two distinct views of the same elements at the same time allows retrieving the scale of the motion, unlike the case of monocular visual odometry. This same phenomenon is also observed in human sight: both eyes observe the same scene. However, since the eyes are located at distinct positions in the face, each eye observes the scene's elements with a horizontal displacement relatively to the other eye. This displacement, known as disparity, allows the brain to compute the distance of the elements from the viewing person.

Imaging configurations can also be classified according to their FoV. A single perspective camera, or a rig of cameras with highly overlapping FoVs, can only capture a limited part of the scene, i.e, the viewing angle is small. On the other hand, by placing cameras facing different parts of the scene, a larger part of the system's surroundings is captured. In particular, if the viewing angle reaches 360° , the system is classified as omnidirectional. A possible omnidirectional configuration consists of a set of four cameras, one facing the front, one the back and the remaining two facing both sides. Another known family of omnidirectional configurations are catadioptric systems, consisting of a perspective camera facing a curved, convex mirror. The 360° surroundings of the system are reflected on the mirror and an image of the latter is captured by the camera, yielding a ring-shaped image. The most obvious advantage of omnidirectional vision is that it allows following points in the scene along many image frames, even if the camera suffers a drastic turn. Furthermore, the part of the surroundings which has more (or better) features is always available and can be used for estimating the relative pose.

Another possible way of dividing imaging systems is in central and non-central systems. In central systems all viewing rays converge to a single viewpoint, which is the case of perspective cameras. Central omnidirectional sensors are usually catadioptric cameras, which are carefully designed and calibrated in order to keep

all rays meeting at the same point. Towards such restrictions, non-central systems became an interesting topic of research and new models, calibration methods and algorithms have emerged to deal with this type of configuration.

1.2 Proposed work

As discussed before, localization is an important part of any mobile autonomous system. GPS is not always accessible, and may suffer jumps in position depending on the number of available satellites. As such, visual odometry is an alternative solution, which can be used alone in illuminated and textured environments, or complemented with other sensors when visual conditions are not favorable.

This master theses proposes an idea for handling images of multiple, non-overlapping cameras of an omnidirectional rig. According to this method, in each frame, monocular visual odometry is computed for each camera separately. In a second step, the best relative pose estimate is selected through a voting scheme, where all cameras vote for each of the available estimates. Results show that the trajectory formed by the voted relative poses is, in some aspects, more accurate than the individual monocular estimates.

This project also aims at comparing the performance of the stereo visual odometry method Libviso2 with the proposed approach. Results show that the stereo approach leads to more accurate results than the omnidirectional multi-camera approach.

Tests are carried out using an autonomous vehicle with cameras attached. Visual data from cameras Ladybug 2 and Bumblebee XB3, both developed by FLIR (former Point Grey Research), have been used in the tests.

The remainder of this document is organized as follows: Chapter 2 explores theoretical background concepts required for the understanding of the study. Chapter 3 defines visual odometry in greater detail, pointing out some related aspects, and reviews the state of art of stereo, monocular and omnidirectional visual odometry. The evaluated stereo and omnidirectional multi-camera visual odometry approaches are described in Chapters 4 and 5 respectively. Chapter 6 is dedicated to experiments, including setup, analysis, equipment and results. Finally, Chapter 7 adds concluding remarks of the project and points out ideas for future work.

Chapter 2

Background

The purpose of this chapter is to provide a minimal explanation on some concepts, which are referred along this dissertation. This chapter is organized as follows: first, the perspective camera model is explained and, using this model, the geometry for the case of two cameras is explored. Finally, a section about omnidirectional vision is presented, where central and non-central configurations are distinguished.

Notation remark: The left superscript of an element always indicates the coordinate frame in which it is defined. The left subscript indicates the frame obtained after the transformation. A tilde over a point vector indicates that it is written in homogeneous coordinates. For an introduction to reference frames, please refer to Appendix A.

2.1 The Perspective Camera Model

A picture taken by a camera allows extracting some 3D information about the scene. Light rays are reflected from 3D objects and enter the camera's aperture, hitting a photosensitive material and creating the picture. A camera model describes this process through a set of equations, allowing to map 3D points in the world to 2D pixels in an image. A simple and widely adopted representation is the **perspective camera model**, which is described in this section. For more information on the perspective camera model please refer to (TRUCCO; VERRI,

1998).

2.1.1 Description

Figure 1 represents a *camera obscura*, which means “dark room” in Latin. This optical device consists of a box or room with a small hole in one side. A light source illuminates the objects and some wavelengths of the light rays are reflected towards the walls of the room. Some rays pass through the hole and illuminate the opposite wall, inside the room, forming an upside-down image of the illuminated objects outside. If the hole is very small, all rays intersect in a point and the resulting image is in focus, but also very dark, since little light gets in the room. Photographic cameras follow the same principle, but include a complex system of lenses, in order to provide images which are both clear and in focus.

Figure 1: *Camera obscura*, or dark room.

Figure 2 provides more detail about the perspective camera model. The Center of Projection (CoP) is equivalent to the hole in the wall and the image plane corresponds to the opposite wall inside the room, where the image is formed. Notice that the image plane lies between the CoP and the objects. Although in reality the image plane is behind the CoP, this convention has been adopted to allow the visualization of a straight image.

There are three important coordinate reference frames in the model. The **camera coordinate system** has its origin in the CoP and its orientation is such that

Figure 2: The perspective pinhole camera model. The coordinate frames of the world, camera and image are represented in black, red and blue, respectively.

the x-axis points left, the y-axis points up and the z-axis points towards the image plane. In order to determine the 2D position of the points in the image, one must use the **image coordinate system**. The image is converted to a digital format and is quantized in pixels. So, the image coordinates are referred in pixels, and the origin usually lies at the upper left corner of the image. These two coordinate systems exist inside a camera device. Since cameras are movable, a fixed **world coordinate system** is also required, so that the dynamic position of the camera relatively to a static reference can be described.

Let the camera observe a point in the world, P . A ray is projected from P , through the CoP, to the image plane, at point p , in camera coordinates. Its position on the image is described by the 2D point p_{im} , in pixels. Notice that, if the ray p is extended, it reaches the world point P in 3D space.

Now that the perspective model has been explained, one can notice that the geometry of the perspective camera can be described by a set of parameters, indicated in Figure 3. Such parameters are divided into **extrinsic** and **intrinsic**

parameters and are enumerated in the following list:

1. **Extrinsic parameters:** The pose of the camera relative to the world coordinate frame.
 - a) The translation t
 - b) The rotation R
2. **Intrinsic parameters:** The mapping of the points in the image plane from camera to image coordinates.
 - a) The focal length f : The distance of the image plane from the CoP.
 - b) The principal point (o_x, o_y) : The pixel where the z-axis of the camera coordinate system intersects the image plane.
 - c) The horizontal and vertical size of the pixels, s_x and s_y .
 - d) Parameters related to the radial distortion caused by the lens.

Figure 3: The perspective pinhole camera model, showing the intrinsic and extrinsic parameters.

Notice that, as the names suggest, the extrinsic parameters are related to the transformation from world to camera coordinates, while the intrinsic parameters solve the transformation between camera and image coordinates.

It is now shown how a world point, observed by a camera, is converted to pixels. This is achieved following three steps: first, the world point ${}^W P$ is described in camera coordinates, obtaining ${}^C P$.

$${}^C P = M_{ext} {}^W \tilde{P} \quad (1)$$

$$\begin{bmatrix} {}^C X \\ {}^C Y \\ {}^C Z \end{bmatrix} = \begin{bmatrix} {}^C_W R & {}^C_W t \end{bmatrix} \cdot \begin{bmatrix} {}^W X \\ {}^W Y \\ {}^W Z \\ 1 \end{bmatrix} \quad (2)$$

The extrinsic matrix, M_{ext} , features the extrinsic parameters only.

The second step consists of finding the projection of ${}^C P$ in the image plane. The resulting vector, ${}^C p$, has the same direction of ${}^C P$, but its z -component is always equal to the focal length, f . Thus, by similarity of the triangles, the following relation is obtained:

$$\begin{cases} {}^C x = f \cdot \frac{{}^C X}{{}^C Z} \\ {}^C y = f \cdot \frac{{}^C Y}{{}^C Z} \\ {}^C z = f \end{cases} \quad (3)$$

Finally the 3D point in the image plane is converted to pixels. The pixel coordinates do not consider distances, so the pixel size (s_x, s_y) is divided from the 3D coordinates. The subtraction of the image center (o_x, o_y) drags the pixel origin to the upper left corner of the image.

$$\begin{cases} x_{im} = -\frac{1}{s_x} {}^C x + o_x \\ y_{im} = -\frac{1}{s_y} {}^C y + o_y \end{cases} \quad (4)$$

Merging (3) into (4), the following relation is obtained:

$$\tilde{p}_{im} = M_{int} {}^C P \quad (5)$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} {}^C X \\ {}^C Y \\ {}^C Z \end{bmatrix} \quad \begin{cases} x_{im} = \frac{x_1}{x_3} \\ y_{im} = \frac{x_2}{x_3} \end{cases} \quad (6)$$

where M_{int} , the intrinsic matrix, concerns intrinsic parameters only.

Pixel points are directly obtained from 3D world points combining the previous steps. The multiplication of the matrices M_{int} and M_{ext} is often referred as the projection matrix P .

$$\tilde{p}_{im} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = P \cdot {}^W \tilde{P} = M_{int} \cdot M_{ext} \cdot {}^W \tilde{P} \quad \begin{cases} x_{im} = \frac{x_1}{x_3} \\ y_{im} = \frac{x_2}{x_3} \end{cases} \quad (7)$$

2.1.2 Calibration

The act of determining these parameters is known as **Calibration**. There are several methods to calibrate a perspective camera and they often require having patterns of known dimensions (for example, chessboard patterns) in front of the camera. By collecting enough points of the patterns, the algorithms are able to determine all parameters. For instance, a simple calibration method, thoroughly described in (TRUCCO; VERRI, 1998), requires the knowledge of the 3D position and image coordinates of at least seven non-coplanar points. ZHANG (2000) proposed a practical calibration method, where neither the absolute 3D position of the pattern nor the camera motion need to be known. Instead, one can simply move a known pattern in front of the camera, displaying it in different orientations. Calibration is obtained by solving a closed-form problem. Afterwards, a non-linear refinement based on maximum likelihood criterion is carried out. Zhang's calibration method also models the lens' radial distortion. Due to its practicality, this method is widely used.

2.1.3 Two views

Having one view of a scene, thanks to the perspective camera model it is possible to map 2D pixels to rays that point to the corresponding 3D point in the scene. However, without further information about the scene, it is impossible to determine where these rays end. In other words, one cannot know the 3D points' depth, i.e. how far they are from the camera. In order to overcome this problem, at least two views of the same scene are required.

This section discusses the case where two cameras observe the same scene. This could happen because there are actually two cameras being used or, instead, only one moving camera. In the latter case, the two views are two consecutive images taken by the camera.

2.1.4 The Correspondence Problem

If both cameras are observing the same scene, this means that their images contain corresponding points, i.e., pixels in the images that correspond to the same 3D point in the world. A challenge related to binocular view is to find out whether two points are correspondent or not.

Assuming that both views observe most points and that corresponding regions are similar, matching techniques can be divided into two classes, namely appearance-based and feature-based methods (TRUCCO; VERRI, 1998).

2.1.4.1 Appearance-based Methods

In appearance- or correlated-based methods, a window of pixels in one view is compared to several equally sized and shaped windows in the other view, within a search region. In order to compare the regions, it is common to calculate the cross-correlation or the sum of square differences between windows. Figure 4 illustrates region matching in a pair of stereo images.

Figure 4: Appearance-based corresponding methods. Whole image regions are compared and matched. Images taken from the dataset in (GEIGER; LENZ; URTASUN, 2012).

2.1.4.2 Feature-based Methods

A feature is a set of pixels that is different from the immediate neighbourhood pixels in terms of color, intensity or texture. They can be corners, i.e. an intersection of two or more edges, or blobs, which are small regions with the fore-mentioned characteristics. As the name suggests, in feature-based methods features are compared instead of regions (Figure 5).

Figure 5: Feature-based corresponding methods. Point features are compared and matched. Images taken from the dataset in (GEIGER; LENZ; URTASUN, 2012).

In order to find features in an image, a feature detector is used. Ideally, features shall be accurately located, robust to noise, distinctive, invariant to photometric and geometric changes and repeatable, i.e., they can be redetected in latter views. A feature detector shall also be computationally efficient. According to (FRAUNDORFER; SCARAMUZZA, 2012), there are two types of feature detectors, namely corner and blob detectors. Corner detectors, such as Moravec (MORAVEC, 1980), Harris (HARRIS; STEPHENS, 1988) and FAST (ROSTEN; DRUMMOND, 2006), have a faster performance and are well localized in space. On the other hand, blob detectors, such as SIFT (LOWE, 2004), SURF (BAY; TUYTELAARS; GOOL, 2008) and CenSurE (AGRAWAL; KONOLIGE; BLAS, 2008), are more distinctive, repeatable and better invariant to scale and viewpoint. Recently, other detectors were developed, which provide a trade-off between invariance, repeatability and speed (ex. BRIEF (CALONDER et al., 2010), ORB (RUBLEE et al., 2011) and BRISK (LEUTENEGGER; CHLI; SIEGWART, 2011)). The choice for a feature detector shall be based on computational and real-time constraints, on the environment and on the viewpoint variability.

Once a set of features is detected, a descriptor is computed for each feature. As the name suggests, a descriptor describes a feature in several aspects, providing

a way to compare and possibly match them. There are two common approaches when it comes to feature matching: one can find features in one view and track them in the following, or one can find features in both views independently and match them with a similarity criterion. While the former method is suitable for views with similar viewpoints (for instance, views taken before and after a small motion), the latter approach is recommended for large viewpoint changes.

While appearance-based methods are easier to implement, they are usually computationally expensive, depending on the implementation, and fail towards large illumination or viewpoint variations. On the other hand, feature-based methods tend to be more robust, facing the same situations. Furthermore, if the environment is known *a priori*, features can be advantageous. For instance, line features are suitable for indoor, man-made environments, while points are good for outdoor, urban areas (TRUCCO; VERRI, 1998).

The process of matching features or regions becomes less time consuming by limiting the search region. If something about the 3D position of the features or the transformation between both views is known, for example via an additional sensor, one can use epipolar geometry in order to reduce the search region.

2.1.4.3 Dealing with Outliers

Even if a robust feature matching technique is used, one must be aware that wrong matches may always occur. Furthermore, a point in one view may become occluded in the other, or belong to an independently moving object. These misleading correspondences are known as outliers and may corrupt the estimation of the relative position between two views. It is desired to ensure that all correspondences used in the relative pose estimation are correct.

In 1981, Fischler and Bolles proposed an iterative technique known as **Random Sample Consensus**, or shortly **RANSAC**, which rejects outliers when trying to find the model that better fits the available points (FISCHLER; BOLLES, 1981). Let the target model require N points to be computed. The first step consists of randomly selecting a sample of N points from the population and use it to compute the model. Afterwards, the remaining points are tested against the obtained model. If a sufficiently large number of points fits the model with an

error below a threshold (inliers), a better model is computed out of all inliers and the algorithm ends here. Otherwise, the whole procedure is repeated until enough inliers are obtained. If this does not occur for a maximum number of iterations, the algorithm returns no answer. This number is computed such that the failure probability is as small as desired.

In visual odometry, the model usually consists of the relative motion between two views. The number of points required to estimate it depends on the visual odometry algorithm, or motion restrictions that might be taken into account.

2.1.5 Epipolar Geometry

This section explores the geometry of the situation where two cameras, A and B , are observing one same point P , illustrated by Figure 6a.

Figure 6: Epipolar geometry. Blue points: epipoles. Green lines: epipolar lines.

The vectors A_p and A_P share the same direction, but not the scale. Both start at the CoP of camera A , A_p reaches the image plane π_A and A_P the 3D world point P . The same goes for camera B . A_e is the projection of camera B 's CoP in camera A 's image plane, and vice-versa. A_e and B_e are known as the **epipoles** and their location in the image plane (or out of it) depends on the translation t and rotation R between both camera coordinate frames. The translation t is computed as $CoP_B - CoP_A$ (Figure 6b).

Note: t is the translation of camera B described in the coordinate frame of camera A . R is the rotation of camera A relatively to camera B . For sake of

clarity, the symbols t and R are used, instead of the more complete ones, ${}^A_B t$ and ${}^B_A R$.

Note: ${}^A P$ and ${}^B P$ point to the same point in the world, P . The only difference is the coordinate frame in which this point is described. ${}^A p$ and ${}^B p$, also described in the coordinate frame of camera A and B respectively, point to different points in the world, namely the projection of P in the image planes of cameras A and B respectively.

Note: When discussing several views, the extrinsic parameters no longer refer to the pose of the cameras in the world reference frame. Instead, they describe the relative pose (translation and orientation) of the cameras among themselves. Usually, one of the cameras is considered to have null translation and orientation and the position of the remaining cameras is described relatively to the reference camera. In this example, the left camera is chosen as the reference.

The plane formed by the points P , CoP_A and CoP_B is known as the **epipolar plane** and intersects the image planes π_A and π_B in **epipolar lines** ${}^A u$ and ${}^B u$, respectively. Notice that, for a point in the first view, ${}^A p$, and for a given transformation between cameras (t, R) , the corresponding point in the second view, ${}^B p$, can be at any location along the epipolar line ${}^B u$. This reduces the search region to a line, saving a large amount of computational effort and helping identifying false matches. Each image point has only one associated epipolar line, except for the epipole, which features in all lines. For further information consult (TRUCCO; VERRI, 1998).

2.1.5.1 The Essential Matrix

Once the reader is aware of the existence of the epipolar geometry, it is time to explore how two corresponding image points are related and how to find the epipolar lines.

The vectors ${}^A P$, t and ${}^A P - t$, all written in the coordinate frame of camera A , are coplanar (6, b)). Thus, the following constraint holds:

$$({}^A P - t)^T t \times {}^A P = 0 \quad (8)$$

It is observed that

$${}^B P = R({}^A P - t) \quad \leftrightarrow \quad R^{TB} P = {}^A P - t \quad (9)$$

Replacing ${}^A P - t$ for $R^{TB} P$ in the previous equation, the following is obtained:

$$(R^{TB} P)^T t \times {}^A P = {}^B P^T R t \times {}^A P = 0 \quad (10)$$

Finally, knowing that the cross product $t \times x$ is equivalent to the multiplication $[t]_{\times} x$, where

$$[t]_{\times} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}, \quad (11)$$

the equation becomes

$${}^B P^T E {}^A P = 0 \quad \text{with} \quad E = [t]_{\times} R \quad (12)$$

The matrix E is known as **Essential Matrix** and relates two corresponding 3D rays ${}^A P$ (written in the coordinate frame of camera A) and ${}^B P$ (written in the coordinate frame of camera B). The epipolar line in the view B , ${}^B u$, is defined as

$${}^B u = E {}^A P. \quad (13)$$

In addition to relating two corresponding image points, the Essential Matrix also allows determining the epipolar line associated to a given image point.

2.1.5.2 The Fundamental Matrix

The essential matrix works with the 3D points described in the camera coordinate system. However, it may happen that the intrinsic parameters are unknown, and the only available information is the pixel images. Knowing that 3D points and corresponding 2D pixels are related as

$$p = M_{int}^{-1} p_{im} \quad (14)$$

where M_{int} is the matrix with the intrinsic parameters, Equations (??) and (13) can be rewritten as

$${}^B p_{im}^T F {}^A p_{im} = 0 \quad \text{with} \quad F = M_{B,int}^{-T} E M_{A,int}^{-1} \quad (15)$$

$${}^B u_{im} = F^A p_{im} \quad (16)$$

The essential and fundamental matrices play similar roles. While the essential matrix relates corresponding 3D points, the fundamental matrix relates corresponding 2D pixels. Thus, while the essential matrix encodes information about the extrinsic parameters only (translation and rotation between both cameras), the fundamental matrix also retains information about both cameras' intrinsic parameters, which relate 3D points and 2D pixels.

If only two images from a stereo pair are available, one can compute the essential and fundamental matrices and recover the extrinsic and intrinsic parameters out of them.

2.1.5.3 Reconstruction

If the intrinsic and extrinsic parameters of the stereo pair are known, it is possible to determine the 3D location of the observed points. The main idea is briefly explained in this section.

Figure 7a illustrates two cameras observing one point. Since the cameras are calibrated, the viewing rays ${}^A p$ and ${}^B p$ are known. By the figure it is clear that, if these rays are augmented, they will intersect at the observed point in 3D space. Due to measurement or calibration inaccuracies, it is unlikely that the rays ever meet. In this case the segment that joins the augmented rays when they are closest is computed. The observed 3D point is considered to be at the middle of that segment (Figure 7b).

In order to solve for the observed 3D point, it is important to describe all vectors in the same reference frame. Here, the coordinate frame of camera A is chosen. The translation vector and the viewing vector of camera A are already described by the selected reference frame. The viewing ray of camera B becomes ${}^B R^T \cdot {}^B p$. The segment shall be orthogonal to both viewing rays, so its direction is given by their cross-product: ${}^A p \times {}^B R^T \cdot {}^B p$. The viewing rays and the segment are resized (by multiplying the constants a , b and c) to form a closed path.

$$a \cdot {}^A p - b \cdot {}^B R^T \cdot {}^B p + c \cdot ({}^A p \times {}^B R^T \cdot {}^B p) = {}^A t \quad (17)$$

Figure 7: 3D reconstruction. a) Ideal case. b) Realistic case, with measurement/calibration errors. Red: viewing rays. Blue: translation vector. Green: vector orthogonal to both viewing rays.

Solving the equations for the three constants, the middle point of the segment corresponds approximately to the observed 3D point. Notice that the obtained point is described in coordinates of camera A .

$$P = a \cdot \mathbf{A}p + \frac{1}{2} \cdot c \cdot (\mathbf{A}p \times \mathbf{B}p) \quad (18)$$

2.1.5.4 Homographies

When a camera observes a 3D plane, it is said that the points on the image plane and the points on the observed plane are related by a **homography**. This relationship can be written in the form of a 3x3 matrix, as shown in this section.

As demonstrated previously, and considering homogeneous coordinates, an observed 3D point in the world and its image pixel are related by a 3x4 projection matrix.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (19)$$

If it is known that all the observed points lie on a plane which satisfies $aX + bY + cZ = d$, then all points have the following constraint:

$$Z = -\frac{aX + bY + d}{c} \quad (20)$$

and Equation (19) can be rewritten like this:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ -\frac{aX+bY+d}{c} \\ 1 \end{bmatrix} \quad (21)$$

The Z component of the points disappears and the projection matrix may be rearranged to a 3x3 homography matrix H .

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = H \cdot \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \quad H = \begin{bmatrix} p_{11} - \frac{a}{c}p_{13} & p_{12} - \frac{b}{c}p_{13} & p_{14} - \frac{d}{c}p_{13} \\ p_{21} - \frac{a}{c}p_{23} & p_{22} - \frac{b}{c}p_{23} & p_{24} - \frac{d}{c}p_{23} \\ p_{31} - \frac{a}{c}p_{33} & p_{32} - \frac{b}{c}p_{33} & p_{34} - \frac{d}{c}p_{33} \end{bmatrix} \quad (22)$$

Therefore, the homography matrix performs a projective transformation between two planes, namely a known plane in the world and the image plane.

Consider Figure 8, where two cameras A and B observe the same 3D plane.

Figure 8: Homography. Two cameras observe a 3D plane. Red: viewing rays. Blue: image points. Green: world plane.

There is, for each camera, a homography that relates the world plane (W) and the image plane (A and B).

$$\begin{cases} \tilde{p}_{im,A} = {}^A_W H \cdot \tilde{P} \\ \tilde{p}_{im,B} = {}^B_W H \cdot \tilde{P} \end{cases} \quad (23)$$

The homographies ${}^A_W H$ and ${}^B_W H$ can be combined to form a homography that relates the image planes of the two cameras.

$${}^A_B H = {}^A_W H \cdot {}^W_B H = {}^A_W H \cdot {}^B_W H^{-1} \quad \tilde{p}_{im,A} = {}^A_B H \cdot \tilde{p}_{im,B} \quad (24)$$

The observed plane therefore induces a homography between the two views, allowing transferring points from one image plane to the other.

If the distance from the cameras to the observed points is much larger than the distance differences among points, one can assume that the points lie on a plane far away. This assumption is known as weak perspective. Since the plane is far away, its c component is a very large number, and the homography matrix may be simplified:

$$H \approx \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} \quad (25)$$

This technique is employed in the elaboration of image panoramas (Figure 9).

Figure 9: Panorama built from several images. Taken from (BOBICK, 2014).

For further information, the lecture notes (BOBICK, 2014) are recommended, as they explain homography very clearly, as well as other computer vision topics.

2.2 Omnidirectional Vision

This section enumerates some omnidirectional configurations, classifying central and non-central setups.

A traditional perspective camera has a limited FoV, which means that it only captures a part of the scene around it. There are, however, several ways of capturing the whole surroundings, providing omnidirectional vision. This can be an advantage in many robotics applications, surveillance operations and other tasks, since features or objects that are being tracked are captured for a long time, independently of the camera's orientation. Using single or stereo perspective cameras, features disappear at the borders of the images as the camera moves.

2.2.1 Examples of Omnidirectional Systems

NAYAR (1997) and GROSSBERG; NAYAR (2001) enumerate alternatives for omnidirectional vision, which are briefly described in the following paragraphs.

A solution for omnidirectional vision is to use a catadioptric camera, shown in Figure 10a. This setup consists of having a perspective camera pointed at a convex mirror. The scene is reflected into the surface of the mirror and the camera captures this reflection. Mirrors may have different shapes, such as spheres, parabolas, ellipsis or hyperbolas.

One can obtain a large FoV attaching a wide-angle lens to a perspective camera, as illustrated in Figure 10b. Popular wide-angle lenses are fish-eye lenses.

If working in a static scene, omnidirectional vision can be obtained by rotating a camera in a way that it covers the whole surroundings. This configuration is popular in surveillance applications and an example of such a system can be seen in Figure 10c.

Finally, one can mount several perspective cameras in a way that their FoVs cover 360° around the setup, creating a camera cluster. Figure 10d shows two examples of omnidirectional camera clusters.

The catadioptric camera and the fish-eye lens setup suffer from low resolution and distortions. The rotating camera is not suitable for real-time applications, since it needs some time to capture the whole surroundings, requires a static

Figure 10: Examples of omnidirectional vision sensors. a) A catadioptric camera, taken from (XIANG; SUN; DAI, 2012). b) A camera with a fish-eye lens, taken from <http://www.premiumbeat.com/blog/10-incredible-camera-lenses>. c) A rotating camera system, taken from <http://qualityelectric-inc.com/blog/security/video-surveillance>. d) A camera cluster, taken from <https://www.ptgrey.com/ladybug5-30-mp-usb-30-spherical-digital-video-camera-red>.

scene and has all technical disadvantages of mechanical moving setups. It also produces a large amount of visual information that has to be processed, implying a significant computational cost. The latter disadvantage is shared by camera clusters. Since none of the setups presented here is flawless, the adequacy of the chosen configuration depends on the particular application and available resources. Omnidirectional camera clusters are also referred as omnidirectional multi-camera systems along this document.

There is, however, another characteristic of cameras which is not obvious, but can make a large difference on how the visual information is handled: cameras can be classified as being **central** or **non-central**.

2.2.2 Central Configurations

2.2.2.1 Description

In a central configuration, all rays from the scene which are projected into the image plane meet at one point only. This is the case of traditional perspective cameras, where all rays meet at the CoP, as shown in Figure 11a. In order to obtain a central omnidirectional sensor, one can build a system where a perspective camera rotates about its CoP, as shown in Figure 11b. It is also possible to design a central catadioptric camera. Here, the rays hitting the mirror would meet at a single viewpoint, if they were not reflected by the mirror surface to the CoP. Figure 11c shows a central catadioptric camera, consisting of a perspective camera facing a parabolic mirror. BAKER; NAYAR (1998) determines the mirror shapes that allow a catadioptric system to be central. They conclude that the mirror needs to be a quadric surface, including parabolas, hyperbolas and ellipses.

Figure 11: Central cameras. Red: the single viewpoint. a) A perspective camera. b) A perspective camera rotating about its CoP. c) A central catadioptric camera.

2.2.2.2 Epipolar Geometry

Suppose that there are two central catadioptric cameras imaging of the same scene, camera 1 and camera 2, and they capture a point P in the world, projecting the corresponding points p_1 and p_2 in their images. SVOBODA; PAJDLA (2002) proves that p_2 lies on an epipolar curve associated to the ray through p_1 . In other words, just as there are epipolar lines for two perspective views, there are epipolar

curves for two catadioptric views. The authors deduce those curves for parabolic, elliptic and hyperbolic mirrors.

The existence of one viewpoint implies several advantages, being one of them the possibility of retrieving undistorted perspective views from the omnidirectional images. In fact, GEYER; DANIILIDIS (2000) proposes a two-step method for obtaining perspective images out of catadioptric images. The first step consists of projecting the image into a spherical surface. Secondly, the image on the sphere is projected into a plane. This way, countless computer vision algorithms that have been developed for perspective cameras can equally be applied for catadioptric systems, as long as they are central (NAYAR, 1997).

Unfortunately, it is difficult to manufacture a central camera in practice. For a catadioptric system to be central, the mirror has to be placed accurately relatively to the camera, which makes its design difficult. Central systems also require a careful calibration. NAYAR (1997) proposes a catadioptric system composed of a parabolic mirror and an orthographic camera, so that the mirror can be placed at any distance from the camera. However, in general, central systems are not practical, since any small inaccuracy jeopardizes centrality, invalidating the central model. As such, much research started being carried out around non-central systems.

2.2.3 Non-central Configurations

2.2.3.1 Description

In non-central systems, viewing rays do not meet at one point, but in a loci of points. The multiple convergence points of light rays are known as caustics and can be observed, for instance, at the bottom of pools. Here, the uneven surface of the water deviates the light rays to meet at specific places, causing shiny shapes to appear at the bottom surface (Figure 12).

In general, omnidirectional or wide-angle imaging systems are non-central. For instance, in multi-camera systems, in the general case where the CoP's of the cameras are not aligned, the rays meet at several points, one for each camera, as shown in Figure 13a. An arbitrary catadioptric system, built without the caution of having all rays meeting at one point, is non-central (Figure 13b). Any rotating

Figure 12: Caustics at the bottom of a bowl with water.

system where the camera does not rotate about its CoP is also not central, and an example of such configurations is shown in Figure 13c. Many wide-angle lenses do not allow centrality either (Figure 13d). Furthermore, if the scientific community one day manages to replicate the ommatidia in insects' eyes, this omnidirectional viewing system will also be non-central (Figure 13e) (GROSSBERG; NAYAR, 2001).

2.2.3.2 General Imaging Model

Facing the excessive restrictions of central systems, researches embraced non-centrality, working on models that are general enough to include non-central imaging systems. A popular model is the **General Imaging Model**, proposed by GROSSBERG; NAYAR (2001). The authors state that all imaging systems can be seen as a set of photo-sensitive elements (pixels) that measure light from several directions in 3D space. Thus, any viewing system is a mapping between incoming scene rays to pixels, as shown in Figure 14a. Each pixel measures light from one main direction and this mapping is abstracted to a *raxel*, which is an abbreviation of the term *ray pixel*. A raxel is composed of a photo-sensitive element, that measures incoming light, and a lens that collects a bundle of light rays (Figure 14b). This bundle is simplified to a single ray with its central direction (Figure 14c). A raxel has radiometric, optical and geometric properties, yet, in most cases, only the latter are taken into account, namely its position and direction. Calibration consists of determining, for each pixel, the position and direction of its corresponding

Figure 13: Non-central systems. a) A multi-camera system. b) An arbitrary cata-dioptric system. c) An arbitrarily rotating camera. d) Camera with an arbitrary wide-angle lens. e) Compound camera system, where each element possesses a receptor and a lens.

ray, and a method based on structured light patterns is proposed by the authors.

Figure 14: a) A general imaging system. A ray P_i travels an arbitrary path and is mapped to a photo-sensitive element i . b) A raxel is composed by a photo-sensitive material and a lens, which collects a bundle of light rays. c) A simplified raxel, consisting of its position and main direction of the bundle.

2.2.3.3 Plücker Lines

Based on the General Imaging Model, PLESS (2003) proposes representing the 3D rays in Plücker lines, which are described by the direction q and moment q' of the line. Plücker lines usually follow the notation $l = \langle q, q' \rangle$. The 3-dimensional direction vector q can have any length and is usually forced to have an unit module. The 3-dimensional moment vector q' can be obtained as the cross product of a point P on the line and the direction: $q' = q \times P$. Plücker vectors are homogeneous, which means that scale does not change the line they are representing. Also, for any line the constraint $q \cdot q' = 0$ is satisfied. If q has unit length, $q \times q'$ gives the point on the line closest to the origin.

Two lines $l_A = \langle q_A, q'_A \rangle$ and $l_B = \langle q_B, q'_B \rangle$ intersect if

$$q_B \cdot q'_A + q'_B \cdot q_A = 0 \quad (26)$$

2.2.3.4 Epipolar Geometry

PLESS (2003) deduces a generalized epipolar geometry in the sense that corresponding points constrain the camera motion. Figure 15 illustrates the case where two general cameras observe a 3D point.

Figure 15: Two general cameras observe the same 3D point P . Each camera has its own coordinate system, represented in blue and red respectively. 3D rays l_A and l_B are the rays associated to the pixels corresponding to the images of point P in camera A and B respectively.

$l_A = \langle q_A, q'_A \rangle$ and $l_B = \langle q_B, q'_B \rangle$ are expressed in the coordinate system of cameras A and B respectively. Representing l_A in the coordinate system of camera B, the following equation is obtained:

$$\langle Rq_A, Rq'_A + R(t \times q_A) \rangle, \quad (27)$$

where R and t are the rotation and translation of camera A described in the coordinate system of camera B. Having l_A and l_B described in the same coordinate system and knowing that they intersect in point P , one can apply the constraint (26), obtaining

$$q_B^T Rq'_A + q_B^T R[t]_{\times} q_A + q'^T_B Rq_A = 0, \quad (28)$$

or, writing in the matricial form (KNEIP; LI, 2014):

$$l_B^T \cdot \varepsilon \cdot l_A = 0, \quad \varepsilon = \begin{bmatrix} E & R \\ R & 0 \end{bmatrix}, \quad (29)$$

where $l_A = [q_A, q'_A]^T$ and $l_B = [q_B, q'_B]^T$, $E = R[t]_{\times}$ and ε is known as the generalized camera matrix. MIRALDO; ARAUJO (2015) deduces important properties of its singular value decomposition.

Chapter 3

Visual odometry

This chapter explains the concept of visual odometry, emphasizing some characteristics. Afterwards, the state of art is described, by summarizing several approaches and dividing them into categories.

3.1 Definition

In this section, the concept of visual odometry is clarified and some possible approaches are further explained.

Visual odometry is the process of retrieving the relative motion of one or more cameras based solely on their images. In other words, for example, given two images taken to the same scene from locations A and B , visual odometry aims at finding the transformation which brings the camera from A to B (A_BT). Applying this process to a sequence of images, a chain of relative transformations is obtained, which can be multiplied to form the pose of the camera relatively to the initial pose, usually arbitrated as the identity matrix. Due to the relative nature of visual odometry, eventual estimation errors accumulate along the path, leading to trajectory drifts. Without a global reference, like a GPS or a map of the environment, these errors cannot be corrected. In order to improve localization, visual odometry is usually combined with other sensors, like an IMU or a GPS.

A related localization process is SLAM (Simultaneous Localization And Mapping). As the name suggests, in addition to localizing the robot, SLAM also

provides a map of the environment. The localization and mapping tasks are cooperative, as finding out the relative motion is necessary for building the map and the map provides a global reference to improve localization. Having a map, it is possible to recognize previously visited places, eliminating trajectory drifts. This technique, known as loop closure, has been subject to deep research, as it opens several challenges, like memory management. Visual odometry, on the other hand, only focuses on the localization task and not the mapping. It may store 3D points of the environment, but only for improving the motion estimation locally (local bundle adjustment) and not for building a map. Therefore visual odometry is commonly simpler to implement than SLAM, but, if not complemented with other sensors, it usually provides worse results.

3.1.1 Point alignment

Independently of the used method to retrieve rotation and translation between cameras, any method checks whether or not the available feature matches (2D points) between previous and current images agree with an estimated relative pose. As an example, let us consider the following case: A camera captures an image, moves in a specific way and captures another image. Then, it is expected that the features in the first image are found in the second image in certain locations (different from the ones in the first image due to the motion). However in our case the camera motion is not known and the only available information is the 2D position of matching features in the first and second images. In this case, relative pose estimates must be tested. Given features in the first image and a candidate motion estimate, the estimate is correct if the expected position of the features in the second image given that motion is equal to the one actually observed. In other words, relative pose is optimized such that expected and observed 2D points in the second image are in the same position, or aligned. In practice, measurement and calibration inaccuracies prevent a perfect alignment from all points. As such, the goal is to minimize the difference between all expected and observed points, leading to the best possible alignment. There are three common approaches for point alignment, namely 3D-to-3D, 3D-to-2D and 2D-to-2D methods.

3.1.1.1 3D-to-3D Point Alignment

Methods that use 3D-to-3D point alignment usually use stereo vision (Figure 16).

Figure 16: 3D-to-3D point alignment problem in stereo systems. Red: 2D and estimated 3D feature points at frame $k-1$. Blue: 2D and estimated 3D feature points at frame k . Green: Estimated transformation between frames.

The following steps are carried out: Given a left and right images at an instant $k-1$, points are triangulated, and their 3D positions are achieved. The same is performed for the left and right images of the following instant, k . Having two sets of 3D points, described in the coordinate system of frames $k-1$ and k respectively, the transformation between both frames can be inferred. Rotation and translation are computed such that the 3D error between corresponding triangulated points is minimized.

3.1.1.2 3D-to-2D Point Alignment

This type of approach is common in both monocular and stereo vision systems. 3D points are only triangulated after computing the relative motion between two frames. In stereo systems, triangulation is performed using one frame and, in

monocular systems, two frames are used for the effect (Figure 17). Knowing the 3D position of the points, depending on the transformation, they reproject at certain 2D positions in the next frame. The transformation between frames is computed, so that the consequent 2D feature positions in the next frame are as close as possible to the actual, observed positions. In other words, the reprojection error is minimized. Since 3D-to-2D methods triangulate points only once each step, they accumulate a smaller error than 3D-to-3D approaches, which triangulate twice.

Figure 17: 3D-to-2D point alignment problem in stereo systems. a) Stereo systems. b) Monocular systems. Red: 2D and estimated 3D feature points at frame $k - 1$ (stereo case) or $k - 2$ and $k - 1$ (monocular case). Blue: 2D feature points at frame k . Green: Estimated transformation between frames which leads to the reprojection of the 3D points to feature positions in the most recent frame, k .

3.1.1.3 2D-to-2D Point Alignment

2D-to-2D methods do not require the triangulation of 3D points (Figure 18). Instead, 2D feature points are enough to compute the fundamental or essential matrices, from which rotation and translation between frames may be extracted.

As triangulation is never performed, these methods allow achieving a more accurate motion estimation.

Figure 18: 2D-to-2D point alignment problem. Red: 2D feature points at frame $k - 1$. Blue: 2D feature points at frame k . Green: Estimated transformation between frames.

3.1.2 Simplification of the model

In some cases it is not necessary to estimate the full 6 DoF of the motion. For instance, many robotic applications consist of having a wheeled robot navigating on a planar floor. In this case, the height and the roll and pitch angles of the camera are known and do not change and this information simplifies the visual odometry process, allowing the use of less points to estimate the relative pose and to retrieve the translation scale. Such methods calculate a 3 DoF relative pose, as only the translation to the front and to the side and the yaw angle are retrieved.

3.1.3 Overview

Chapters 2 and 3 introduced some aspects that may characterize a visual odometry algorithm (SCARAMUZZA; FRAUNDORFER, 2011; FRAUNDORFER; SCARAMUZZA, 2012). They are summarized in the following list:

- **Number of cameras observing the scene:** While monocular vision systems do not allow retrieving translation scale, stereo systems with a baseline between the cameras do.

- ❑ **Field of view:** A vision system with a large field of view has the advantage of capturing more information of the surroundings. If the whole 360 degrees of the environment are covered, then the system is known as omnidirectional.
- ❑ **Centrality:** A vision system is central if the 3D rays from all image pixels converge in one single view point. In this case, translation scale is not retrievable. On the other hand, if the 3D rays converge to more than one viewpoints, then the system is non-central and translation scale can be recovered for non-degenerate motions.
- ❑ **Information extraction:** The two most common ways to extract information from captured images are using feature-based and appearance-based methods. While the former approach searches for corners or blobs, the latter uses larger areas of the images.
- ❑ **Point alignment:** Visual odometry methods may differ through the way expected and actual points are aligned, leading to the motion estimation that minimizes the alignment error.
- ❑ **Motion DoF:** Finally, a visual odometry approach is characterized by the number of DoFs of the motion estimation.

3.2 State of art

This section describes existent approaches of visual odometry. The methods are divided in three great groups, namely stereo, non-omnidirectional monocular and omnidirectional monocular visual odometry. Distinctions are made within each group: in stereo visual odometry, approaches are divided into 3D-to-3D, 3D-to-2D and 2D-to-2D point alignment problems. Non-omnidirectional monocular visual odometry methods are divided into feature- and appearance-based. Finally, omnidirectional visual odometry strategies are divided into methods that use exclusively central cameras and that also allow non-central configurations.

The way points are aligned (3D-to-3D, 3D-to-2D or 2D-to-2D) and the way the image information is perceived (using features or general appearance) are independent characteristics. In fact, a complete analysis of the state of art would

include each possible combination of the mentioned approaches. For instance, a visual odometry strategy may use a catadioptric camera (omnidirectional visual odometry), solve a 3D-to-2D point alignment problem and be feature-based.

As for centrality, according to the definition, methods where the imaging system consists of one simple camera alone (no mirror or wide-angle lens) are always central. Methods that include more than one camera are, in general, non-central, since the global imaging system normally possesses more than one effective view point. Methods that use one camera attached to a mirror or a wide-angle lens may or may not be central, depending on the system design. The concept of centrality is, however, rarely used in non-omnidirectional vision.

At the end of each section, a table summarizes the main characteristics of the discussed methods. For the tables to fit in the page, the following initials have been adopted.

- ❑ P.A. - Point alignment type
- ❑ D.E. - Data extraction type
 - F. - Features
 - A. - Appearance
- ❑ I.R. - Iterative refinement
- ❑ K.F. - Kalman filter
- ❑ S. - Scale is retrieved
- ❑ DoF - Degrees of freedom of the estimated motion
- ❑ N.Pts. - Number of points required to estimate motion
- ❑ R.T. - Achieves real-time performance

Fields with a question mark denote that it was not possible to retrieve the information with certainty from the work's reference.

3.2.1 Stereo Visual Odometry

Stereo visual odometry consists of estimating the motion of the camera through more than one image of the same scene. Thus, not only are features matched temporally, i.e. frame by frame, but also spatially, i.e. among the different cameras in the rig. The most common configuration of stereo visual odometry features two cameras, a left and a right one. Stereo visual odometry can be divided into three strategies: 3D-to-3D, 3D-to-2D and 2D-to-2D point alignment methods. Some examples on each approach are briefly described in the next sections. SCARAMUZZA; FRAUNDORFER (2011) indicates important works on each technique and KRESO(2014) explains the main steps for each approach.

3.2.1.1 3D-to-3D Point Correspondences

In (OLSON et al., 2003) the authors follow a 3D-to-3D approach. They use a maximum-likelihood formulation to estimate motion, modeling the error with great accuracy. They also select features in an optimum manner, improve motion prediction and use several outlier rejection strategies, increasing robustness and accuracy comparing to contemporary works.

The work in (MILELLA; SIEGWART, 2006) uses the intensity information of each stereo pair to generate a density map. This helps defining the stereo-confidence level of features, detected using the Shi-Tomasi method. Matches between consecutive frames are determined, allowing an approximate motion estimation. This estimation is refined using the Iterative Closest Point (ICP) algorithm.

3D points far away introduce little disparity in stereo images. With disparities of few pixels, points are triangulated with a large uncertainty and their estimated 3D position hardly corresponds to reality. POJAR; JEONG; NEDEVSKI (2012) overcomes this problem by associating weights to each point according to their certainty. This means that, in the process of estimating motion, points with a large uncertainty level will have a smaller influence than trustworthy ones.

Another way to get around the uncertainty problem is suggested by OTSU; KUBOTA (2014) and consists of choosing points in a clever manner. The authors solve the orientation and translation separately. First, a point far away is used to determine the camera's rotation. Afterwards they use a near point (precisely

triangulated) to determine the translation, which is possible once the orientation is known. This method uses triangulation to determine the depth of the points. In addition, it only requires two points, resulting in a computationally efficient approach.

3.2.1.2 3D-to-2D Point Correspondences

NISTER; NARODITSKY; BERGEN (2004) follows a 3D-to-2D approach, using pre-emptive RANSAC and an iterative refinement to estimate the cameras' pose.

CUMANI; GUIDUCCI (2008) suggests a new feature detector and descriptor, as a faster version of the SURF detector. In order to eliminate accumulation errors, this method detects whether a scene was already visited or not. Features of non-repetitive scenes are stored and periodically compared with the current scene. A bundle adjustment is performed, minimizing the reprojection error.

GEIGER; ZIEGLER; STILLER (2011) uses two Central Processing Units (CPUs) to perform sparse stereo matching and visual odometry at a high frequency and dense matching and 3D reconstruction at a lower frequency. The egomotion is estimated by minimizing the reprojection error within a Kalman filter framework, where velocity is assumed to be constant. The authors also use RANSAC to guarantee that all features used in the estimation belong to static objects in the scene. Dense 3D reconstruction is performed in an accurate way over several thousands of features, which have been efficiently matched using ELAS (Efficient Large-scale Stereo Matching) (GEIGER; ROSER; URTASUN, 2010).

MOU; WANG; SEET (2014) develops a novel feature descriptor which is faster than the existing ones. Thus, in the motion estimation process, they can use as many features as possible, obtaining more accurate results than the existing methods, which use fewer features.

BASTANLAR (2014) achieves accurate results by combining a perspective camera with an omnidirectional one. The author uses the sphere camera model, which suits both the perspective and the central omnidirectional cameras. Through an automated hybrid feature point matching, the epipolar geometry is estimated. 3D points are triangulated and used in the motion estimation. In a bundle adjustment framework, the 2D reprojection error is minimized using the Levenberg-Marquardt

method. Combining both types of cameras leads to a reduced drift.

PERSSON et al. (2015) generalizes existent monocular visual odometry methods to the stereo case. Monocular methods are usually more robust than stereo ones, due to the necessity of providing a good motion estimation with less resources. Thus, if adapted to the stereo case, they shall outperform existent stereo and monocular visual odometry methods. In this work, motion is predicted by assuming a constant velocity and RANSAC-PNP is used to estimate the pose.

3.2.1.3 2D-to-2D Point Correspondences

COMPORT; MALIS; RIVES (2007) does not triangulate 3D points throughout the process, by making use of a quadrifocal tensor between consecutive image pairs.

KITT; GEIGER; LATEGAHN (2010) avoids triangulation by means of a trifocal tensor. It relates the left and right previous views and the right (or left) current view. The authors use an Iterated Sigma Point Kalman filter (ISPKF), where the prediction step is based on the assumption of constant velocity and the update step is based on measurements of corner-like features. The choice for the ISPKF is justified by its out-performance towards highly non-linear equations, comparing to other Kalman filter variations. The chosen features are evenly distributed within a region of interest, in order to avoid cases where most features lie on independently moving objects, yielding to erroneous motion estimations. Finally, RANSAC is used for outlier removal.

3.2.1.4 Summary - Stereo Visual Odometry

Table 1 summarizes the main characteristics of the mentioned approaches, listed in chronological order.

Furthermore, OLSON et al. (2003) complements visual odometry with an orientation sensor and CUMANI; GUIDUCCI (2008) recognizes loop closures.

The table shows that most mentioned authors adopt 3D-to-3D or 3D-to-2D point alignment approaches and extract features from the image, instead of the whole appearance. Most works achieve a real-time performance, even if carrying out iterative refinement. All works retrieve the 6 DoF motion of the camera, which includes the translation scale, availing of the properties of stereo visual odometry.

	3D-3D	P.A.		D.E.		I.R.	K.F.	S.	DoF	N.Pts.	R.T.
		3D-2D	2D-2D	F.	A.						
(OLSON et al., 2003)	X			X		X		X	6	?	X
(NISTER; NARODITSKY; BERGEN, 2004)		X		X		X		X	6	3	X
(MILELLA; SIEGWART, 2006)	X			X	X	X		X	6	?	X
(COMPORT; MALIS; RIVES, 2007)			X		X	X	X	X	6	-	X
(CUMANI; GUIDUCCI, 2008)		X		X		X		X	6	?	X
(KIT; GEIGER; LATEGAHN, 2010)			X	X			X	X	6	3	X
(GEIGER; ZIEGLER; STILLER, 2011)		X		X		X	X	X	6	3	X
(POJAR; JEONG; NEDEVSKI, 2012)	X			X				X	6	?	X
(SILVA; BERNARDINO; SILVA, 2014)	X			X	X		X	X	6	?	X
(OTSU; KUBOTA, 2014)	X			X				X	6	2	X
(MOU; WANG; SEET, 2014)		X		X				X	6	3	X
(BASTANLAR, 2014)		X		X		X		X	6	?	
(PERSSON et al., 2015)		X		X		X		X	6	3	?

Table 1: Stereo visual odometry algorithms.

□ P.A. - Point alignment type

□ D.E. - Data extraction type

– F. - Features

– A. - Appearance

□ I.R. - Iterative refinement

□ K.F. - Kalman filter

□ S. - Scale is retrieved

□ DoF - Degrees of freedom of the estimated motion

□ N.Pts. - Number of points required to estimate motion

□ R.T. - Achieves real-time performance

3.2.2 Conventional Monocular Visual Odometry

Estimating the motion of a single, conventional perspective camera is known as monocular visual odometry. One way to classify visual odometry methods is through the way visual information is perceived. This can be done by selecting separate features (feature-based), by considering the whole image (appearance-based) or by combining both approaches (hybrid methods). Both techniques are explained in Chapter 2, Section 2.1.4. The next sections enumerate some examples for each approach. Some references referred here are indicated in (SCARAMUZZA; FRAUNDORFER, 2011).

3.2.2.1 Feature-based Methods

As the name suggests, in feature-based methods the images are scanned for sparse features, which are then used to estimate motion. Features may take many forms: they may be corners (crossing between two edges), lines (commonly found in indoor environments), or simply small regions that are easily distinguished from their immediate neighborhood (blobs).

NISTER; NARODITSKY; BERGEN (2004) solves a 3D-to-2D point alignment problem with Harris corners. Motion is estimated using the 5-point algorithm, preemptive RANSAC and an iterative refinement.

MOURAGNON et al. (2006) not only performs real-time localization, but also reconstructs 3D points. Frames that are distant from each other, but still maintain a common minimum number of features, are selected as keyframes. Everytime a keyframe appears, 3D points are computed and a local bundle adjustment is performed. Motion is estimated between the last acquired image and the last keyframe using the 5-point algorithm on Harris corners, followed by RANSAC. The reprojection error is minimized, using the previously triangulated 3D points.

The work in (DAVISON et al., 2007) proves, for the first time, that it is possible to perform real-time SLAM with a single hand-held camera. The method uses a probabilistic map containing sparse but persistent features, motion estimates and the uncertainty associated to them. As the camera moves and new features appear, the map is updated by an Extended Kalman filter (EKF). Features consist of large image patches and their matching is supported by the available localization

information. In order to eliminate drift, loop closure is performed. This approach is meant to be used in Augmented Reality (AR).

KLEIN; MURRAY (2007) suggests a tracking and mapping method which is also suitable for AR applications. This approach assumes no previous knowledge about the scene and is also meant to be applied on a hand-held camera. The method consists of two tasks, each one handled by an individual core. In the mapping task, the 3D position of thousands of landmarks is computed using the 5-point algorithm and bundle adjustment. Here, a dominant plane is detected, for instance the floor or the top of a table, in order to place virtual characters on it. The tracking task estimates the motion of the camera by reprojecting the computed 3D points in the images. The fact that each task has a dedicated core allows the extraction of a large number of points, achieving a robust and accurate mapping, without interfering on the real-time constraints of the tracking task.

The method in (SCARAMUZZA, 2011) requires only one point, resulting in good planar motion estimates even if the environment contains few features, many independently moving objects or is cluttered. For outlier removal, the algorithm uses RANSAC, which is very efficient since it only uses one point, and histogram voting. The performance of these techniques is comparable to the one of the 5-point algorithm, as both produce a similar number of inliers. In case the flat road assumption is not satisfied, this is detected through the computed histograms, and the algorithm switches to RANSAC with five points.

KNEIP; CHLI; SIEGWART (2011) estimates the pose of a Micro Aerial Vehicle (MAV) using a monocular camera and an IMU. Frames with enough disparity between them are selected as keyframes. For every pair of keyframes, a 3D point cloud of common features is computed. For every non-keyframe, the camera pose is computed using the 3D points obtained in the previous keyframe. The IMU provides information on orientation, allowing translation to be achieved with only two points, using RANSAC. Pose is estimated minimizing the reprojection error and refined with an iterative non-linear optimization method.

3.2.2.2 Appearance-based Methods

Appearance-based methods use all intensity information contained in the images, and are, therefore, dense methods. In order to estimate motion, whole image regions are compared by a correlation method.

GOECKE et al. (2007) uses the Fourier-Mellin Transform (FMT) to estimate the motion between consecutive views. Here, the Fourier transform of the images is efficiently computed using the Fast Fourier Transform (FFT) and the Fourier shift and rotation theorems are applied to recover translation and rotation respectively. The camera is fixed on a vehicle's roof, tilted down, at a fixed height. The fact that these distances are fixed and known is used to map the camera coordinates to the ground plane, allowing the retrieval of depth.

The work in (MILFORD; WYETH, 2008) consists of a visual odometry pipeline inspired in natural phenomena, namely the localization logic of rodents and bees. Here, the goal is not to understand the geometric structure of the scene. Instead, visual information is used to determine the rotation and speed of the camera. Rotation is recovered in the following way: the columns within a region of the images are concatenated to form a one-dimensional array. The array of one image is compared to the shifted array of the next image. The shift which provides the smallest difference is used to compute rotation. Speed estimation is based on the rate of image change. Finally, the algorithm stores the computed arrays, in order to recognize already visited places. This method, known as Template Learning, allows closing loops and, thereby, accumulated errors are eliminated.

NEWCOMBE; DAVISON (2011) suggests an appearance-based method to perform simultaneous tracking of a hand-held RGB camera and dense 3D reconstruction of the environment. The algorithm provides a detailed textured depth map, obtained by a non-convex optimization framework. Motion estimation is achieved by aligning the whole images against the 3D map.

3.2.2.3 Hybrid Methods

Hybrid methods combine both feature- and appearance-based approaches.

The work in (FORSTER; PIZZOLI; SCARAMUZZA, 2014) combines the advantages of feature- and appearance-based methods. While feature-based strategies allow

the parallel tracking of many features, mapping and the selection of keyframes, appearance-based ones are faster and more accurate. This approach counts on two parallel threads, namely a mapping and a motion estimation thread. For each keyframe, the mapping thread extracts features and estimates their depth through a Bayesian filter. The outcome is a set of reliable 3D points with few outliers. In the motion estimation thread, the photometric error between the computed 3D points is minimized. Afterwards, several small image patches are aligned in order to refine the pixel locations corresponding to the 3D points. Finally, the reprojection error is minimized. This algorithm is known as SVO, for Semi-Direct Visual Odometry, and it is available as an open-source software and is currently considered a reference method.

3.2.2.4 Summary - Conventional Monocular Visual Odometry

To conclude this section, Table 2 summarizes the main characteristics of the mentioned monocular visual odometry methods.

Furthermore, KNEIP; CHLI; SIEGWART (2011) uses an IMU to complement visual odometry and DAVISON et al. (2007) and MILFORD; WYETH (2008) do loop closure. Planar motion constraint is assumed in (GOECKE et al., 2007), (MILFORD; WYETH, 2008) and (SCARAMUZZA, 2011).

Summarizing monocular visual odometry works, most authors minimize the 3D-to-2D point alignment error and use feature-based methods. The majority of works perform iterative refinement and very few make use of a Kalman filter. Most authors determine translation scale, either by complementing visual odometry with estimates of other sensors or by constraining the motion, and therefore retrieving less than 6 DoF.

	3D-3D	P.A. 3D-2D	2D-2D	D.E.		I.R.	K.F.	S.	DoF	N.Pts.	R.T.
				F.	A.						
(NISTER; NARODITSKY; BERGEN, 2004)		X		X		X			6	5	X
(MOURAGNON et al., 2006)		X		X		X			6	3?	X
(DAVISON et al., 2007)			X?	X			X	X	6	?	X
(GOECKE et al., 2007)			X		X			X	3?	-	
(KLEIN; MURRAY, 2007)		X		X		X	X	X	6	5	X
(MILFORD; WYETH, 2008)	-	-	-		X			X	3	-	X
(SCARAMUZZA, 2011)		X		X				X	3	1	X
(KNEIP; CHLI; SIEGWART, 2011)		X		X		X		X	6	2	X
(NEWCOMBE; DAVISON, 2011)		X			X	X			6	-	X
(FORSTER; PIZZOLI; SCARAMUZZA, 2014)		X		X	X	X		X	6	?	X

Table 2: Conventional monocular visual odometry algorithms.

❑ P.A. - Point alignment type

❑ D.E. - Data extraction type

– F. - Features

– A. - Appearance

❑ I.R. - Iterative refinement

❑ K.F. - Kalman filter

❑ S. - Scale is retrieved

❑ DoF - Degrees of freedom of the estimated motion

❑ N.Pts. - Number of points required to estimate motion

❑ R.T. - Achieves real-time performance

3.2.3 Omnidirectional Monocular Visual Odometry

The next section briefly describes some works where visual odometry is computed using one omnidirectional system. This includes, for instance, a catadioptric camera or a set of several cameras, pointing to different parts of the scene. The latter systems are considered monocular because the FoVs of the cameras do not overlap, thus providing no disparity information.

3.2.3.1 Central Cameras

The only requirement for a system to be central is that all light rays converge into one single viewpoint. This is always true for simple perspective cameras, but harder to achieve for omnidirectional systems. Nevertheless, a lot of work has been carried out on central omnidirectional visual odometry.

In order to require less points to compute the camera's egomotion, many authors have opted for using a planar motion constraint (CORKE; STRELOW; SINGH, 2004; SCARAMUZZA; SIEGWART, 2008; SCARAMUZZA; FRAUNDORFER; POLLEFEYS, 2010; HOANG; HERNÁNDEZ; JO, 2013). Such approaches rely on the fact that, for wheeled vehicles, the camera height relatively to the ground is constant and known. Points on the road are usually used and the ground is approximated by a plane, as shown in Figure 19.

Figure 19: Planar motion assumption.

In addition to using the planar motion constraint, SCARAMUZZA; SIEGWART (2008), SCARAMUZZA; FRAUNDORFER; POLLEFEYS (2010) and HOANG; HERNÁNDEZ; JO (2013) also use image appearance to effectively estimate rotation. The term "appearance" here means that all pixels of the image are used in the so-called **visual compass**. First, the catadioptric images (Figure 20a) are unwrapped into rectangles (Figure 20b). A region of interest, which usually includes the front part

of the camera (middle of the rectangle), is considered. The shift of the pixels in that region is directly related to the rotation undergone by the catadioptric camera. In Figure 20c, let the shift be s pixels. If the rectangle has a total width of w pixels, the relative rotation $\Delta\theta$ is computed as follows:

$$\Delta\theta = s \cdot \frac{360^\circ}{w} \quad (30)$$

Figure 20: Visual compass. Grey: road. Green: surroundings of the road. Red: front side of the road. Blue: back side of the road. Black: mirror support. Yellow square: region of interest.

In (CORKE; STRELOW; SINGH, 2004) the authors suggest two methods to perform visual odometry with a central catadioptric camera fixed on a planetary rover. The first method consists of robust optical flow. Here, the ground is approximated by a plane and the algorithm computes the 3 DoF motion of the vehicle, its velocity and the camera's intrinsic parameters. The second method, an online structure-from-motion algorithm, uses an Iterated Extended Kalman filter (IEKF) and computes the full 6 DoF motion, as well as the 3D position of landmarks. For both methods, features are tracked along multiple frames using Lucas-Kanade. While the structure-from-motion method provides more accurate motion estimation than the optical flow method, it also requires a larger computational effort, as well as a careful calibration of the catadioptric configuration.

TARDIF; PAVLIDIS; DANIILIDIS (2008) uses a multi-camera system (a Ladybug 2) alone to predict the motion of a vehicle. Strictly speaking, this sensor is a multi-camera system with unaligned camera centers, so it should be classified as a

non-central sensor. However, as the distance between camera centers is very small compared to the distance from the observed world points, Ladybug 2 behaves like a central system, and may be modelled as one. The main novelty of this work is the decoupling of rotation from translation. While translation is obtained by the 3D map of landmarks, which they also compute via triangulation, rotation is retrieved through the epipolar geometry between previous and current images of corresponding points. The epipolar constraint ensures that, even with a small number of correspondences, there is no error propagation. No iterative optimization such as bundle adjustment is performed, keeping the algorithm simple and fast. The authors obtain an accurate trajectory estimation over a distance of 2.5 km.

SCARAMUZZA; SIEGWART (2008) proposes an algorithm for estimating the planar motion of a vehicle based on images of a central catadioptric camera fixed on its roof. Scale invariant features on the ground plane are matched and a homography matrix is obtained and decomposed into translation and rotation. Since this rotation estimation is highly sensitive to calibration errors, appearance is used in a visual compass, as previously explained. In order to neglect the effects of translation on the optical flow, only the part of the images corresponding to the front and back views of the vehicle are considered. The authors prove that their algorithm provides an accurate motion estimation for outdoor environments and long distances.

SCARAMUZZA; FRAUNDORFER; POLLEFEYS (2010) computes the relative motion between consecutive frames using features and improves the rotation estimation by using a similar technique as SCARAMUZZA; SIEGWART (2008). Furthermore, the authors also developed a closing loop scheme based on visual words. This efficient and highly scalable method reduces the visual odometry drift in real-time. Here, the reprojection error is minimized in a bundle adjustment framework.

LE et al. (2012) requires one point and one edge to compute the 5 DoF motion of the vehicle (3D position, yaw and pitch). While yaw is computed using the Chamfer method, pitch is achieved by transforming part of the omnidirectional images into perspective ones. The translation scale is retrieved through wheel odometry. Furthermore, in an EKF framework, GPS is used as a reference for visual odometry. In case one of the sensors fails, the other is used alone.

HOANG; HERNÁNDEZ; JO (2013) uses an EKF, where the state consists of rotation and translation vectors, and constant velocity is assumed. In the prediction step, an estimate of rotation is obtained by matching the edges of the road and using them as a visual compass. In the observation step, one point is used in a RANSAC framework to compute the epipolar geometry and to estimate the mapped landmarks. Again, the visual compass leads to accurate rotation estimation, even in cluttered scenes with different illumination conditions. This technique is also suitable for real-time applications.

LE et al. (2013) uses a similar approach as LE et al. (2012), but does not use wheel odometry.

In (HOANG et al., 2013), visual odometry is combined with a laser rangefinder in a Kalman filter framework. The rangefinder predicts the 6 DoF motion and the error is corrected using the epipolar constraint between consecutive frames.

3.2.3.2 Non-central Cameras

Because of the limiting design restrictions of central systems, much research has been done on non-central omnidirectional systems. Several authors, namely STEWENIUS et al. (2005), LI; HARTLEY; KIM (2008), LEE; FAUNDORFER; POLLEFEYS (2013) and VENTURA; ARTH; LEPETIT (2015) adopted the General Imaging Model. There is also a group of works that focus specially on multi-camera systems, such as (STEWENIUS; ÅSTRÖM, 2004; KIM et al., 2007; CLIPP et al., 2008; LEE; FAUNDORFER; POLLEFEYS, 2013; KNEIP; LI, 2014; VENTURA; ARTH; LEPETIT, 2015). Minimal solutions are analyzed in (STEWENIUS; ÅSTRÖM, 2004; STEWENIUS et al., 2005; LEE; FAUNDORFER; POLLEFEYS, 2013; VENTURA; ARTH; LEPETIT, 2015).

STRELOW et al. (2001) suggests two methods for structure and motion estimation, namely an offline technique which uses all frames simultaneously and an online approach which computes the relative motion as new frames appear. The authors also suggest a calibration method which improves the accuracy of motion estimations. 2D features are tracked using the Lucas-Kanade algorithm. Afterwards, the 3D position of the points is determined and the reprojection error is minimized. In the online method, an EKF is employed, where the state consists of the 6 DoF camera relative position and the 3D position of each point. In order

to avoid using points that are mistracked, become occluded or belong to independently moving objects, motion is computed in an iterative framework, where points which fit the model with an error larger than a threshold are discarded in every iteration.

STEWENIUS; ÅSTRÖM (2004) explores several minimal cases, focusing on the one requiring three images and six points. Here, relative motion is computed through a trifocal tensor. This method is suitable for RANSAC and real-time applications.

STEWENIUS et al. (2005) uses six corresponding points in two images. Constraining that the 3D viewing rays of corresponding points must meet in space, 64 solutions are obtained. A 64-by-64 matrix is computed in a closed-form fashion and its eigenvalues are extracted. The minimal nature of this algorithm allows it to be used within a RANSAC framework.

In (KIM et al., 2007) the egomotion of a set of firmly mounted cameras is predicted. For all cameras, the essential matrix is computed, by finding five temporal correspondences in the images, and decomposed, providing the rotation and translation of all cameras. The rotation of the rig is obtained by averaging all cameras' rotations. The vectors with the translation and rotation are used to find vectors (one for each camera) that intercept at the new position of one of the cameras. This is a triangulation problem that is solved using the L_{inf} norm. In case motion possesses no rotation, the lines do not cross, and the scale of the translation cannot be retrieved.

In (LI; HARTLEY; KIM, 2008) the authors notice that a well-known algorithm which uses 17 correspondences cannot be linearly solved for many common camera configurations. They propose a simple, efficient and mainly linear algorithm to solve the same problem, achieving a reasonable accuracy.

CLIPP et al. (2008) proposes an algorithm for predicting the egomotion of a vehicle with two cameras mounted at each side of the roof of a vehicle, with a known configuration. Since the cameras do not overlap, there is no spacial baseline between the two cameras, but a temporal one. The algorithm requires five temporal correspondences in one of the cameras and one correspondence in the other one. After applying the 5-Point Algorithm for the first camera, the rotation and the translation direction of the first camera is obtained. Because the scale of

the translation is unknown, there is a line in 3D space where the second camera can be. By using one temporal feature correspondence on the second camera, it is possible to recover the scale of the translation, as long as the motion is not critical. Such critical conditions happen when there is no - or a too smooth - rotation. This algorithm can be easily expanded for a larger number of cameras.

In (KAZIK et al., 2012), the 6 DoF relative motion of a pair of opposite-facing cameras is computed. This real-time method is tested indoors, in difficult environments. First, monocular visual odometry is performed separately for both cameras. Everytime a keyframe arrives, bundle-adjustment is employed, minimizing the reprojection error. As long as motion is not degenerate, scale is retrieved, as the rigid transformation between the two cameras is known. Degenerate motion cases are detected and excluded in order to avoid propagating erroneous scales. Finally, both odometry estimates are statistically fused via an EKF, leading to optimal results.

LEE; FAUNDORFER; POLLEFEYS (2013) computes the non-holonomic motion of a vehicle with four cameras fixed on the roof, with small FoV overlapping. Applying the Ackermann steering principle, the generalized epipolar constraint is simplified and motion may be obtained with two corresponding points. Two cases are studied, namely the case where at least one inter-camera correspondence is available and the case where there are only intra-camera correspondences. For both situations, a maximum number of six solutions is obtained. A method to retrieve scale in degenerate motions (straight motions with no rotation) is proposed, where an additional inter-camera correspondence is required. Motion is estimated within a Kalman filter framework, where velocity is assumed to be constant, and loop closure is also carried out. Motion and structure estimations are refined via bundle adjustment. The fact that only two points are required to compute motion allows this approach to be used in real-time applications and within a RANSAC framework.

The work in (KNEIP; LI, 2014) deduces a translation independent epipolar plane coplanarity constraint for general cameras. They also perform an iterative rank minimization on rotation. This work presents a number of advantages, such as a constant computational complexity, making it suitable for real-time applications, robustness to noise and the ability to provide an unique solution, as long as there

are at least seven omnidirectional correspondences. The achieved computer complexity is comparable to the one of linear solvers, while the accuracy is similar to the one obtained with bundle adjustment. The generality of this method makes it suitable for any camera type, distribution and combination of correspondences.

VENTURA; ARTH; LEPETIT (2015) uses a first-order approximation to the relative pose problem and solves the polynomials efficiently. This approach leads to a similar accuracy as state of art methods, but a faster performance, allowing it to be employed in real-time applications and within RANSAC frameworks.

3.2.3.3 Summary - Omnidirectional Monocular Visual Odometry

Omnidirectional visual odometry projects are summarized in Table 3. Characteristics that are specific to omnidirectional vision are summarized in Table 4.

Furthermore, LE et al. (2012) also uses a GPS and wheel odometry, HOANG et al. (2013) uses a laser rangefinder and LE et al. (2013) uses a GPS.

The planar motion constraint is imposed in (CORKE; STRELOW; SINGH, 2004) (in the first proposed algorithm), (STEWENIUS; ÅSTRÖM, 2004), (SCARAMUZZA; SIEGWART, 2008), (SCARAMUZZA; FRAUNDORFER; POLLEFEYS, 2010), (LEE; FAUNDORFER; POLLEFEYS, 2013), (HOANG; HERNÁNDEZ; JO, 2013) and (LE et al., 2013) (planar ground with slope).

The work in (SCARAMUZZA; FRAUNDORFER; POLLEFEYS, 2010) and (LEE; FAUNDORFER; POLLEFEYS, 2013) also perform loop closure recognition.

Like in the monocular and stereo case, most omnidirectional visual odometry works adopt a 3D-to-2D point alignment approach, as well as feature-based methods. The majority of works retrieve the 6 DoF relative pose (i.e. including translation scale), as long as motion is not degenerate. Furthermore, most works achieve real-time performance.

	3D-3D	P.A. 3D-2D	2D-2D	D.E.		I.R.	K.F.	S.	DoF	N.Pts.	R.T.
				F.	A.						
(STRELOW et al., 2001)		X		X			-,X		6	?	-,X
(STEWENIUS; ÅSTRÖM, 2004)		X		X				X*	6	6	X
(CORKE; STRELOW; SINGH, 2004)		X		X			-,X	X	3,6	?	X
(STEWENIUS et al., 2005)	X			X				X*	6	6	?
(KIM et al., 2007)	X			X		X		X*	6	5	?
(LI; HARTLEY; KIM, 2008)	?	?	?	X		X		X*	6	17	X
(TARDIF; PAVLIDIS; DANIILIDIS, 2008)		X		X				X	6	7	X
(SCARAMUZZA; SIEGWART, 2008)		X		X	X			X	6	4	X
(CLIPP et al., 2008)		X?	?	X		X	X	X*	6,5	6	?
(SCARAMUZZA; FRAUNDORFER; POLLEFEYS, 2010)		X		X	X	X		X	3	2	X
(KAZIK et al., 2012)		X		X		X	X	X*	6	?	X
(LE et al., 2012)	?	?	?		X		X	X	3	?	X
(LEE; FAUNDORFER; POLLEFEYS, 2013)		X		X		X	X	X	6	2	X
(HOANG; HERNÁNDEZ; JO, 2013)	?	?	?	X	X		X	X	3	1+e.	X
(LE et al., 2013)	?	?	?	X	X		X	X	5	1+e.	
(HOANG et al., 2013)	?	?	?	X			X	X	6	?	
(KNEIP; LI, 2014)		X		X		X		X*	6	7	X
(VENTURA; ARTH; LEPETIT, 2015)	X			X				X*	6	6	X

Table 3: Omnidirectional monocular visual odometry algorithms. Symbols: * - as long as the motion is not degenerate. ° - if the motion is degenerate. 1+e. - one point and one edge.

□ P.A. - Point alignment type

□ D.E. - Data extraction type

– F. - Features

– A. - Appearance

□ I.R. - Iterative refinement

□ K.F. - Kalman filter

□ S. - Scale is retrieved

□ DoF - Degrees of freedom of the estimated motion

□ N.Pts. - Number of points required to estimate motion

□ R.T. - Achieves real-time performance

	Centrality		Multi-camera	General Imaging Model
	Central	Non-central		
(STRELOW et al., 2001)		X		
(STEWENIUS; ÅSTRÖM, 2004)		X	X	
(CORKE; STRELOW; SINGH, 2004)	X			
(STEWENIUS et al., 2005)		X		X
(KIM et al., 2007)		X	X	
(LI; HARTLEY; KIM, 2008)		X		X
(TARDIF; PAVLIDIS; DANIILIDIS, 2008)	X		X	
(SCARAMUZZA; SIEGWART, 2008)	X			
(CLIPP et al., 2008)		X	X	
(SCARAMUZZA; FRAUNDORFER; POLLEFEYS, 2010)	X			
(KAZIK et al., 2012)		X	X	
(LE et al., 2012)	X			
(LEE; FAUNDORFER; POLLEFEYS, 2013)		X	X	X
(HOANG; HERNÁNDEZ; JO, 2013)	X			
(LE et al., 2013)	X			
(HOANG et al., 2013)	X			
(KNEIP; LI, 2014)		X	X	X
(VENTURA; ARTH; LEPETIT, 2015)		X	X	X

Table 4: Omnidirectional vision specific characteristics of algorithms.

Chapter 4

Stereo Visual Odometry

This chapter describes the existing stereo visual odometry algorithm Libviso2 (GEIGER; ZIEGLER; STILLER, 2011), which is briefly described in Chapter 3, Section 3.1.1.2. Since the scope of this project does not include 3D reconstruction, this chapter will solely expand on the stereo visual odometry.

This visual odometry algorithm estimates the 6 DoF motion of a vehicle through the video captured by a camera pair fixed on the roof, using only $N = 3$ features. The C++ library Libviso2, openly provided by the authors, is employed, as well as the wrapper for ROS developed by (WIRTH, 2015).

4.1 Feature Extraction and Matching

In order to extract features, the rectified grayscale images are filtered with 5x5 pixel blob and corner detectors. Non-maximum and non-minimum suppression is performed over the result, yielding to blob minima, blob maxima, corner minima and corner maxima features. For efficiency, only features within these classes are matched. Assuming a smooth trajectory, i.e. without abrupt feature rotations, computationally expensive methods as SIFT are avoided. Instead, 11x11 pixel blocks are filtered with a horizontal and vertical Sobel edge detector and compared using the sum of absolute differences (Figure 21).

Features are matched in a "circle" (Figure 22): starting with the current left image, features are matched to the previous left image, then the previous right

Figure 21: Flowchart with overview of Libviso2 feature matching steps.

image, following the current right image, and back to the current left image. If this feature coincides with the starting feature, it is retained.

Figure 22: Libviso2 "circle" feature matching procedure. Top: Left (L) and right (R) images captured in the previous instant ($k-1$). Bottom: Left and right images captured in the current instant (k).

The epipolar constraint is used to match features between left and right images. 2D Delaunay triangulation is used in order to remove outliers. Here, two matches support each other if the disparity and flow differences between them are less than

specified thresholds. At this point, several thousands of features are available, which would lead to long matching times. In order to speed up feature matching, the number of features is reduced by using a technique developed in their previous work (GEIGER; ROSER; URTASUN, 2010)

4.2 Visual Odometry

Using the previous frame $k-1$, 3D points are triangulated out of corresponding features. Expressing the resulting 3D points in the coordinate frame of the left camera at instant $k-1$, feature points are reprojected in the current (k) frame as follows:

$$\hat{\tilde{x}}_{L,k} = F_L(X_{k-1}, r, t) = \begin{bmatrix} f & 0 & o_x \\ 0 & f & o_y \\ 0 & 0 & 1 \end{bmatrix} \cdot [R(r) \quad t] \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (31)$$

$$\hat{\tilde{x}}_{R,k} = F_R(X_{k-1}, r, t) = \begin{bmatrix} f & 0 & o_x \\ 0 & f & o_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \left([R(r) \quad t] \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} - \begin{bmatrix} s \\ 0 \\ 0 \end{bmatrix} \right) \quad (32)$$

where $\hat{\tilde{x}}_{L,k}$ and $\hat{\tilde{x}}_{R,k}$ are the estimated image pixels in the current left and right images, in homogeneous coordinates, X_{k-1} (or $[X, Y, Z, 1]^T$) is the corresponding 3D point triangulated in the previous frame, f is the focal length, (o_x, o_y) is the principal point, $r = [r_x, r_y, r_z]^T$ is a vector with the rotation angles about the x-, y- and z-axis, and t is the translation vector. s is the baseline distance between the left and right cameras. Since the world points are described in the left camera coordinates, this term does not appear in the reprojection to the current left image. The following expression (the reprojection error) is minimized with respect to r and t , using iterative Gauss-Newton optimization:

$$\sum_{i=1}^N \|\tilde{x}_{L,k,i} - F_L(X_{k-1,i}, r, t)\| + \|\tilde{x}_{R,k,i} - F_R(X_{k-1,i}, r, t)\| \quad (33)$$

where $\tilde{x}_{a,k-1,i}$ is the homogeneous pixel point of the i^{th} observed feature in camera a (L (left) or R (right)) in frame $k - 1$. N is the number of features. Even if r and t are initialized with zeros, they converge in 4 to 8 iterations.

The estimation of the rotation and translation vectors is performed within a RANSAC framework in order to ensure that no features of independently moving objects are being considered. r and t are computed 200 times using 3 random, independently selected features. Inliers of the iteration where the most features fit the obtained model are used to compute the final transformation.

Chapter 5

Visual odometry for an omnidirectional multi-camera system

In this chapter, the omnidirectional multi-camera visual odometry approach is explained in detail. This feature-based method consists of first running a monocular algorithm using each camera's image and choosing the best one through a voting scheme.

5.1 Feature extraction and matching

In each of the rectified grayscale images, FAST features are extracted (ROSTEN; DRUMMOND, 2006). This algorithm determines whether a pixel is or not a corner in the following manner: the intensity difference between the pixel to be tested and pixels located at a specified radius around it is analyzed. If a number of surrounding and consecutive pixels is considerably brighter or darker than the middle pixel, then the middle pixel is considered a corner. Efficient implementations have been carried out, by not considering all pixels in the circle, but only a few. Following these steps, it may happen that adjacent pixels are also considered corners. This problem is solved by assigning a score to each pixel, which is the intensity difference to the surrounding pixels. If several adjacent pixels are classified

Figure 23: FAST feature detection. Pixel p is considered a corner, as the number of consecutive surrounding pixels with a large intensity difference to p is larger than a threshold. Image taken from (ROSTEN; DRUMMOND, 2006).

as corners, only the one with the highest score remains.

The detected features are tracked using the Lucas-Kanade method (LUCAS, 1981). This approach assumes that the intensity of corresponding pixels from consecutive frames does not change and that direct neighbors of a pixel suffer the same displacement. The displacement of each pixel is retrieved, assuming small motions. Bigger motions are handled by gradually down-sampling the images.

As the camera moves, features tend to disappear in the image's borders. Once the number of inlier features drops below 50, a new FAST feature extraction is carried out.

The feature extraction and matching implementation uses the functions developed by (SINGH, 2015), who uses the available OpenCV functions.

5.2 Visual odometry

The omnidirectional multi-camera visual odometry algorithm is obtained by having each camera estimate motion using a monocular approach and carrying out a voting scheme, selecting the best estimate. The idea is to take advantage of the fact that several estimations are available, and that some might be better than others. As the vision system is omnidirectional, the richest part of the environment for feature extraction is always available, regardless of its position. The following sections address the chosen monocular approach and the voting scheme.

5.2.1 Monocular visual odometry algorithm

Initially, all available points are normalized, such that their centroid is displaced to zero and the average distance of the points to the origin is $\sqrt{2}$. This normalization is a crucial step in any least squares method, avoiding numerical errors due to ill-conditioned matrices.

The 8-Point algorithm (HARTLEY, 1997) was chosen as the monocular method due to its simplicity and ease to perceive and implement. It determines the rotation and translation of the camera by first calculating the fundamental matrix F . Using N (at least eight) corresponding points in the previous and current images, the $N \times 9$ matrix A is constructed, with each row being in the following form:

$$A_i = \begin{bmatrix} x_{i,k} \cdot x_{i,k-1} \\ x_{i,k} \cdot y_{i,k-1} \\ x_{i,k} \\ y_{i,k} \cdot x_{i,k-1} \\ y_{i,k} \cdot y_{i,k-1} \\ y_{i,k} \\ x_{i,k-1} \\ y_{i,k-1} \\ 1 \end{bmatrix}^T, \quad (34)$$

for corresponding points $p_{i,k-1} = [x_{i,k-1}, y_{i,k-1}]$ and $p_{i,k} = [x_{i,k}, y_{i,k}]$, taken at consecutive instants $k-1$ and k . The SVD of A is computed and F is formed by the elements of the eigenvector corresponding to the lowest eigenvalue of A .

This runs within a RANSAC framework: in each iteration, 8 feature matches are randomly chosen and the fundamental matrix is built out of those points. It is tested against all matches by computing the Sampson cost.

$$d_S = \frac{\tilde{p}_{i,k}^T F \tilde{p}_{i,k-1}}{(F \tilde{p}_{i,k-1})_x^2 + (F \tilde{p}_{i,k-1})_y^2 + (F^T \tilde{p}_{i,k})_x^2 + (F^T \tilde{p}_{i,k})_y^2} \quad (35)$$

where the tilde denotes homogeneous coordinates and the subscripts x and y indicate the first and second components of the vector (HARTLEY, 1997). Given a point in the previous image, this cost is the distance from the actual current corresponding point to the epipolar line generated by F in the current image (Figure

Figure 24: Sampson distance: distance from current observed pixel $p_{im,k}$ to epipolar line generated by F .

24). If the Sampson error of a match is lower then 0.0001, the match is considered an inlier. At the end of the RANSAC loop, the fundamental matrix with the most inliers is refined using all agreeing points.

After re-enforcing rank-2 to the resulting optimized fundamental matrix, the essential matrix E is obtained with

$$E = M_{int}^T \cdot F \cdot M_{int} \quad (36)$$

and rotation R and translation t are extracted from E in the following way: assuming matrices

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad Z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad (37)$$

rotation candidates R_a and R_b and the skew symmetric matrix of translation, S , are calculated with

$$R_a = U \cdot W \cdot V^T \quad (38)$$

$$R_b = U \cdot W^T \cdot V^T \quad (39)$$

$$S = U \cdot Z \cdot U^T, \quad (40)$$

where U and V are the outcomes of the SVD of E . If R_a or R_b have a negative determinant, its sign must be inverted. The components t_x , t_y and t_z can be directly extracted from S , as it is formed as

$$S = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}. \quad (41)$$

Possible rotations are R_a and R_b and possible translations are t and $-t$, leading to four possible combinations. Further testing must therefore be carried out to obtain an unique solution. A common way to test the validity of the solutions is by applying the Chirality constraint (HARTLEY, 1997), according to which all points must be in front of both the previous and current cameras. Thus, for each solution, all matches are triangulated and it is checked whether the Z component of the obtained 3D points is positive. The solution with the most valid points is chosen.

Both rotation and translation returned by the 8-Point algorithm define the pose of the camera in the previous instant described in coordinates of the camera in the current instant. Consequently, assuming that ${}^k_{k-1}T$ is the transformation containing the rotation and translation, it must be inverted when integrated to obtain the absolute pose.

$${}^0_kT = {}^0_{k-1}T \cdot {}^{k-1}_kT = {}^0_{k-1}T \cdot ({}^k_{k-1}T)^{-1} \quad (42)$$

Except for some details, the implementation of monocular visual odometry was guided by (GEIGER; ZIEGLER; STILLER, 2011), the monocular version.

5.2.2 Voting scheme

If the omnidirectional multi-camera system has N_{cams} individual cameras, the previous step returns N_{cams} relative pose estimations, computed with the feature matches found in each camera in previous and current frames. The voting scheme consists of having each of the N_{cams} cameras vote for each of the N_{cams} available

motion estimates, ${}_k^{x,k-1}T$, where x is the camera providing that estimate. For sake of readability, the expression ${}_k^{x,k-1}T$ is shortened as xT in this chapter.

Let us suppose that camera i is testing camera j 's estimate. First, the transformation jT provided by camera j (and consequently described in camera j 's coordinates) must be converted to camera i 's coordinates. This is done through the following expression:

$${}^iT = {}^iT \cdot {}^jT \cdot {}^j_iT \quad (43)$$

where ${}_b^aT$ is the rigid transformation of camera b relatively to camera a .

Assuming that iR_j and it_j are the rotation and translation provided by camera j and described in coordinates of camera i , a fundamental matrix is built using the resulting transformation and camera i 's intrinsic parameters.

$$F_{ij} = M_i \cdot [{}^iR_j | {}^it_j]^{-1} \cdot M_i^{-1} \quad (44)$$

The Sampson cost is calculated for F_{ij} and all point matches found in camera i . A match with an error below 0.0001 counts as a vote for camera j 's relative pose estimate. This value was obtained experimentally, based on the number and quality of the consequent inliers. The rotation and translation with the most votes is the output of the algorithm. The voting scheme is illustrated as a flowchart in Figure 25.

Since the multi-camera system used in this project has its individual cameras close together, it behaves like a central system and scale cannot be reliably retrieved. The translation norm is corrected by forcing it to be equal to the ground truth translation norm at corresponding time intervals.

$$t_{vo, corrected\ scale} = t_{vo} \cdot \frac{\|t_{gt}\|}{\|t_{vo}\|} \quad (45)$$

where vo and gt stand for "visual odometry" and "ground truth", respectively, and for the same time interval.

Figure 25: Flowchart of the voting scheme. Red box: Camera i evaluates estimate T_j . Green box: T_j with the most inliers is selected.

Chapter 6

Experiments and results

This chapter is dedicated to all experimental work carried out for this project. It describes the experimental setup and procedure, the analysis of the results, the used equipment and, finally, the results obtained with both stereo and omnidirectional multi-camera approaches.

6.1 Experimental procedure

Experiments were carried out using an autonomous vehicle, Carina2, which is a small, urban vehicle. The autonomous car is an ongoing project held by the Mobile Robotics Laboratory (LRM), which is a research group within the Institute of Mathematical and Computer Sciences (ICMC), which belongs to the University of São Paulo (USP).

In order to test the visual odometry algorithms, both Ladybug 2 and Bumblebee XB3 cameras are attached to the roof of the vehicle (Figure 26). The former is placed on top of a pole, preventing the top of the car to occupy a large portion of the images. The Ladybug reference frame has its x -axis pointing to the front. The stereo camera is also placed at a height above the roof and faces the front.

Figure 27 illustrates how the reference frames of the vehicle, Ladybug 2 (suffix *LB*) and the Bumblebee XB3 (suffix *BB*) cameras relate. The Ladybug camera and the vehicle reference frames share the same orientation. The reference frame of the Bumblebee camera, however, has a different orientation. Consequently, the

Figure 26: Left: Autonomous vehicle with mounted Ladybug 2 and Bumblebee XB3 cameras. Left: Scenario 1. Right: Highlighted cameras.

stereo odometry resulting transforms must be rotated, such that their orientation matches the one of the vehicle frame. The rigid relative position between the vehicle and the cameras is not relevant in this project, since the relative pose estimations of the methods do not depend on where exactly the cameras are relatively to the vehicle. It is assumed that all three frames have their origin at $[0, 0, 0]^T$.

Figure 27: Vehicle, Ladybug 2 and Bumblebee XB3 reference frames.

6.1.1 Scenarios

The average velocity of the vehicle is 23.5 km. Two scenarios, shown in Figure 28, are analyzed.

Figure 28: Experimental trajectories. Left: Scenario 1. Right: Scenario 2.

In scenario 1 the vehicle traveled approximately 0.75 km and in scenario 2 around 1.5 km. The labeled markers along the trajectory aim at helping associating sections of the relative motion error graphs to parts of the trajectory. Both trips were carried out in the same place, namely Campus 2 of University of São Paulo, in the city of São Carlos. The environment includes parked cars, trees at the sides of the road and some buildings. Some parts of the scenario do not contain nearby elements. Figure 29 shows a few photos captured by Ladybug, taken in the environment of both scenarios.

The software system of Carina 2 is based on the Robot Operating System (ROS) framework, which offers the possibility to record many types of data in bagfiles. The data is composed by timestamped messages belonging to a particular topic. The data sequence recorded in a bagfile can be later reproduced, by playing the bagfile, and at any desired speed. In this case, the bagfiles contain GPS and IMU estimations, as well as the stream of stereo and set of Ladybug images. The algorithms are tested offline, by letting the programs access the data in a playing bagfile.

6.1.2 Analysis

The results analysis is carried out by observing the 3D estimated trajectory and the relative motion errors.

Figure 29: Example pictures of the environment.

The trajectory graphs show the trajectory on the XY-plane, as seen from above, and on the XZ-plane, to provide a vertical view, allowing to check how much a visual odometry estimate deviates vertically from the ground truth.

As for the relative motion error, it is computed as

$$\Delta Err = (\alpha_{gt,t} - \alpha_{gt,t-1}) - (\alpha_{vo,t} - \alpha_{vo,t-1}) \quad (46)$$

where α is each component of the motion (translation in x , y , z and rotation about x -, y -, z -axes). gt stands for ground truth and vo for visual odometry. The graphs show the relative error of each motion component in function of time. In order to help corresponding a part of the graph to the concerned trajectory section, the graphs are crossed by labeled vertical lines, at the same instants as the markers occur in the trajectories.

Finally, all graphs follow a color code, being each estimation associated to a particular color:

- Ground truth: black

- ❑ Stereo V.O.: blue
- ❑ Omnidirectional multi-camera V.O.: red
 - Monocular V.O. from camera 0: yellow
 - Monocular V.O. from camera 1: grey
 - Monocular V.O. from camera 2: blue
 - Monocular V.O. from camera 3: green
 - Monocular V.O. from camera 4: magenta

Stereo visual odometry and monocular odometry from any of the Ladybug individual cameras are never compared or shown together in a graph. Thus, the fact that stereo V.O. and monocular V.O. from camera 2 share the same color should introduce no confusion.

Ground truth position (x , y and z) is directly given by a GPS, which has RTK correction and a position precision of 0.4 m horizontally and 0.8 m vertically. Yaw angle is obtained by computing the angle of the GPS position in the XY-plane and roll and pitch angles are directly given by an IMU.

6.2 Equipment

In this section, information about the main hardware and software used along this project is provided.

6.2.1 Hardware

Hardware refers to the physical structure of computers and other physical devices. In this project, two types of camera are involved.

6.2.1.1 Bumblebee XB3

The Bumblebee XB3 system, developed and commercialized by FLIR, is a stereo camera rig, which means that there is a large overlap of the individual

Figure 30: Left: The Bumblebee XB3 camera. Right: Dimensions in mm. Pictures taken from (FLIR, 2012)

cameras' FoV. The system communicates with the computer through an IEEE-1394 FireWire interface.

Bumblebee XB3 is a set of three co-linear 1.3 MP cameras. As shown in Figure 30, the rig allows the use of two baselines, namely a shorter one (12 cm) and a wider one (24 cm). The images can have a maximum resolution of 1280 x 960 pixels and the maximum streaming rate is 16 fps. This information can be found in (FLIR, 2012). In this project, the 24 cm baseline is used.

Before each experiment, the cameras were calibrated using Zhang's method (ZHANG, 2000). To do so, the ROS package `camera_calibration` was used (BOWMAN, 2014), according to the tutorial described in (ANONYMOUS, 2012).

Figure 31 shows the relevant coordinate frames to be taken into account when using a Bumblebee camera.

Figure 31: Left: Individual and global coordinate frames. Right: Image coordinate frame.

Each individual camera has its own coordinate frame, where the z-axis points towards the scene and the y-axis points up. Rather than using all three cameras at the same time, usually users only use a camera pair at once, having the choice

between a short baseline (left and middle lenses) and a wide one (left and right lenses). For this reason, the coordinate frames are labeled as L , RS and RW for the left, middle and right cameras respectively, where S stands for short and W stands for wide baseline. The global coordinate frame, which is used to describe the position of the sensor, is here aligned with the left camera. Finally, the image coordinate system has its origin where the z-axis intercepts the image plane, which is about the center of the image. The u-axis describes the horizontal offset and the v-axis the vertical displacement.

6.2.1.2 Ladybug 2

Ladybug 2 is a camera cluster, also developed and commercialized by FLIR (Figure 32). It is composed by six 0.8 MP cameras: Five of them are evenly disposed on a ring, facing outwards, and the sixth camera is facing up. The first five cameras, which have small overlapping FoVs, provide an omnidirectional view of the surroundings. The addition of the sixth camera provides spherical vision, covering more than 75% of the whole sphere. It provides images with a maximum resolution of 1024 x 768 and allows streaming to disk at a maximum frame rate of 15 fps (However, in the performed experiments only 6 fps were achieved.). This and additional information can be found at (FLIR, 2013). Ladybug 2 communicates with the computer via a IEEE-1394 FireWire interface.

Figure 32: Left: The Ladybug 2 camera. Right: Its dimensions in mm. Figures taken from (FLIR, 2013).

Ladybug 2 has been previously calibrated by FLIR and the parameters are

provided. As it is built in a rigid fashion, the intrinsic and extrinsic parameters do not change and there is no need to calibrate it before each experiment.

There is a global 3D coordinate frame of the Ladybug, placed at the center of the object, according to Figure 33a. The origin is at the center of the CoPs of the five lateral cameras, the z-axis points up and the x-axis points towards camera 0. Each individual camera has its own 3D coordinate frame, where the origin is at the CoP, the z-axis is the optical axis, the x-axis points right and the y-axis points down (Figure 33b). Being LBT_x the transformation of camera x relatively to the Ladybug central frame, the transformation from camera a to camera b can be obtained through

$${}^a_bT = ({}^LBT_a)^{-1} \cdot {}^LBT_b. \quad (47)$$

There is also a 2D coordinate frame for each individual rectified image, where the origin is at the intersection of the optical axis and the image plane, the u-axis points right and the v-axis points down, as shown in Figure 33c.

Figure 33: Geometry of the Ladybug 2 camera. a) Global 3D coordinate frame. b) Individual 3D coordinate frames. c) 2D image coordinate frame. Figures taken from (FLIR, 2014).

When using Ladybug in outdoor environments, the distance to 3D landmarks is much larger than the distance between the individual cameras' CoPs. As such,

despite being a non-central system, Ladybug behaves like a central one for this type of environment, i.e., as if all CoPs were at the same point. Thus, this approximation may be conducted.

6.2.1.3 Computer

Experiments ran offline, in a Dell Vostro laptop. This dual-core computer has an i5 3317U CPU at 1.7 Hz and 6 GB RAM.

6.2.2 Middleware

Middleware can be thought off as a layer which stands between hardware and software. Its purpose is mainly to abstract hardware characteristics in order to facilitate working with software modules.

6.2.2.1 Robot Operating System (ROS)

ROS (Figure 34), developed by Willow Garage (QUIGLEY et al., 2009), is an open source development platform aimed at the creation of robotic applications. Users may freely download or create packages which provide different types of utilities, such as hardware abstraction, device drivers, software libraries, implementations of state of art algorithms and more. It also includes useful services that allow visualizing shapes, 3D points and trajectories. ROS code is usually organized in nodes and, generally, each node requires some input information, performs some computation with it and outputs the results. Data flow follows a publisher-subscriber policy: a node publishes messages on a topic and, for another node to access this information, it must subscribe to the same topic. The division of tasks in nodes and the consistent communication convention between them, even across different computers, facilitates the development of distributed systems, namely altering, switching and adding modules. This is very useful if, for instance, an heterogeneous group of people is collaborating in the development of a complex network of nodes. This information was extracted from (THOMAS, 2014).

Figure 34: ROS Logo

Supported languages include C++, Python and Lisp. ROS supports Unix platforms, such as Linux and Mac OS X, and integrates OpenCV, which will be addressed in the software section.

In this project, the version Indigo is used.

6.2.3 Software

Software is defined as the programs that run on computers and perform particular tasks. Libraries containing functions are also classified as software, as essential building blocks of programs.

6.2.3.1 OpenCV

The Open Source Computer Vision Library, or OpenCV (Figure 35), is a library suited for real-time computer vision applications. It was created by (BRADSKI, 2000) in 1999 and started getting support by Willow Garage in 2008. It is open source and widely used in robotic applications and other tasks that involve computer vision and image processing. It is written in C++, also offering C, Python and Java interfaces, and supports Windows, Linux, Mac OS, iOS and Android.

Figure 35: OpenCV Logo

6.2.3.2 Eigen

Eigen is a C++ library developed by (GUENNEBAUD; JACOB et al., 2010) (Figure 36). It contains several linear algebra utilities, including matrix and vector classes,

as well as several types of algebraical operations.

Figure 36: Eigen Logo

6.3 Results

This section presents the results obtained with the stereo and omnidirectional multi-camera visual odometry approaches. The estimated 3D trajectory is shown, as well as the relative motion errors. For the proposed omnidirectional multi-camera method, the voted odometry is also compared to the monocular visual odometry computed by the images of each Ladybug's camera.

6.3.1 Stereo visual odometry

6.3.1.1 Scenario 1

Figure 37 shows the trajectory of the path for scenario 1. The estimated trajectory drifts from the ground truth at a slow pace, specially after marker C, preventing it from closing the loop. However, despite this accumulating error, the ground truth is easily recognized in the estimated trajectory. The worst vertical error occurs around marker E and reaches approximately 20 mm.

Analyzing the error of each component of the relative motion (Figure 38), except for translation in z , the error varies around a constant value close to zero. The error for translation in y suffers major variations in the curves (between markers A and C and D and F). As for translation in x , the magnitude of the error reaches higher values than for y (up to 2 m), but no specific behavior is noticed for particular parts of the path. The error for translation in z is not constantly centered around zero, taking positive and negative values, depending on the trajectory section. Concerning relative rotation, the yaw angle suffers the lowest error, undergoing high variations during curves.

Figure 37: Trajectory estimated by stereo V.O.. Left: XY-plane. Right: XZ-plane.

Figure 38: Relative motion errors. Left: Translation. Right: Rotation.

6.3.1.2 Scenario 2

Similarly as for the first scenario, one can easily recognize the ground truth in the stereo visual odometry trajectory for scenario 2, as shown in Figure 39. The largest absolute vertical error is of approximately 70 m and occurs around marker H.

The relative motion errors for scenario 2 are similar to the ones obtained for scenario 1. Relative translation in y suffers large variations at curves or slightly curvy roads. Translation error in z tends to adopt positive values that vary along the path.

Figure 39: Trajectory estimated by stereo V.O.. Left: XY-plane. Right: XZ-plane.

Figure 40: Relative motion errors. Left: Translation. Right: Rotation.

6.3.2 Omnidirectional multi-camera visual odometry

6.3.2.1 Scenario 1

Omnidirectional multi-camera V.O.

As shown in Figure 41, left, the estimated trajectory provided by omnidirectional multi-camera visual odometry tends to drift away from the ground truth. Straight paths, like from the starting point to marker A and from marker F to the ending point, are estimated as wide curves. On the other hand, estimates of tight curves (A to C and D to F) stay similar to the real curves. Figure 41, right,

reveals a vertical drift in the negative z direction, reaching a maximum absolute error of approximately 25 m around marker E.

Figure 42 shows the error in each component of the relative motion. Unlike the stereo case, one can observe that the average translational error in x and y suffers large variations along the trajectory, while in z it tends to constantly stay around zero. The largest translational error is of -4 m in the x direction. As for rotation, the errors obtained with omnidirectional multi-camera visual odometry have a similar behavior as the ones obtained with stereo.

Figure 41: Trajectory estimated by omnidirectional multi-camera V.O.. Left: XY-plane. Right: XZ-plane.

Figure 42: Relative motion errors. Left: Translation. Right: Rotation.

Comparing with monocular V.O.

Monocular odometry from cameras 0, 2 and 3 provide a similar trajectory as the voted visual odometry, at least in the XY-plane (Figure 43). Considering the translation in z (Figure 44), monocular estimates tend to deviate from the true height by around 100 m for cameras 2, 3 and 4. The voted odometry presents the smallest vertical drift, reaching up to 30 m in the negative direction. Cameras 0 and 4 also tend to descend, while cameras 2 and 3 tend to ascend.

Figure 43: Trajectory estimated by each Ladybug camera's monocular V.O. and the voted (omnidirectional multi-camera) V.O. in the XY-plane.

Figure 44: Trajectory estimated by each Ladybug camera's monocular V.O. and the voted (omnidirectional multi-camera) V.O. in the XZ-plane.

Figure 45 shows the voted camera in each frame through colorful markers. Though cameras 0, 2 and 3 have been selected the most, all cameras contributed at some point. Furthermore, tendencies are observed in different parts of the path. For instance, camera 0 is frequently selected in the first curve, while cameras 2, 3 and 4 contribute the most for the second curve. The close-up in Figure 45 right shows that the trajectory suffers glitches, going backwards in some frames.

Figure 45: Voted camera at each frame. Left: Full trajectory. Right: Detail.

Figure 46 shows the relative translational errors of each monocular odometry, together with the one of the omnidirectional multi-camera approach. The x and y components are similar for all approaches, except for cameras 1 and 4. However,

the error in z is in general smaller for the omnidirectional multi-camera method, especially when comparing with cameras 0, 2 and 3. The relative rotational errors of all cameras are similar to the ones obtained with omnidirectional multi-camera odometry and are therefore not presented.

Figure 46: Relative translation errors of omnidirectional multi-camera V.O. and each monocular estimate.

6.3.2.2 Scenario 2

Omnidirectional multi-camera V.O.

In the second scenario, the trajectory estimated by the omnidirectional multi-camera approach is not as similar to the ground truth as in the first scenario, considering the XY-plane (Figure 47, left). In particular, the straight path between the starting point and C is estimated as a wide curve. Figure 47, right, shows that the voted trajectory drifts vertically from the true path in the negative z direction, reaching a magnitude of approximately 100 m around marker D.

As illustrated in Figure 48, relative translation in x and y directions present larger variation along the trajectory, reaching a magnitude of 5 m in the worst case, whereas translation error in z stays close to zero (less than 1 m magnitude). Similarly as in the first scenario, relative yaw errors assume lower amplitudes than roll and pitch, suffering larger variations around trajectory curves.

Figure 47: Trajectory estimated by omnidirectional multi-camera V.O.. Left: XY-plane. Right: XZ-plane.

Figure 48: Relative motion errors. Left: Translation. Right: Rotation.

Comparing with monocular V.O.

Figures 49 and 50 show the trajectories returned by monocular visual odometry of all cameras, as well as the voted odometry, in the XY- and XZ-plane, respectively. Adopting a qualitative point-of-view, none of the monocular estimates is similar to the ground truth, and, just by observing the XY-plane, the true trajectory is mostly recognizable in the voted estimate, considering the overall path. It also has the smallest vertical drift, reaching a magnitude of approximately 100 m in the negative z direction, while all monocular odometries deviate at least 200 m in the worst case. While cameras 0, 1 and 4 and the voted odometry tend to descend, cameras 2 and 3 tend to ascend.

Figure 49: Trajectory estimated by each Ladybug camera's monocular V.O. and the voted (omnidirectional multi-camera) V.O. in the XY-plane.

Figure 50: Trajectory estimated by each Ladybug camera's monocular V.O. and the voted (omnidirectional multi-camera) V.O. in the XZ-plane.

Figure 51 indicates that, once again, cameras 0, 2 and 3 are selected the most,

while camera 1 is rarely selected. As in the first scenario, several trajectory glitches are observed.

Figure 51: Voted camera at each frame. Left: Full trajectory. Right: Detail.

Figure 52 shows the relative translation errors of each monocular odometry and the voted odometry. Cameras 2 and 3 have a smaller error in the x direction as the voted odometry, while cameras 0, 1 and 2 outperform in the y direction. As for vertical relative translation, the omnidirectional multi-camera approach leads to the smallest error comparing to all cameras.

Figure 52: Relative translation errors of omnidirectional multi-camera V.O. and each monocular estimate.

6.3.3 Comparing stereo and omnidirectional multi-camera approaches

The Libviso2 library presents an accurate trajectory, as it allows easily recognizing the ground truth, despite some drift. The error graphs confirm the accuracy, as most of them stay around zero during the full experiment. Curves lead to large error variations in relative translation in y , though the error is still centered in zero. In order to obtain such results, the bagfile had to be played at 30% of the original speed.

As for the proposed omnidirectional multi-camera approach, the voted trajectories tend to be more similar to the ground truth than the monocular estimates, especially in the z direction. This behavior is confirmed by the error graphs. The proposed approach is also not compliant with real-time applications, as the bagfile had to be played at 30% of the original speed. The long execution time is a consequence of running the feature extraction and matching and the 8-Point algorithm on all the five cameras.

For either method, playing the bagfile at lower speeds lead to similar results, as faster speeds caused too many frames to be lost and lead to visibly worse trajectories. The remaining of this section compares both methods in terms of accuracy and real-time performance.

6.3.3.1 Accuracy

Figures 53 and 54 show the trajectories obtained by the stereo and omnidirectional multi-camera algorithms for the first and second scenario, respectively. The stereo approach outperforms the proposed one, as the obtained trajectory lies closer to the ground truth. A significant difference between both methods is that, while stereo visual odometry estimates straight paths correctly, the omnidirectional multi-camera method estimates them as wide curves. As for tight curves, their angle is similar to the real one for both methods. Stereo odometry tends to drift less in the z direction, although in scenario 2 both techniques lead to a similar maximum magnitude.

Figure 53: Scenario 1. Trajectories estimated by stereo and omnidirectional multi-camera V.O.. Left: XY-plane. Right: XZ-plane.

Figure 54: Scenario 2. Trajectories estimated by stereo and omnidirectional multi-camera V.O.. Left: XY-plane. Right: XZ-plane.

Analyzing the relative motion errors (Figures 55 and 56), the main differences lie in the x and y translation components. For both approaches, translation in x suffers the largest error. However, the error for stereo odometry has its average close to zero during the whole path, while the mean omnidirectional multi-camera error varies along the trajectory. As for the y direction, stereo leads to a very small error, with variations in the curves, while for the omnidirectional multi-camera approach it has a similar behavior as the x error. Stereo odometry also outperforms the omnidirectional multi-camera method in vertical translation, though the dif-

ference is much smaller, comparing to the other translation components. Relative rotation errors are similar for both approaches.

Figure 55: Scenario 1. Relative motion errors. Left: Translation. Right: Rotation.

Figure 56: Scenario 2. Relative motion errors. Left: Translation. Right: Rotation.

A reason for the significant gap between stereo and omnidirectional multi-camera odometry results could lie on the different frame rates of the two vision systems: while the Bumblebee camera transmits a 16 Hz image stream, the frame rate of the Ladybug system is of only 6 Hz. Consequently, the content of the captured images changes more between subsequent frames, possibly leading to less successful feature tracking results.

6.3.3.2 Real-time performance

Both methods lead to the presented results at 30% of the real speed, meaning that none of them is suitable for real-time applications with the current hardware. However, the algorithms may be speeded up by using a more powerful computer, with more cores, and running some parts in parallel. Downsampling the images before feature extraction and using more efficient visual odometry algorithms should also improve real-time performance.

Chapter 7

Conclusion

This chapter concludes this dissertation, summarizing the results of the experiments and indicating some ideas for future work.

7.1 Summary of results

In this dissertation two visual odometry approaches are compared, namely a well-established stereo visual odometry algorithm (GEIGER; ZIEGLER; STILLER, 2011) and a proposed omnidirectional multi-camera method. The dataset was collected by the stereo camera Bumblebee XB3 and the omnidirectional multi-camera system Ladybug 2, both developed by FLIR, fixed on the roof of a vehicle.

The proposed omnidirectional multi-camera approach consists in running a monocular visual odometry method (the 8-Point algorithm (HARTLEY, 1997)) on each camera and carrying out a voting scheme to select the most successful relative pose estimate. In the voting process each estimate is tested for all cameras: for each estimate, all feature matches in all image pairs are classified as inliers or outliers, depending on the distance of the observed 2D points from the epipolar lines dictated by the transformation to be tested. The estimate with the most inliers is the output of the algorithm. This process is repeated every time a new frame arrives, leading to a chain of the voted transformations. Using an omnidirectional multi-camera system, the part of the surroundings which is richest in features is always available, and can therefore be a candidate for the voted motion

estimation. Results show that the voted odometry provides better results than the individual monocular odometries, in particular concerning the vertical drift of the trajectories. Relative translation scale is not retrieved, as there is almost no overlapping between the cameras.

Comparing both stereo and omnidirectional multi-camera methods, the stereo approach proves to provide more accurate results, leading to trajectories closer to the ground truth. This is confirmed by the small relative motion errors, specially in translation. Furthermore, the overlapping between cameras allows retrieving the translation scale. Both algorithms achieve their best results running at 30% of the real speed.

7.2 Future work

There are various aspects in which the proposed omnidirectional multi-camera visual odometry method could improve.

Concerning real-time performance, in addition to a more powerful hardware, the visual odometry pipeline can be adapted by, for instance, adding an initial step where images are downsampled. Faster and more robust feature tracking and visual odometry techniques can also be used.

The low frame rate of the Ladybug camera, which is possibly a cause for the inaccurate motion estimation, can be overcome by driving the vehicle slower, such that the content of the captured images changes less across subsequent frames. The environment itself is also not a favorable place for a feature-based visual odometry algorithm, since it contains few close-by, stable elements, such as buildings. A large part of the images is occupied by the sky, which has no good features, and many visible elements lie far away, preventing features to be located precisely. It would be interesting to analyze if sections of the path with more nearby elements lead to more accurate relative pose estimations. Possibly the omnidirectional multi-camera visual odometry approach would present better results in more feature-rich outdoor environments, or even indoor places.

The inability to compute translation scale can be overcome by fusing visual odometry results with another sensor, such as an IMU or an encoder. Simplifying assumptions can also be taken into account. For instance, planar motion (3 DoF)

could be imposed for an initial estimate of the relative pose and, in a second step, it could be refined for 6 DoF motion. This way, the known distance between camera and floor plane could allow retrieving scale.

Finally, and drifting away from the main structure of the proposed algorithm, instead of handling the images of the vision system separately, they could be fused into one omnidirectional panoramic image. This would allow following features across more frames, availing of the main advantages of omnidirectional vision systems. The 8-Point algorithm (or any other monocular odometry method) would then be applied to this single image.

Bibliography

AGRAWAL, M.; KONOLIGE, K.; BLAS, M. R. Censure: Center surround extremas for realtime feature detection and matching. In: _____. **10th European Conference on Computer Vision**. [S.l.]: Springer Berlin Heidelberg, 2008. p. 102–115.

ANONYMOUS. **camera_calibration - Tutorial - Stereo calibration**. 2012. [Http://wiki.ros.org/camera_calibration/Tutorials/StereoCalibration](http://wiki.ros.org/camera_calibration/Tutorials/StereoCalibration). Accessed: 2015-07-05.

BAKER, S.; NAYAR, S. A theory of catadioptric image formation. In: **Sixth International Conference on Computer Vision, 1998**. [S.l.: s.n.], 1998. p. 35–42.

BASTANLAR, Y. Reduced egomotion estimation drift using omnidirectional views. **Electronic Letters on Computer Vision and Image Analysis**, v. 13, n. 3, 2014. ISSN 1577-5097.

BAY, H.; TUYTELAARS, T.; GOOL, L. V. Surf: Speeded up robust features. **Computer Vision and Image Understanding**, p. 346–359, 2008.

BOBICK, A. N-views (1) – homographies and projection. Lecture notes of Computer Vision – CS4495, School of Interactive Computing, Georgia Institute of Technology. 2014. Disponível em: <<http://www.cc.gatech.edu/~afb/classes/CS4495-Fall2014/slides/CS4495-TwoViews-1.pdf>>.

BOWMAN, P. M. J. **camera_calibration**. 2014. [Http://wiki.ros.org/camera_calibration](http://wiki.ros.org/camera_calibration). Accessed: 2015-07-05.

BRADSKI, G. The opencv library. **Dr. Dobb's Journal of Software Tools**, 2000.

CALONDER, M.; LEPETIT, V.; STRECHA, C.; FUA, P. Brief: Binary robust independent elementary features. In: _____. **Computer Vision – ECCV 2010: 11th European Conference on Computer Vision**. [S.l.]: Springer Berlin Heidelberg, 2010. p. 778–792.

- CLIPP, B.; KIM, J.; FRAHM, J.; POLLEFEYS, M.; HARTLEY, R. I. Robust 6dof motion estimation for non-overlapping, multi-camera systems. In: **9th IEEE Workshop on Applications of Computer Vision (WACV 2008), 7-9 January 2008, Copper Mountain, CO, USA**. [S.l.: s.n.], 2008. p. 1–8.
- COMPORT, A.; MALIS, E.; RIVES, P. Accurate quadrifocal tracking for robust 3d visual odometry. In: **IEEE International Conference on Robotics and Automation, 2007**. [S.l.: s.n.], 2007. p. 40–45.
- CORKE, P.; STRELOW, D.; SINGH, S. Omnidirectional visual odometry for a planetary rover. In: **Proceedings. 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004. (IROS 2004)**. [S.l.: s.n.], 2004. v. 4, p. 4007–4012 vol.4.
- CORKE, P. I. **Robotics, Vision & Control: Fundamental Algorithms in Matlab**. [S.l.]: Springer, 2011. ISBN 978-3-642-20143-1.
- CUMANI, A.; GUIDUCCI, A. Fast stereo-based visual odometry for rover navigation. **WSEAS Transactions on Circuits and Systems**, v. 7, n. 7, p. 648–657, 2008. Cited By 6.
- DAVISON, A.; REID, I.; MOLTON, N.; STASSE, O. Monoslam: Real-time single camera slam. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 29, n. 6, p. 1052–1067, 2007.
- FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. **Commun. ACM**, v. 24, n. 6, p. 381–395, jun. 1981. ISSN 0001-0782.
- FLIR. **Bumblebee - Stereo Vision Camera Systems**. [S.l.], 2012.
- _____. **Ladybug 2 - Compact size + 1394b + 30fps video**. [S.l.], 2013.
- _____. **Technical Application Note - Geometric Vision using Ladybug Cameras**. [S.l.], 2014.
- FORSTER, C.; PIZZOLI, M.; SCARAMUZZA, D. Svo: Fast semi-direct monocular visual odometry. In: **IEEE International Conference on Robotics and Automation (ICRA), 2014**. [S.l.: s.n.], 2014. p. 15–22.
- FRAUNDORFER, F.; SCARAMUZZA, D. Visual odometry - part ii: Matching, robustness, optimization, and applications. **Robotics Automation Magazine, IEEE**, v. 19, n. 2, p. 78–90, 2012. ISSN 1070-9932.

- GEIGER, A.; LENZ, P.; URTASUN, R. Are we ready for autonomous driving? the kitti vision benchmark suite. In: **Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2012.
- GEIGER, A.; ROSER, M.; URTASUN, R. Efficient large-scale stereo matching. In: **Asian Conference on Computer Vision (ACCV)**. [S.l.: s.n.], 2010.
- GEIGER, A.; ZIEGLER, J.; STILLER, C. Stereoscan: Dense 3d reconstruction in real-time. In: **Intelligent Vehicles Symposium (IV)**. [S.l.: s.n.], 2011.
- GEYER, C.; DANIILIDIS, K. A unifying theory for central panoramic systems and practical implications. In: VERNON, D. (Ed.). **Computer Vision — ECCV 2000**. [S.l.: Springer Berlin Heidelberg, 2000, (Lecture Notes in Computer Science, v. 1843). p. 445–461. ISBN 978-3-540-67686-7.
- GOECKE, R.; ASTHANA, A.; PETTERSSON, N.; PETERSSON, L. Visual vehicle egomotion estimation using the fourier-mellin transform. In: **Intelligent Vehicles Symposium, 2007 IEEE**. [S.l.: s.n.], 2007. p. 450–455.
- GROSSBERG, M. D.; NAYAR, S. K. A general imaging model and a method for finding its parameters. In: **In Proc. ICCV**. [S.l.: s.n.], 2001. p. 108–115.
- GUENNEBAUD, G.; JACOB, B. et al. **Eigen v3**. 2010.
[Http://eigen.tuxfamily.org](http://eigen.tuxfamily.org).
- HARRIS, C.; STEPHENS, M. A combined corner and edge detector. In: **Proceedings of the 4th Alvey Vision Conference**. [S.l.: s.n.], 1988. p. 147–151.
- HARTLEY, R. In defence of the 8-point algorithm. In: **IEEE Transactions on Pattern Analysis and Machine Intelligence**. [S.l.: s.n.], 1997. p. 580–593.
- HOANG, V.-D.; HERNÁNDEZ, D.; JO, K.-H. Combining edge and one-point ransac algorithm to estimate visual odometry. In: HUANG, D.-S.; BEVILACQUA, V.; FIGUEROA, J.; PREMARATNE, P. (Ed.). **Intelligent Computing Theories**. [S.l.: Springer Berlin Heidelberg, 2013, (Lecture Notes in Computer Science, v. 7995). p. 556–565. ISBN 978-3-642-39478-2.
- HOANG, V.-D.; LE, M.-H.; HERNANDEZ, D.; JO, K.-H. Localization estimation based on extended kalman filter using multiple sensors. In: **Industrial Electronics Society, IECON 2013 - 39th Annual Conference of the IEEE**. [S.l.: s.n.], 2013. p. 5498–5503. ISSN 1553-572X.

KAZIK, T.; KNEIP, L.; NIKOLIC, J.; POLLEFEYS, M.; SIEGWART, R. Real-time 6d stereo visual odometry with non-overlapping fields of view. In: **Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on**. [S.l.: s.n.], 2012. p. 1529–1536.

KIM, J.-H.; HARTLEY, R.; FRAHM, J.-M.; POLLEFEYS, M. Visual odometry for non-overlapping views using second-order cone programming. In: **Computer Vision – ACCV 2007**. [S.l.]: Springer Berlin Heidelberg, 2007. v. 4844, p. 353–362. ISBN 978-3-540-76389-5.

KITT, B.; GEIGER, A.; LATEGAHN, H. Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. In: **Intelligent Vehicles Symposium (IV), 2010 IEEE**. [S.l.: s.n.], 2010. p. 486–492.

KLEIN, G.; MURRAY, D. Parallel tracking and mapping for small ar workspaces. In: **6th IEEE and ACM International Symposium on Mixed and Augmented Reality, 2007. ISMAR 2007**. [S.l.: s.n.], 2007. p. 225–234.

KNEIP, L.; CHLI, M.; SIEGWART, R. Y. Robust real-time visual odometry with a single camera and an imu. In: **Proceedings of the British Machine Vision Conference 2011**. [S.l.]: British Machine Vision Association, 2011. ISBN 1-901725-43-X.

KNEIP, L.; LI, H. Efficient computation of relative pose for multi-camera systems. In: **IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014**. [S.l.: s.n.], 2014. p. 446–453.

KRESO, I. Overview of stereoscopic visual odometry methods. 2014.

LE, M.-H.; HOANG, V.-D.; VAVILIN, A.; JO, K.-H. Vehicle localization using omnidirectional camera with gps supporting in wide urban area. In: PARK, J.-I.; KIM, J. (Ed.). **Computer Vision - ACCV 2012 Workshops**. [S.l.]: Springer Berlin Heidelberg, 2012, (Lecture Notes in Computer Science, v. 7728). p. 230–241. ISBN 978-3-642-37409-8.

_____. One-point-plus for 5-dof localization of vehicle-mounted omnidirectional camera in long-range motion. **International Journal of Control, Automation and Systems**, Springer Berlin Heidelberg, v. 11, n. 5, p. 1018–1027, 2013. ISSN 1598-6446.

LEE, G. H.; FAUNDORFER, F.; POLLEFEYS, M. Motion estimation for self-driving cars with a generalized camera. In: **Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on**. [S.l.: s.n.], 2013. p. 2746–2753.

- LEUTENEGGER, S.; CHLI, M.; SIEGWART, R. Y. Brisk: Binary robust invariant scalable keypoints. In: **Proceedings of the 2011 International Conference on Computer Vision**. [S.l.: s.n.], 2011. p. 2548–2555.
- LI, H.; HARTLEY, R.; KIM, J. hak. A linear approach to motion estimation using generalized camera models. In: **Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on**. [S.l.: s.n.], 2008. p. 1–8.
- LOWE, D. G. Distinctive image features from scale-invariant keypoints. **International Journal of Computer Vision**, v. 60, p. 91–110, 2004.
- LUCAS, T. K. B. D. An iterative image registration technique with an application to stereo vision. In: **Proceedings of Imaging Understanding Workshop**. [S.l.: s.n.], 1981. p. 121–130.
- MILELLA, A.; SIEGWART, R. Stereo-based ego-motion estimation using pixel tracking and iterative closest point. In: **ICVS '06. IEEE International Conference on Computer Vision Systems, 2006**. [S.l.: s.n.], 2006. p. 21–21.
- MILFORD, M.; WYETH, G. Single camera vision-only slam on a suburban road network. In: **IEEE International Conference on Robotics and Automation, 2008. ICRA 2008**. [S.l.: s.n.], 2008. p. 3684–3689.
- MIRALDO, P.; ARAUJO, H. Generalized essential matrix: Properties of the singular value decomposition. **Image and Vision Computing**, v. 34, p. 45–50, 2015. ISSN 0262-8856.
- MORAVEC, H. **Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover**. [S.l.], 1980.
- MOU, W.; WANG, H.; SEET, G. Efficient visual odometry estimation using stereo camera. In: **11th IEEE International Conference on Control Automation (ICCA)**. [S.l.: s.n.], 2014. p. 1399–1403.
- MOURAGNON, E.; LHUILLIER, M.; DHOME, M.; DEKEYSER, F.; SAYD, P. Real time localization and 3d reconstruction. In: **IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006**. [S.l.: s.n.], 2006. v. 1, p. 363–370.
- NAYAR, S. Catadioptric omnidirectional camera. In: **Proceedings., 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997**. [S.l.: s.n.], 1997. p. 482–488. ISSN 1063-6919.

- NEWCOMBE, S. J. L. R. A.; DAVISON, A. J. Dtam: Dense tracking and mapping in real-time. In: **IEEE International Conference on Computer Vision**. [S.l.: s.n.], 2011. p. 2320–2327.
- NISTER, D.; NARODITSKY, O.; BERGEN, J. Visual odometry. In: **CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004**. [S.l.: s.n.], 2004. v. 1, p. I-652–I-659 Vol.1.
- OLSON, C.; MATTHIES, L.; SCHOPPERS, M.; MAIMONE, M. Rover navigation using stereo ego-motion. In: **Robotics and Autonomous Systems**. [S.l.: s.n.], 2003.
- OTSU, K.; KUBOTA, T. A two-point algorithm for stereo visual odometry in open outdoor environments. In: **IEEE International Conference on Robotics and Automation (ICRA), 2014**. [S.l.: s.n.], 2014. p. 1042–1047.
- PERSSON, M.; PICCINI, T.; FELSBURG, M.; MESTER, R. Robust stereo visual odometry from monocular techniques. In: **Intelligent Vehicles Symposium (IV), 2015 IEEE**. [S.l.: s.n.], 2015. p. 686–691.
- PLESS, R. Using many cameras as one. In: **In CVPR**. [S.l.: s.n.], 2003. p. 587–593.
- POJAR, D.; JEONG, P.; NEDEVSKI, S. Robust visual odometry using stereo reconstruction error model. In: **IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), 2012**. [S.l.: s.n.], 2012. p. 149–154.
- QUIGLEY, M.; CONLEY, K.; GERKEY, B. P.; FAUST, J.; FOOTE, T.; LEIBS, J.; WHEELER, R.; NG, A. Y. Ros: an open-source robot operating system. In: **ICRA Workshop on Open Source Software**. [S.l.: s.n.], 2009.
- ROSTEN, E.; DRUMMOND, T. Machine learning for high speed corner detection. In: **9th European Conference on Computer Vision**. [S.l.: s.n.], 2006. p. 430–443.
- RUBLEE, E.; RABAUD, V.; KONOLIGE, K.; BRADSKI, G. R. Orb: An efficient alternative to sift or surf. In: **ICCV 2011**. [S.l.: s.n.], 2011. p. 2564–2571.
- SCARAMUZZA, D. 1-point-ransac structure from motion for vehicle-mounted cameras by exploiting non-holonomic constraints. **International Journal of Computer Vision**, Springer US, v. 95, n. 1, p. 74–85, 2011. ISSN 0920-5691.

SCARAMUZZA, D.; FRAUNDORFER, F. Visual odometry - part i: The first 30 years and fundamentals. **Robotics Automation Magazine, IEEE**, v. 18, n. 4, p. 80–92, 2011. ISSN 1070-9932.

SCARAMUZZA, D.; FRAUNDORFER, F.; POLLEFEYS, M. Closing the loop in appearance-guided omnidirectional visual odometry by using vocabulary trees. **Robot. Auton. Syst.**, North-Holland Publishing Co., v. 58, n. 6, p. 820–827, 2010. ISSN 0921-8890.

SCARAMUZZA, D.; SIEGWART, R. Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles. **IEEE Transactions on Robotics**, v. 24, n. 5, p. 1015–1026, 2008. ISSN 1552-3098.

SILVA, H.; BERNARDINO, A.; SILVA, E. Probabilistic egomotion for stereo visual odometry. **Journal of Intelligent & Robotic Systems**, v. 77, n. 2, p. 265–280, 2014.

SINGH, A. **An OpenCV based implementation of Monocular Visual Odometry**. 2015. Github repository. Disponível em: <<https://github.com/avisingh599/mono-vo>>.

STEWENIUS, H.; ÅSTRÖM, K. Structure and motion problems for multiple rigidly moving cameras. In: _____. **Computer Vision - ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part III**. [S.l.]: Springer Berlin Heidelberg, 2004. p. 252–263.

STEWENIUS, H.; NISTÉR, D.; OSKARSSON, M.; ÅSTRÖM, K. Solutions to minimal generalized relative pose problems. In: **In Workshop on Omnidirectional Vision**. [S.l.: s.n.], 2005.

STRELOW, D.; MISHLER, J. S.; SINGH, S.; Herman, H. Extending shape from motion to non-central omnidirectional cameras. In: **Proceedings of IROS 2001**. [S.l.: s.n.], 2001.

SVOBODA, T.; PAJDLA, T. Epipolar geometry for central catadioptric cameras. **International Journal of Computer Vision**, Kluwer Academic Publishers, v. 49, n. 1, p. 23–37, 2002. ISSN 0920-5691.

TARDIF, J.-P.; PAVLIDIS, Y.; DANIILIDIS, K. Monocular visual odometry in urban environments using an omnidirectional camera. In: **IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008**. [S.l.: s.n.], 2008. p. 2531–2538.

THOMAS, D. **ROS Introduction**. 2014. [Http://wiki.ros.org/ROS/Introduction](http://wiki.ros.org/ROS/Introduction).

TRUCCO, E.; VERRI, A. **Introductory Techniques for 3-D Computer Vision**. [S.l.]: Prentice Hall PTR, 1998. ISBN 0132611082.

VENTURA, J.; ARTH, C.; LEPETIT, V. An efficient minimal solution for multi-camera motion. **International Conference on Computer Vision (ICCV)**, IEEE, Santiago, Chile, 2015.

WIRTH, S. **viso2_ros**. 2015. Package summary. Disponível em: <http://wiki.ros.org/viso2_ros>.

XIANG, Z.; SUN, B.; DAI, X. The camera itself as a calibration pattern: A novel self-calibration method for non-central catadioptric cameras. **Sensors**, v. 12, n. 6, p. 7299, 2012. ISSN 1424-8220. Disponível em: <<http://www.mdpi.com/1424-8220/12/6/7299>>.

ZHANG, Z. A flexible new technique for camera calibration. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 22, n. 11, p. 1330–1334, 2000. ISSN 0162-8828.

Appendices

APPENDIX A

Reference frames

This section approaches coordinate frames and their usefulness when working with objects' poses. It also explains how a point in one coordinate frame can be transformed to another frame, introducing the concept of homogeneous coordinates. This section was based on (CORKE, 2011), which contains a complete and clear explanation about this topic.

A.1 Definition

A coordinate frame in 3D space is a set of three orthogonal coordinate axes, usually called x-, y- and z-axis, as in Figure 58a. It serves as a reference when defining the position of objects. A rigid object can be abstracted to a coordinate frame. In this way, the relative pose (i.e., position and orientation) between two objects can be thought of as the relative pose between their coordinate frames, ignoring their particular structures.

Appropriate locations for coordinate frames depend on the nature of the problem. For example, if it is desired to know the pose of a plane, which can travel all around the world, it might be useful to have a reference frame attached somewhere on the planet and another one on the plane, since this allows to describe the pose of the plane unambiguously. If somewhere on the planet exists a room and one wants to know the pose of an object inside that room, it is not interesting to know

the object's geographic coordinates, but its pose relative to the room. In this case, one can attach a coordinate frame to a corner of the room and another one to the object. If the pose of the object relative to the room is known, as well as the room's pose relative to the world, one can compute the pose of the object relative to the world (Figure 57).

Figure 57: World (W , blue), room (X , red) and object (Y , green) reference frames. Black: transformations between world and room, world and object and room and object reference frames.

In 3D space the relative pose is fully described by six values, namely three components of 3D translation and the rotation over the x- (roll), y- (pitch) and z-axis (yaw). It is therefore said that the pose has 6 Degrees of Freedom (DoF). In some applications where objects can only perform planar movements (for instance, a wheeled car moving on a planar ground), there is no need to specify the height, as well as the rotation around the x- and y-axis. Thus, 2D pose consists of translation in the x and y directions and rotation over the z-axis, having only 3 DoF.

A.2 Transformations between Coordinate Frames

The axes of a coordinate frame contain a subscript which identifies the frame they belong to. The relative pose between two coordinate frames is described by the transformation one would need to perform to go from one coordinate frame to the other. For instance, in Figure 58b frame B'' is rotated relatively to frame A by an angle α around the z-axis, β around the y-axis and γ around the x-axis, in

this order. The overlapped origins show that there is no translation between the frames, but only the rotation

$${}^A_B R = R_z(\alpha) \cdot R_y(\beta) \cdot R_x(\gamma) \quad (48)$$

with

$$\begin{aligned} R_z(\alpha) &= \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ R_y(\beta) &= \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \\ R_x(\gamma) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) \\ 0 & \sin(\gamma) & \cos(\gamma) \end{bmatrix} \end{aligned}$$

The multiplication of the individual rotation matrices is not commutative. In this work, the adopted order is ZYX. Each column of the rotation matrix describes the position of an axis in frame B relatively to the corresponding axis in frame A . Rotation matrices are orthonormal, which means that their columns are orthogonal and have unit lengths.

Figure 58c shows a coordinate frame C which has the same orientation as frame B'' , but is translated. The displacement is denoted by the vector

$${}^{B''}_C t = [T_x, T_y, T_z]^T \quad (49)$$

The pose of frame C relatively to frame A consists of a rotation

$${}^A_C R = {}^A_{B''} R \cdot {}^{B''}_C R = {}^A_{B''} R \cdot I = {}^A_{B''} R \quad (50)$$

and a translation

$${}^A_C t = {}^A_{B''} R \cdot {}^{B''}_C t \quad (51)$$

Suppose the coordinates of a point ${}^C P$ in coordinate frame C are known and it is desired to know the same point described in the coordinate frame A , ${}^A P$. Then,

$${}^A P = {}^A_C R \cdot {}^C P + {}^A_C t \quad (52)$$

Figure 58: a) A coordinate frame A . b) Coordinate frame B is A rotated about the z -axis. c) Coordinate frame B' is B rotated about the y -axis. d) Coordinate frame B'' is B' rotated about the x -axis. e) A coordinate frame C is B'' translated.

It is said that point P was transformed from frame C to frame A .

A.3 Homogeneous Coordinates

Equation (52) can be written in a more compact form using homogeneous coordinates. This is achieved by adding a 1 to the vector, increasing its dimension. This means that a vector in $\mathbb{R}^2$ becomes a vector in $\mathbb{R}^3$, a vector in $\mathbb{R}^3$ becomes a vector in $\mathbb{R}^4$, and so on. The transformation of a vector can now be obtained by

left-multiplying the transformation matrix ${}^A_C T$:

$${}^A \tilde{P} = {}^A_C T \cdot {}^C \tilde{P} \quad (53)$$

or, extending the equation:

$$\begin{bmatrix} {}^A x \\ {}^A y \\ {}^A z \\ 1 \end{bmatrix} = \begin{bmatrix} {}^A_C R & {}^A_C t \\ 0^{1 \times 3} & 1 \end{bmatrix} \cdot \begin{bmatrix} {}^C x \\ {}^C y \\ {}^C z \\ 1 \end{bmatrix}$$

The tilde over the point implies that it is written in homogeneous coordinates.