Real Time Extraction of High Level Structures
Using a Semi-Calibrated Stereo System

by

Roberto de Lima Hernández

Thesis Submitted to the Program in Computer Science Department in partial fulfillments requirements for the degree of

Master at Computer Science

at the

Instituto Nacional de Astrofísica, Óptica y Electrónica
August, 2016
Tonantzintla, Puebla

Advisors:

Dr. José Martínez Carranza - Dr. Alicia Morales Reyes
Computer Science Department
INAOE

©INAOE 2016
The author hereby grants to INAOE permission to reproduce an to distribute copies of this thesis document in whole part
Agradecimientos

Al pueblo de México por el pago de sus impuestos a través de la beca otorgada por CONACyT (627047).

Al Instituto Nacional de Astrofísica Óptica y Electrónica por brindarme una excelente formación profesional.

A mis asesores, Dra. Alicia Morales Reyes y Dr. José Martínez Carranza, por su apoyo tanto en aspectos técnicos y académicos de la investigación como en aspectos personales, el cual fue pilar para la realización de ésta tesis.

A los miembros del comité revisor: Dr. Leopoldo Altamirano Robles, Dr. René Armando Cumplido Parra y Dra. Claudia Feregrino Uribe, por su crítica puntual durante el desarrollo de este trabajo.

A mi familia que me apoyó en todo momento y siempre confió en mí.
AGRADECIMIENTOS
Abstract

This research presents a novel methodology that combines stereo vision and parallel processing, based on GPU and the use of binary descriptors, for fast High-Level Structures extraction. Typical stereo algorithms require an image rectification stage that has to run on a frame-to-frame basis, increasing the computational burden and with the possibility of compromising high frame rate operation. Hence, it is proposed to use a semi-calibrated stereo approach, meaning that only calibration of extrinsic parameters of the stereo rig is carried out, thus avoiding a rectification process of the frames captured by the stereo camera. For the latter, the proposed approach relies on feature matching of salient points detected on the stereo images, from which image correspondences are obtained. These correspondences are triangulated to generate a point cloud that is passed to a plane fitting module. As feature matching is a cumbersome task, this study presents a novel GPU architecture to accelerate such process, thus achieving a real-time performance of up to 50 fps for the whole process. To demonstrate our approach, we also present an augmented reality application that exploits the planes extracted with our approach.
## Contents

Agradecimientos iii  

Abstract v  

1 Introduction 1  

1.1 Publications ................................................................. 2  
1.2 Motivation ................................................................. 2  
1.3 Goals ................................................................. 3  
1.4 Development and Results .................................................. 4  
1.5 Thesis Organization .................................................. 5  

2 Related Work 7  

2.1 Stereo Algorithms ................................................................. 7  
2.1.1 Depth Recovery from a Calibrated Stereo Rig ...................... 8  
2.1.2 Depth Recovery from an Un-Calibrated Stereo Rig ............... 11  
2.1.3 Other Stereo Methods .................................................. 12  
2.1.4 Real-Time Stereo Implementations .................................. 13  
2.2 Methods for High Level Structures Extraction ......................... 14  
2.2.1 RGB-D Based Methods .................................................. 14
CONTENTS

2.2.2 Binocular Stereo Based Methods ........................................ 16
2.3 Related Work Taxonomy ....................................................... 17
2.4 Summary .............................................................................. 19

3 Theoretical Framework .......................................................... 21
3.1 Feature Detection .................................................................... 21
3.2 Depth Recovery ...................................................................... 25
  3.2.1 Image Geometry ................................................................. 25
  3.2.2 Stereo Geometry ................................................................. 27
  3.2.3 Depth from Binocular Stereo .............................................. 29
3.3 High Performance Computing .................................................. 35
  3.3.1 GPU’s ............................................................................. 36
  3.3.2 FPGA’s ........................................................................... 37
3.4 Parametrized Algorithms ....................................................... 38
3.5 Summary .............................................................................. 39

4 Proposed Approach ................................................................. 41
4.1 Binary Descriptors Extraction ................................................ 41
4.2 Feature Matching .................................................................... 44
4.3 Point Cloud Recovery ............................................................ 49
4.4 Plane Extraction ..................................................................... 52
4.5 Summary .............................................................................. 57

5 Experiments and Results .......................................................... 59
5.1 Evaluation of the Semi-Calibrated Stereo Rig .......................... 59
5.2 Depth Evaluation .................................................................... 61
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3 Recovered Area</td>
<td>64</td>
</tr>
<tr>
<td>5.4 Processing Time</td>
<td>67</td>
</tr>
<tr>
<td>5.5 Augmented Reality Example</td>
<td>67</td>
</tr>
<tr>
<td>5.6 Analysis Results</td>
<td>68</td>
</tr>
<tr>
<td>6 Conclusions and Future Work</td>
<td>71</td>
</tr>
<tr>
<td>6.1 Limitations</td>
<td>73</td>
</tr>
<tr>
<td>6.2 Future Work</td>
<td>73</td>
</tr>
<tr>
<td>A FPGA-based architecture for ORB descriptors construction.</td>
<td>75</td>
</tr>
</tbody>
</table>
CONTENTS
List of Figures

1.1 Description of the proposed method broken down in 3 stages and an Augmented Reality example, which takes advantage of the plane extraction. The latter stages are showed from a 3-D top-view perspective with respect to the stereo camera (in red). ................................................................. 4

2.1 Typical process for depth estimation using a stereo. ........................................ 9
2.2 Example of block matching process. .............................................................. 10
2.3 Example of Feature Matching. Blue circles are extracted features and green lines are the matched features. ................................................................. 11
2.4 Geometry of projected point (top-view). y and y’ are out of paper due to view perspective. ................................................................. 13
2.5 Taxonomy of the approaches for planes extraction ........................................ 18

3.1 Feature Description process for the binary descriptors whose method relies on pixel intensities method. ................................................................. 24
3.2 Pinhole camera model. ................................................................................ 25
3.3 Perspective Projection. ................................................................................ 26
3.4 Front Perspective Projection. ........................................................................ 26
3.5 Stereo with parallel axes. ............................................................................. 28
3.6 Stereo with Convergent Axes ...................................................................... 28
3.7 Typical process of a stereo vision system. .................................................. 30
<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.8</td>
<td>Stereo Epipolar Geometry.</td>
<td>31</td>
</tr>
<tr>
<td>3.9</td>
<td>Example of image rectification.</td>
<td>33</td>
</tr>
<tr>
<td>3.10</td>
<td>Illustration of triangulation problem.</td>
<td>35</td>
</tr>
<tr>
<td>3.11</td>
<td>CUDA memory model.</td>
<td>36</td>
</tr>
<tr>
<td>3.12</td>
<td>CUDA programming scheme.</td>
<td>37</td>
</tr>
<tr>
<td>3.13</td>
<td>Internal structure of a CUDA block.</td>
<td>37</td>
</tr>
<tr>
<td>3.14</td>
<td>RANSAC overview.</td>
<td>39</td>
</tr>
<tr>
<td>4.1</td>
<td>Work-flow for planes extraction.</td>
<td>41</td>
</tr>
<tr>
<td>4.2</td>
<td>Several explored binary tests distributions (images taken from [Calonder et al., 2010]).</td>
<td>43</td>
</tr>
<tr>
<td>4.3</td>
<td>Thread blocks configuration: Kernel 1 grid: 32 blocks: (32,32) threads; kernel 2 grid: 1024 blocks: (32,32) threads; kernel 3 grid: 1 block: (32,32) threads.</td>
<td>45</td>
</tr>
<tr>
<td>4.4</td>
<td>Hardware design for computing Hamming distance (kernel 1)</td>
<td>46</td>
</tr>
<tr>
<td>4.5</td>
<td>Computing the minimum Hamming distance (kernel 2)</td>
<td>47</td>
</tr>
<tr>
<td>4.6</td>
<td>Discriminating those Hamming distance greater than a threshold (kernel 2)</td>
<td>48</td>
</tr>
<tr>
<td>4.7</td>
<td>Typical process for depth estimation. 1. Distortion removal is performed for all keypoints. 2. Keypoints are matched considering a coplanar stereo rig. 3. Depth estimation is computed by disparity</td>
<td>50</td>
</tr>
<tr>
<td>4.8</td>
<td>Proposed process process for depth estimation. 1. Keypoints are matched considering un-rectified images. 2. Distortion removal is carried out only for matched keypoints. 3. Depth is computed by triangulation using line intersection method</td>
<td>51</td>
</tr>
<tr>
<td>4.9</td>
<td>Extraction of the predominant plane using RANSAC.</td>
<td>55</td>
</tr>
<tr>
<td>4.10</td>
<td>Extraction of the secondary plane using RANSAC.</td>
<td>56</td>
</tr>
<tr>
<td>4.11</td>
<td>Final result of planes extraction using RANSAC.</td>
<td>57</td>
</tr>
<tr>
<td>5.1</td>
<td>Experimental setup for the semi-calibrated stereo assessment. Blue points over the calibration board represent the vicon markers.</td>
<td>60</td>
</tr>
</tbody>
</table>
5.2 Experimental setup of depth evaluation. Blue points over the plane represent the vicon markers. 61
5.3 Depth estimation results using three triangulation methods: DLT, Iterative DLT and line intersection. 26 poses are considering for distances from 55 cm to 600 cm. 62
5.4 Depth estimation results based on line intersection for 19 poses, starting at 55 cm up to 300 cm. 62
5.5 Comparison between commercial sensors and the proposed system, in terms of depth accuracy. 63
5.6 Experimental setup for depth accuracy experiment in outdoors environment. 64
5.7 Depth estimation results in outdoors, based on line intersection. Mean error of 12.6 cm. 64
5.8 Plane extraction for different indoor scenes. Top: extracted planes. Bottom: Top-view 3-D representation. 65
5.9 Example where the proposed approach struggles to extract planes in an outdoor environment. 66
5.10 Plane extraction for different outdoor scenes. Top: extracted planes. Bottom: Top-view 3-D representation. 66
5.11 Snapshots of the augmented reality example performed. The fundamental plane extracted, the video projection over the plane and its 3-D perspective with respect to the stereo system are shown respectively. 68
A.1 Datasets used for evaluation. 76
A.2 (Part 2. Transformations: JPEG compression, illumination changes and rotation.) 77
A.3 (Part 1. Transformations: view-point changes and blur) Recall vs 1 precision graphs for nearest neighbor strategy. Curves are obtained by tuning a threshold of distance between descriptors. In parenthesis, next to the name of each of the methods, the number of found correspondences is shown. 78
A.4 FPGA Architecture for the construction of ORB using our approach. 79
LIST OF FIGURES
List of Tables

2.1 Stereo Matching Approaches .................................................. 10
2.2 Real time implementations of stereo algorithms ....................... 14
2.3 State of the Art overview ....................................................... 18

3.1 Brief overview of the most significant local descriptors reported in the literature. . . 24

5.1 Mean distance(over 320 poses) of the calibration board with respect to stereo Camera, using two optimization methods to obtain extrinsic parameters. Information obtained by Vicon is taken as ground truth. ........................................ 60
5.2 Mean error per triangulation method, based on results shown in Figure 5.4. ........ 62
5.3 Average percent of area recovered in terms of the key points associated to a plane, using Line Intersection triangulation method. ........................................ 65
5.4 Average percent of area recovered in terms of the key points associated to a plane, using Line Intersection triangulation method in outdoor environments. ........ 66
5.5 Processing time in milliseconds of each framework stage, implemented in two different GPU’s. Given the environment set up (Fig.5.8) 1800 features were detected on average. .............................................................. 67
LIST OF TABLES
Chapter 1

Introduction

The ability to perceive the three-dimensional structure of the world around us is an essential characteristic that enables us, as humans, to carry out necessary activities. This is easily performed thanks to the sense of sight. In this regard, since one of the aims of Artificial Intelligence (AI) consist in designing systems able to solve everyday problems by itself, computer vision becomes a fundamental branch of AI, focused on simulating the human visual system to develop activities such as navigation, face recognition, 3-D reconstruction, etc.

Over the last years, several algorithms have been proposed in the literature for depth recovery. Nowadays, this represents an active field of research since many current applications such as Augmented Reality (AR), navigation, localization are demanding more accuracy while fulfilling real-time constraints. In this sense, research areas such as Stereo Vision and High Performance Computing, have won their place as an important component of depth recovery.

Inspired by the way that humans observe the world in three dimensions, Stereo Vision is an area of study of computer vision, which aims at recovering 3-D information of a scene, by using two or more perspectives. Even though, this field have been broadly studied over the last decades, it is still an open research due to the application context is requiring robust stereo systems in terms of accuracy, processing time and operation at different environments.

On the other hand, High-Performance Computing (HPC) focused on reducing the computational burden of a problem by utilizing parallel processing. Typically, HPC relies on Graphics Processing Units (GPU’s), and Field Programmable Gate Arrays (FPGA’s) for the design of parallel schemes. Since most vision tasks are computationally intensive, involving complex processing,
HPC is an attractive approach to satisfy the real-time requirements of vision applications.

A mandatory task in computer vision systems is the 3-D recovery of High-Level Structures HLS. These entities refer to those built by more than three points, such as planes, cubes, cylinders, tetrahedrons, etc. This task has gained great relevance due to HLS are commonly found in man-made environments, thus making it suitable to be the basic core of applications such as reconstruction, mobile navigation, and AR.

Several challenges are involved in the extraction of HLS, for instance: real-time performance, extraction accuracy, operation at indoor and outdoor environments, and depth accuracy; elements that make Stereo Vision as much as HPC reliable approaches for HLS extraction. However, stereo techniques are computationally expensive. Moreover, its performance strongly depends on stereo calibration parameters, thus avoiding to fulfill real-time requirements.

1.1 Publications

This research produced the following list of publications:


1.2 Motivation

HLS extraction task may be broken down in two stages: 1) 3-D estimation of the world. 2) Extracting Planes given a set of visual characteristics in 3-D space. Considering the above, Stereo Vision is a crucial field of study in this research.
Stereo vision algorithms are classified in two approaches: Active and Passive Stereo. The former briefly refers to the replacement of one of the cameras by an external source of light, for instance: structured light. This technique has gained relevance during the last years, due to it yields good performance concerning depth accuracy. Hence, RGB-D sensors such as Kinect, Xtion, etc. take advantage of this technique to acquire 3-D information of a scene. However, as RGB-D working mode depends on an infrared light; its performance is limited for indoor environments. On the other hand, Passive Stereo requires two or more cameras for 3-D acquiring. Thus, depth information is inferred from cameras geometry, in such a way that depth accuracy highly relies on stereo calibration.

Stereo Camera calibration is the process of determining the intrinsic and extrinsic camera parameters, both are necessary to transform object coordinates to a camera-centred coordinate frame. In addition, these parameters are fundamental to infer 3-D information since they provide geometric information of the stereo system. Nevertheless, this process is computationally intensive due to several complex operations are included, such as computation of epipoles and projective transformations, distortion removal, and image rectification; thus preventing real-time performance.

On one hand, by using Active Stereo approach, in particular, RGB-D sensors, the application performance is restricted to operate in indoor environments. On the contrary, the classic techniques of Passive Stereo involve a trade-off between operation at indoor/outdoor environments and the increase of processing time. Motivated by the latter, to generate a fast and accurate point cloud whereby the HLS extraction is carried out, in this thesis is proposed to deal with a semi-calibrated stereo rig, this means, solely orientation and translation stereo parameters are known, in such a way that rectified images are avoided, thus achieving real-time performance.

### 1.3 Goals

**Aim**

The primary goal of this research is to develop a method for HLS extraction of urban scenes, using a semi-calibrated stereo rig and hardware architectures to achieve real-time performance.

**Specific Objectives**

The above through the followed partial objectives:
• To recover a three-dimensional point cloud.

• To extract HLS, in particular planes, from the point cloud.

• To accelerate the algorithmic stages of the proposed method to achieve a total processing time of up to 30 fps.

1.4 Development and Results

The proposed methodology is broken down in the following stages:

1. Visual descriptors extraction

2. Feature Matching via a parallel scheme

3. Point Cloud estimation

4. Plane Extraction

An overview of the proposed methodology is illustrated in Figure 1.1.

Figure 1.1: Description of the proposed method broken down in 3 stages and an Augmented Reality example, which takes advantage of the plane extraction. The latter stages are showed from a 3-D top-view perspective with respect to the stereo camera (in red).

The first three stages aim at generating a 3-D perspective of the world, therefore, in the first step, features of the pair of stereo images are extracted and described employing binary
1.5. Thesis Organization

descriptors, which have been proved to be fast to compute and robust against affine transformations. Accordingly, these features are matched following a parallel scheme implemented in a GPU. Then, a triangulation process is carried out to infer the depth of each matched point. Finally, as we mentioned before, the problem is reduced to the extraction of HLS in the form of planes from a point cloud. Therefore, the Sample Random Consensus (RANSAC) method is used for plane extraction.

The performance of the proposed method is assessed concerning depth information accuracy, quality of planes extracted and processing time. Hence, according to the reported results, this method can be used to estimate planes up to a maximum distance of 300 cm, achieving a total processing time up to 50fps. These qualities make the proposed method well-suited for applications such as navigation, localization, and AR. For the sake of illustration, an example of AR application is presented.

1.5 Thesis Organization

The thesis is organized as follows. First, an overview of the different methodologies for addressing the problem of High-Level Structures extraction is given in Chapter 2. After providing a general panorama of the research developed in the literature, Chapter 3 describes the theoretical concepts that were used as building blocks in this thesis work. The proposed approach for HLS extraction and the experiments performed to validate the method are presented in Chapter 4 and 5. Finally, conclusions and future work are provided in Chapter 6.
CHAPTER 1. INTRODUCTION
Chapter 2

Related Work

In the past chapter, the problem of High Level Structures extraction was introduced. As seen in introduction, this task requires a fundamental previous step which consists in generating a 3-D perspective of the world. The performance of many computer vision systems highly depends on this stage, in such a way that several approaches to generate a depth map have been proposed in the literature. Thus making the extraction of High Level Structures a challenging task, as it involves several computational components such as: estimating an accurate 3-D world perspective, extracting 3-D models given a set of points in 3D-space, accurate estimation of camera calibration parameters, real-time operation, etc. Taking the above into consideration, all the elements developed in the literature around High Level Structures extraction are described in this chapter.

For the sake of clarity, this chapter is divided into three parts. Firstly, a revision of the main approaches to generating a 3-D perspective of the world using stereo vision are given. Secondly, an overview of those algorithms focused on extracting High Level Structures is provided. Finally, a taxonomy of the related work is presented in order to establish a point of comparison with the proposed work. Since one of the main requirements demanded by the application context refers to real-time operation, a revision of those hardware-based works oriented to accelerate critical stages is given in each subsection.

2.1 Stereo Algorithms

Stereo is a classical problem in computer vision with wide-ranging applications, including High Level Structures extraction. Stereo problem refers to the estimation of 3-dimensional structure
CHAPTER 2. RELATED WORK

of a scene from two or more images taken from distinct viewpoints. The fundamental basis for stereo lies in the fact that a single three-dimensional world point is projected in every captured image. Thus, the 3-D estimation is computed from the geometry of the different viewpoints. Stereo involves three primary problems: calibration, correspondence and reconstruction.

Calibration is the process of finding intrinsic and extrinsic camera parameters. The former refers to focal length, optical centers, and lens distortions. The latter refers to the relative position and orientation of each camera and the baseline. These parameters are necessary to relate image information, expressed in pixels, to an external world coordinate system. Therefore, the accuracy in terms of depth information strongly relies on calibration process. Moreover, this is a preliminary task to rectification stage, a process that aims at mapping a stereo rig with convergent axes into a stereo rig with parallel axes.

The correspondence problem is one of the most heavily investigated topics in computer vision and refers to the problem of determining a set of points in the left image, which can be identified as the same points in the right image considering a binocular stereo rig. This is a fundamental stage for estimating the geometry of an object point projected in stereo images.

Reconstruction is the problem of estimating the 3-D dimensional structure of an object. This process is typically known as triangulation problem and has been widely explored in the literature since several factors must be considered to obtain an accurate 3-D model of the world, for instance certainty of correspondence points, calibration procedures, outliers removal, etc.

A revision of the most common stereo algorithms is broken down in those that rely on calibrated images and those that do not require rectified images. In addition, a very brief overview of stereo alternatives and real-time stereo implementations is given at the end of the chapter.

2.1.1 Depth Recovery from a Calibrated Stereo Rig

The classical process for computing depth from a calibrated stereo rig can be described as:

1. Calibration of each stereo camera (left and right).
2. Calibration of the stereo camera.
3. Images Rectification.
5. Triangulation from disparity map.
2.1. STEREO ALGORITHMS

Figure 2.1: Typical process for depth estimation using a stereo.

The three first steps are related to calibration process. Monocular cameras as much as stereo cameras are calibrated by using different tools such as Matlab [Guide, 1998] or OpenCV [Bradski and Kaehler, 2008]; this is a fundamental process to estimate an accurate 3-D perspective of the world, and yields as result the necessary data to carry out the subsequent procedures. Rectification process aligns the stereo images such that one point in the left image can be found in the right image by searching on the same epipolar line, thus making the correspondence problem easier while reducing computation time.

Stereo algorithms aim at generating a disparity map from which 3-dimensional information is computed. Disparity is the difference in position between correspondence points in the two images, \( \text{disparity} = x_{\text{left}} - x_{\text{right}} \). Therefore a disparity map encodes the disparity of all matched points and it is inversely proportional to depth. Hence, the core of stereo algorithms depends on correspondence problem.

Stereo Matching Approaches are classified in local and global. Since the former are faster to compute, this section focuses on giving a brief description of local correspondence methods which are outlined in Table 2.1.

Block Matching methods seek to estimate disparity at a point in one image, by comparing a small region around that point, better known as "matching window", with a series of matching windows extracted from another image. The search is limited by epipolar line. Three classes of matching metrics are commonly found in the literature: Normalized Cross-Correlation (NCC), Sum of Squared Differences (SSD), and Normalized SSD. Figure 2.2 illustrates the mentioned process.

Gradient-Based methods or also known as optical flow, aim at determining small local disparities between two regions of images. This method is similar to block matching methods in the sense that a point in the left image is searching on the right image by moving "windows" over
CHAPTER 2. RELATED WORK

<table>
<thead>
<tr>
<th>Approach</th>
<th>References</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block Matching</td>
<td>[Aschwanden and Guggenbuhl, 1992]</td>
<td>Search for maximum match score over small regions.</td>
</tr>
<tr>
<td></td>
<td>[Veksler, 2003]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[Tippetts et al., 2016]</td>
<td></td>
</tr>
<tr>
<td>Gradient-based methods</td>
<td>[Birchfield and Tomasi, 1999]</td>
<td>Minimize a functional, typically the sum of squared differences, over small regions</td>
</tr>
<tr>
<td></td>
<td>[Mesmakhosroshahi et al., 2014]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[Jung et al., 2013]</td>
<td></td>
</tr>
<tr>
<td>Feature Matching</td>
<td>[Schmid and Zisserman, 1998]</td>
<td>Find similar matches between stereo images</td>
</tr>
<tr>
<td></td>
<td>[Baumberg, 2000]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[Tan et al., 2015]</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1: Stereo Matching Approaches

Figure 2.2: Example of block matching process.

epipolar line. The key idea is based on the fact that image brightness of a point in the scene is constant between the two views.

Feature-based methods as their name indicate, aim at extracting points of interest in both images such as edges, lines, corners, etc. and searching similar features between salient points. In this sense, the searching area may cover the entire image, or otherwise, it may be restricted by epipolar line. Visual descriptors such as SIFT [Heymann et al., 2007], SURF [Bay et al., 2006], ORB [Rublee et al., 2011], etc. play an important role in this method due to they give a short representation of the features, and descriptors comparison is carried out by euclidean distance or hamming distance, according to the descriptor representation. Figure 2.3 depicts an overview of these methods.
2.1. STEREO ALGORITHMS

2.1.2 Depth Recovery from an Un-Calibrated Stereo Rig

Some problems, in specific those oriented to solve real-world applications such as: real-time navigation, augmented reality, localization, etc. Require a computer system that operates under raw images, which means that stereo images are uncalibrated. Therefore, this section provides a summarized revision of the stereo algorithms where calibration parameters are unavailable.

As it has been seen, depth recovery of the world is easily performed from disparity map. However, considering an un-calibrated stereo rig, the problem of determining a 3-dimensional perspective has to be addressed by a different approach because depth is inferred from a stereo with convergent axes. The following methods take as an assumption that points of interest have been previously matched either following feature matching strategies or by some correlation matching process.

Feature matching was briefly introduced in the last subsection, however, in this case the searching area is not restricted by epipolar line, on the contrary, it covers all the features in the image. From the above, a logical solution consists in following a brute force technique, meaning that, one descriptor in the left image is compared against all descriptors in the right image, repeating this process for every left descriptor.

Correlation matching assumes [Deriche et al., 1994] that points of interest have been extracted in the pair of stereo images. Then, given a high curvature point in the left image, a correlation window around this point is selected. After that, a rectangular search area is defined around a random point in the right image, and a correlation operation is performed between the left point and the right points within the area defined. The correlation process is carried out iteratively, similar to feature matching.

A first attempt to compute depth information from an uncalibrated stereo rig and given a
set of matched points is proposed in [Hartley et al., 1992]. This approach depends on calibration parameters of each stereo camera. Therefore, there exists a 3 x 3 matrix $Q$ known as the essential matrix [Faugeras, 1993], such that if a sufficient number of matched points are known, the matrix $Q$ can be computed by the solution of a set of linear equations, then it is possible to determine from $Q$ the relative location of the cameras and consequently the 3-D location of the matched points.

Similar to the above, in [Faugeras, 1992] depth information is computed on the basis of perspective projection. Therefore, three choices of coordinates are select: i) Choose 3D unknown points as the standard projective basis. ii) Choose four salient points from the left image. iii) From the right image, select the matched points of the left image. Then, depth data is inferred by computing perspective matrices of the two stereo views.

More recently, in contrast to mentioned methods, in [Loghman et al., 2014] authors propose a stereo matching approach from an uncalibrated stereo rig. The key idea of this method lies in computing the epipolar line equation by using RANSAC [Derpanis, 2010] and the 8-point algorithm [Hartley, 1997]. Accordingly, the search for point correspondence is performed along the epipolar line. Thus, depth estimation is carried out by calculating a disparity map.

Indeed the described methods make a computer vision system able to infer depth from an uncalibrated stereo rig. Nonetheless, they involve a significant computational cost. Moreover, the basic idea behind those methods may be seen as an on-line calibration. Even though, the techniques used are different the main idea is to establish a relationship between left and right camera, a task performed by the stereo calibration process.

### 2.1.3 Other Stereo Methods

Aside from passive stereo methods for inferring depth, several approaches have been proposed in the literature such as imaging radar [Hoover, 1988], laser-based methods [Iocchi and Pellegrini, 2007], and structured light [Zhang, 2012]. The latter approach has been leveraged by commercial sensors such as kinect and xtion to compute the depth of a scene. Since these sensors offer high frame rate operation and accurate depth information, they have been used in diverse applications, including the extraction of high level structures.

Structured Light refers to systems in which a scene is illuminated by a known pattern of light and observed by a camera separated by a baseline distance ($b$), such as is shown in Figure 2.4. Based on the projected point geometry, the equation 2.1 describes the relationship between the image coordinates ($x', y'$), the projection angle $\theta$ and the object coordinates ($x, y, z$). Note that
2.1. STEREO ALGORITHMS

Field of view is determined by the projection angle $\theta$.

\[ (x, y, z) = b \frac{f \cot \theta - x'}{x'y'f} \]  

(2.1)

Figure 2.4: Geometry of projected point (top-view). $y$ and $y'$ are out of paper due to view perspective.

2.1.4 Real-Time Stereo Implementations

As noted in sections 2.1.1 and 2.1.2, stereo methods require a sequence of intensive algorithmic stages, which notably increase the computational burden. Thus preventing to perform applications with real-time constraints. Motivated by the latter, several approaches for accelerating stereo techniques have been proposed in the literature. Those approaches exploit the rising of computational power dictated by Moore’s law. Thus, Graphic Processor Units (GPU’s) as much as Field Programmable Gate Arrays (FPGA’s), become an appealing hardware platform to reduce the computational burden.

GPU’s emerged as an alternative to typical processors, by providing an architecture with several smaller processors, better known as cores, thus offering a platform that enables to run a process in parallel, thereby accelerating several applications. In the same regard, FPGA’s allows to implement digital circuits specialized to perform specific algorithms in such a way that an FPGA allows to design a dedicated processor. These devices are commonly used to implement parallel designs focused on accelerating stereo algorithms. Table 2.2 depicts a description of recent real-time stereo implementations. It is important to point out that all the reported works require as input, a set of calibrated stereo images. Hence, considering a real-time application, their performance in terms of frame rate, would decrease notably.
Table 2.2: Real time implementations of stereo algorithms

<table>
<thead>
<tr>
<th>Real-time system</th>
<th>Image size</th>
<th>Frame rate</th>
<th>Method</th>
<th>Processor</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Kowalczuk et al., 2013]</td>
<td>320 x 240</td>
<td>62</td>
<td>Iterative Refinement</td>
<td>GPU- GeForce GTX 580</td>
</tr>
<tr>
<td>[Denker and Umlauf, 2011]</td>
<td>800 x 800</td>
<td>7.5</td>
<td>Cross Correlation</td>
<td>GPU - GeForce GTX 285</td>
</tr>
<tr>
<td>[Tippetts et al., 2014]</td>
<td>450 x 375</td>
<td>50</td>
<td>Shape matching</td>
<td>FPGA- e Xilinx Virtex 4 FX60</td>
</tr>
<tr>
<td>[Banz et al., 2010]</td>
<td>640 x 480</td>
<td>30</td>
<td>Semi global matching disparity</td>
<td>FPGA - Xilinx Virtex 5</td>
</tr>
</tbody>
</table>

2.2 Methods for High Level Structures Extraction

The above section was oriented to provide a very brief revision of the methods for generating a 3-dimensional perspective of the world from stereo vision. The above methods arguably, aim at generating a 3-D point cloud, which gives a description in real-world coordinates of the scene. This set of 3-dimensional points is the previous step for several applications such as: reconstruction, scene recognition, object tracking, augmented reality, navigation, etc. But the must important, it is a fundamental stage for High Level Structures extraction. Hence, this section provides a revision of the different methodologies to address the presented problem by classifying them according to the visual sensor used to capture the scenario, this is, RGB-D sensors and stereo cameras. In addition, a taxonomy is given in order to establish a point of comparison between the state of the art and the proposed work, thus easily locating the research contributions.

2.2.1 RGB-D Based Methods

RGB-D sensors are typically used for applications that require an accurate perspective of an object in 3D-space. Basically, RGB-D based visual systems make use of structured light techniques to provide a dense point cloud of the world. In such a way that the problem of High Level Structures extraction is reduced to estimate a model from a set of 3-dimensional points. In this regard, parametrized algorithms have proved to be a reliable approach for fitting geometric primitive shapes (e.g. circle, lines, planes, etc), given a noisy point cloud. More specifically, Sample Random Consensus RANSAC [Schnabel et al., 2007] and Hough Transform HT [Borrmann et al., 2011], have won their place as the State of the Art for model fitting.

Hough Transform, initially, was proposed for estimating models in 2D-space, for example lines, and circles, by determining specific values of parameters which characterize these patterns. Thereby, given place to different HT alternatives [Mukhopadhyay and Chaudhuri, 2015], and most
recently in [Borrmann et al., 2011] HT was improved to fit models in 3D-space.

On the other hand RANSAC emerged as an alternative to HT by offering a simple but effective method for model fitting. RANSAC is used to detect mathematical features like lines, circles, planes, etc. Basically, RANSAC is a greedy algorithm that takes random points from a point cloud and compares them against a mathematical model considering a threshold, thus grouping those points belonging to the sought model. In [Tarsha-Kurdi et al., 2007], RANSAC and 3D HT are compared for automatically detect roof planes from a point cloud. Authors conclude that RANSAC is more efficient in both segmented results and running time. It can process a large amount of input data in negligible time. On the other hand, 3D Hough transform is slower and more sensitive to parameterizing values.

Given the RANSAC effectiveness, several improvements of this method have been proposed in the literature, with the purpose of making RANSAC faster and more robust. In [Schnabel et al., 2007] proposed an algorithm that uses RANSAC to automatically detect basic shapes in an unorganized point cloud, it includes speed optimization while maintaining the accuracy of the result. With the aim of expanding the restriction of primitive shapes, in [Gelfand and Guibas, 2004] proposed a method for detecting High Level Structures including sphere, helix, plane, and cylinder. These works represent a framework for modeling a mathematical model in two and three dimensions.

The last paragraphs focused on giving a perspective of the most common methods for detecting geometric models given a set of 3-D points. Due to the proposed research is conducted by the extraction of High Level Structures in form of planes, the next paragraphs give an overview of some computer system whose main component relies on planes extraction based on parametrized methods and other alternatives.

In [Holz et al., 2011, Qian and Ye, 2013, Qian and Ye, 2014] depth estimation is carried out by using the kinect camera. Afterward, RANSAC algorithm is used to extract planes from the point cloud generated. The final application is oriented to robot navigation and object recognition in indoor environments. On the other hand, in [Dube and Zell, 2011, Drost and Ilic, 2015] HT method is modified for plane fitting, such that the reported method is optimized in terms of speed whereas result accuracy is maintained.

Note that RANSAC computational cost is related to how dense is the point cloud, meaning that the more image resolution, the denser is the point cloud. Motivated by the above, those works reported in [Alehdaghi et al., 2015] and [Tang et al., 2013] have focused on accelerating RANSAC by means of hardware designs implemented on GPU’s and FPGA’s devices respectively. Thus
optimizing RANSAC for real-time applications.

In spite of the fact that RANSAC and HT methods have proved to be an efficient approach for planes extraction given a dense point cloud like the one provided by a RGB-D sensor, some alternatives have been proposed in the literature. In [Sano et al., 2015] authors propose an improvement of the 6-DOF [Zhai and Milgram, 1998] algorithm in order to extract cube’s planes, thus performing an Augmented Reality application. In this context, geometric-features-based techniques are used in [Fernández-Moral et al., 2013] to extract planar structures; authors take advantage of the fact that man-made environments are usually dominated by large planes surfaces, in such a way that planes are extracted by formulating hypothesis based on global features such as: area, normal, centroid, vectors, etc.

2.2.2 Binocular Stereo Based Methods

In order to expand the restriction of indoor operation, binocular stereo vision becomes an appealing approach for High Level Structure extraction since it exhibits good performance in terms of depth accuracy and is robust to different environment conditions such as indoor and outdoor. In contrast to RGB-D based systems, there exist different techniques for estimating a 3-D point cloud whose major difference, as seen, lies in whether the stereo rig is calibrated or uncalibrated.

In general, by using calibrated stereo systems, a 3-D point cloud is composed of the following tasks: feature extraction, feature matching, and plane extraction. All steps are performed under the assumption that stereo images are rectified. Thus taking advantage of epipolar geometry but increasing the computational burden. In [Fouhey et al., 2010] authors propose a plane extraction method to detect buildings; First SIFT descriptors are extracted, after that, salient points are matched using nearest-neighbor search, and finally planes are extracted by means of J-linkage [Toldo and Fusiello, 2008], a randomized multi-model estimation algorithm. In [Barrera et al., 2013] plane extraction is carried out by computing a disparity map following a local window based approach, and then planes are extracted by defining a set of hypothesis according to color information of pixels. In [Pradeep et al., 2008] planes are extracted for step detection; first SIFT features are detected and matched, afterward, planes are extracted following a voting based method considering normals of each 3-D point.

On the other hand uncalibrated stereo systems consider un-rectified stereo images. Thus, in [Kanazawa and Kawakami, 2004] proposed a method for detecting planar regions with an uncalibrated stereo rig. the 3-D point cloud is estimated by first computing correspondence points
2.3 Related Work Taxonomy

So far, the description of the main approaches to address the problem of High Level Structures extraction, has been detailed. These approaches are outlined in Table 2.3, thus highlighting two main gaps uncovered by the state of the art: real-time operation and robustness to environment conditions. These constraints are essential qualities to perform applications in real-world for instance: aerial navigation/localization, augmented reality, obstacle avoidance, etc.

Motivated by developing a system capable of fulfilling the above constraints, this research tackles the problem of extracting 3-D models in form of planes, from man-made environments in real-time. The problem is carried out on the basis of binocular stereo, thus overcoming the environment operation constraint but dealing with the computational burden involved by rectification stage that has to run every frame-to-frame. Therefore, the latter step is avoided, thus dealing directly with un-rectified images. Hence, the 3-dimensional recovery is obtained from a semi-calibrated stereo rig, meaning that solely extrinsic stereo calibration parameters are known. In order to overcome the bottleneck of correspondence problem, a parallel design for feature matching is implemented on a GPU architecture.

For the sake of comparison, Figure 2.5 shows a taxonomy of the different approaches used for the state of the art, including the proposed research. This taxonomy helps to locate the computational elements used and the most important, it highlights the contributions of the proposed work.
<table>
<thead>
<tr>
<th>Approach</th>
<th>Depth recovery method</th>
<th>Plane extraction Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Holz et al., 2011]</td>
<td>Structured Light</td>
<td>RANSAC</td>
</tr>
<tr>
<td>[Qian and Ye, 2013]</td>
<td>Structured Light</td>
<td>RANSAC</td>
</tr>
<tr>
<td>[Qian and Ye, 2014]</td>
<td>Structured Light</td>
<td>RANSAC</td>
</tr>
<tr>
<td>[Dube and Zell, 2011]</td>
<td>Structured Light</td>
<td>HT</td>
</tr>
<tr>
<td>[Drost and Ilic, 2015]</td>
<td>Structured Light</td>
<td>HT</td>
</tr>
<tr>
<td>[Sano et al., 2015]</td>
<td>Structured Light</td>
<td>6-DOF</td>
</tr>
<tr>
<td>[Fouhey et al., 2010]</td>
<td>N/A</td>
<td>SIFT and J-linkage</td>
</tr>
<tr>
<td>[Barrera et al., 2013]</td>
<td>multi spectral stereo</td>
<td>Disparity map and Color information</td>
</tr>
<tr>
<td>[Pradeep et al., 2008]</td>
<td>Calibrated stereo methods</td>
<td>SIFT and voting scheme</td>
</tr>
<tr>
<td>[Kanazawa and Kawakami, 2004]</td>
<td>Fundamental matrix (Uncalibrated)</td>
<td>RANSAC</td>
</tr>
<tr>
<td>[Alehdaghi et al., 2015]</td>
<td>Structured Light</td>
<td>RANSAC on GPU</td>
</tr>
<tr>
<td>[Tang et al., 2013]</td>
<td>N/A</td>
<td>RANSAC on FPGA</td>
</tr>
</tbody>
</table>

**Table 2.3:** State of the Art overview

**Figure 2.5:** Taxonomy of the approaches for planes extraction
2.4 Summary

This chapter provided a general panorama of the different approaches for High Level Structure extraction. This revision included two fundamental tasks, the different methodologies for recovering a 3-D perspective of the world given an image, and the approaches for extracting 3-dimensional models from a point cloud. By this revision, two main gaps uncovered by the state of the art were highlighted: real-time operation and robustness to environment conditions. Thus given place to a research opportunity which is the motivation for this research. In addition, a point of comparison between the proposed work and the state of the art was given, in order to underline the main contributions of this study.
CHAPTER 2. RELATED WORK
Chapter 3

Theoretical Framework

Last chapter emphasized the major differences between the related work and the proposed approach. The current section provides the theoretical basis that gives support to the proposed method. Despite some concepts were introduced in the previous section, a further explanation of the basis related to stereo vision, high performance computing, and model fitting is given in this chapter.

This revision is conducted by the research motivation, a computer vision system capable of extracting High Level Structures at a high frame rate, using a semi-calibrated stereo rig. Thus, the fundamentals basis are presented considering a visual system that receives as input, a pair of un-rectified stereo images where only extrinsic calibration parameters are known, and yields as output, a set of 3-D points belonging to the planes of the captured scene.

3.1 Feature Detection

One of the main components in the vision system is the capability of giving a short, unique and computational representation of an image. In this sense, feature detection has won its place as an effective approach to solving this task. Feature detection refers to techniques that aim at computing abstractions of image transformation, this process involves two stages: feature extraction and description. Since this task has been widely explored in the literature, these are classified in global and local features.

Global features refer to characteristics of image regions such as size, perimeter, histograms, etc. They are commonly used in systems that require generalizing an entire object with a single
vector, therefore their use in standard classification techniques is straightforward. On the other hand, Local features refer to characteristics of small regions in the object, typically they are patterns which differ from their immediate neighborhood. Consequently, they are computed at different points and yield better results in terms of robustness against occlusion and clutter.

Given the qualities of Local Features, they play an important role in those applications that require an accurate representation of the image, such as image retrieval, object recognition, object tracking, etc. Several properties are considered to define a local feature as efficient and robust, but the main are repeatability and distinctiveness. The former takes importance when we have two images of the same scene, for example in stereo vision, thus repeatability refers to the capability of finding a high percentage of features on the visible part of both images, this is, the region of features extracted in the first image should be the same area of features found in the second image. The second property, distinctiveness, dictates that detected features should show a lot of variation, such that, features can be distinguished and matched.

Over the last years, a great variety of local descriptors has been proposed in the literature. Actually, this is a very active field of study since current vision applications are demanding accurate and fast local features. Accordingly, these have been grouped according to their output data type: Floating point and Binary features descriptors. The former have proved to fulfill the properties above mentioned, such that they are a reliable approach for applications such as face recognition, reconstruction, image retrieval, etc. However, they demand complex operations thus increasing the computational burden. Binary descriptors, on the other hand, emerged as an alternative to floating-point features, by offering low memory footprint, ease to compute, and a fast descriptor comparison, thus becoming a suitable approach to those applications with real-time requirements. In the remaining section, an overview of floating point as much as binary descriptors is presented.

**Floating point features.** SIFT [Dalal and Triggs, 2005] is a feature detector and descriptor considered the most popular floating point descriptor since it yields good performance in terms of robustness to affine transformations such as rotation, illumination changes, view-point change, blurring, etc. Basically, it consists of four stages: scale-space extrema detection, keypoint (point of interest in the image) localization, orientation assignment, and keypoint descriptor. Firstly, potential interest points are detected by using difference-of Gaussian DOG [Lowe, 2004]. Then those points with low contrast are eliminated by means of Hessian matrix [Nelder and Mead, 1965]. In the third step, an orientation component is computed by using orientation histogram, formed from those points within a region around the keypoint detected. Thus constructing a descriptor with arrays of histograms and orientation bins.
3.1. FEATURE DETECTION

With the spirit of developing a descriptor with a similar performance to SIFT but faster to compute, SURF (Speeded-Up Robust Features) [Bay et al., 2006], emerged as a solution to those applications that require fast computations. The SURF detector is an improvement of Hessian detector that relies on integral images to reduce the computation time. In order to construct the SURF descriptor, firstly an orientation component is computed by calculating the Haar-wavelet [Chen and Hsiao, 1997] responses in x and y-direction of the keypoint. Accordingly, the descriptor extraction is carried out by describing a distribution of Haar-wavelet responses within a square region centered around the interest point, and oriented along the orientation component.

In spite of the fact that floating-point descriptors are appealing for practical uses as they show good performance in terms of repeatability and distinctiveness, they involve complex operations, thus increasing the processing time. Therefore, hardware implementations of the above descriptors, have been proposed in the literature, such as those works reported in [Heymann et al., 2007, Yao et al., 2009, Svab et al., 2009]. However, the reported time consumption evaluation is not well enough, especially given the rise of real-time and mobile-based applications thus giving way to Binary Descriptors.

**Binary Descriptors.** BRIEF is one of the initial binary descriptors, proposed by M. Calonder et al. [Calonder et al., 2010]. Basically BRIEF takes a smoothed image patch around a keypoint and it makes pixel intensities comparisons in order to construct a binary descriptor. Its performance is similar to floating point descriptors in many aspects, including robustness to lighting, blur, and perspective distortion. However, it is very sensitive to in-plane rotation.

From the above, Ethan Rublee et al [Rublee et al., 2011]. Proposed a BRIEF descriptor that is invariant to orientation called ORB (Oriented FAST and rotated BRIEF), the main contributions in this work lies in adding an orientation component to FAST [Rosten and Drummond, 2006] feature detector and proposing a learning method for choosing pairwise tests with good discrimination power and low correlation response among them.

Similar to ORB, in [Leutenegger et al., 2011] Leutenegger et. al proposed a binary descriptor invariant to rotation and scale. It uses the AGAST corner detector [Mair et al., 2010], which is an improvement of FAST. This binary descriptor is constructed by pixel comparison whose distribution forms a concentric circle surrounding the feature.

Based on human retina, in [Alahi et al., 2012] a binary descriptor was proposed, it received the name of FREAK(Fast Retina Keypoint), this is only a feature descriptor that heavily relies on a robust feature detector algorithm for keypoint detection, therefore SURF and SIFT feature
detectors are commonly used as a previous step to FREAK.

In [Alcantarilla and Solutions, 2011] Pablo F. Alcantarilla et al. Proposed an improvement of KAZE features [Alcantarilla et al., 2012] in terms of computational complexity and descriptors storage, by using numerical schemes called Fast Explicit Diffusion (FED) [Goesele et al., Grewenig et al., 2013], and modifying the LDB descriptor [Yang and Cheng, 2012]. This binary descriptor named A-KAZE, is faster to compute than SIFT, SURF, and KAZE, and exhibits similar performance.

For the sake of clarity, Table 3.1 shows a brief overview of the descriptors presented. In addition, Figure 3.1 depicts the feature description process for those binary descriptors whose method relies on pixel intensity comparisons. Note that a distinguishable difference among them, is the pattern from which the binary test are carried out.

<table>
<thead>
<tr>
<th>Feature Descriptor</th>
<th>Feature Detector</th>
<th>Method-based</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIFT</td>
<td>SIFT</td>
<td>DOG / Orientation Histograms</td>
<td>Floating-point</td>
</tr>
<tr>
<td>SURF</td>
<td>SURF</td>
<td>Hessian Matrix / Haar-wavelet</td>
<td>Floating-point</td>
</tr>
<tr>
<td>BRIEF</td>
<td>FAST</td>
<td>Pixels intensity comparison</td>
<td>Binary descriptor</td>
</tr>
<tr>
<td>ORB</td>
<td>o-FAST</td>
<td>Pixels intensity comparison</td>
<td>Binary descriptor</td>
</tr>
<tr>
<td>BRISK</td>
<td>AGAST</td>
<td>Pixels intensity comparison</td>
<td>Binary descriptor</td>
</tr>
<tr>
<td>FREAK</td>
<td>N/A</td>
<td>Pixels intensity comparison</td>
<td>Binary descriptor</td>
</tr>
<tr>
<td>A-KAZE</td>
<td>A-KAZE</td>
<td>Gradient Information</td>
<td>Binary descriptor</td>
</tr>
</tbody>
</table>

**Table 3.1**: Brief overview of the most significant local descriptors reported in the literature.

![Feature Detection](image1.png)  
**Figure 3.1**: Feature Detection process for the binary descriptors whose method relies on pixel intensities method.
3.2 Depth Recovery

The aim of computer vision is to recover a model of the real world from two-dimensional images. Therefore, this section focuses on twofold: Image Formation and Depth Recovery. Hence, this section introduces the image formation process, following with the theoretical fundamentals to recover 3-D information from stereo vision.

3.2.1 Image Geometry

The image formation process involves two stages: the image formation geometry and the physics of the light. The former determines the relationship between the real-world and image coordinates. The second one determines the brightness of a point in the image plane. Since most of the stereo vision algorithms do not require an understanding of the physics of light, this section only introduces the image geometry.

Before giving the technical details involved in the image formation geometry, it is important to take into consideration the camera operation. Basically, a camera is composed of lens which take advantage of the refraction effect to create a two-dimensional projection of the 3-D world. The basic camera model is known as the pinhole, Figure 3.2 depicts the operation mode of this model, which considers a device with a tiny aperture in the center and light rays traveling thorough the pinhole to the image plane, eventually forming the 2-D image from the 3-D object.

![Figure 3.2: Pinhole camera model.](image)

Perspective projection refers to the math representation of the camera pinhole model. This is illustrated in the Figure 3.3, where an object point in three dimensions is projected onto the image plane. Note that, the image plane is at a distance \( f \) from the projection center, and the
projected image is inverted. Thus giving way to Front Perspective Projection Model showed in Figure 3.4. This model avoids image inversion by moving the image plane from the back to the front of the projection center. Front Perspective Projection Model is considered to describe the stereo equations in the next sections.

![Figure 3.3: Perspective Projection.](image1)

![Figure 3.4: Front Perspective Projection.](image2)

The perspective projection help us to compute the position \((x',y')\) in the image plane of an object point \(P(x,y,z)\), by finding the intersection of the line of sight passing through the object point, with the image plane as shown in Figure 3.4.

The distance of the object point with respect to z-axis is \(r\), and the distance of the projected point from the origin of the image plane \(r'\). Whereby, a triangle is formed from the z-axis, the line of sight to object point and \(r\). Another triangle is formed from the z-axis, the line of sight to point \((x',y')\) and \(r'\). since both triangles are similar, the equation 3.1 is defined as:

\[
\frac{f}{z} = \frac{r'}{r} \tag{3.1}
\]

Another two similar triangles are formed. The first one from the x and y coordinates and \(r\).
3.2. DEPTH RECOVERY

The second one from \( x' \) and \( y' \) coordinates and \( r' \). From which, equation 3.2 is defined as:

\[
\frac{x'}{x} = \frac{y'}{y} = \frac{r'}{r}
\]  

(3.2)

By combining equations 3.1 and 3.2 yields the position \((x', y')\) in the image of an object point \(P(x, y, z)\), as followed:

\[
x' = \frac{f}{z}x
\]  

(3.3)

\[
y' = \frac{f}{z}y
\]  

(3.4)

3.2.2 Stereo Geometry

The simplest binocular stereo model is the parallel axes configuration showed in Figure 3.5. It considers two identical cameras separated only in the \(x\) direction by a known distance called baseline. In this model, the image planes are coplanar. The stereo field of view refers to the intersection of each camera field of view. This is an important aspect to take into consideration before assembling a stereo rig, since depth information is only inferred for those object points inside this area. In this sense, the stereo field of view is closely related to the baseline, as a larger baseline corresponds to a narrow field of view, on the contrary, a larger field of view corresponds to a small baseline.

Binocular stereo vision takes advantage of a conjugate pair and disparity to compute the depth of an object point. The former refers to two points in left and right images that are the projections of the same point in the stereo field of view. In this regard, the disparity is defined by the difference between points of a conjugate pair \((x_L-x_R)\). Figure 3.5 shows a top-view of the binocular stereo rig, where an object point \(P(x, y, z)\) is projected on left and right images.

Based on geometry of the parallel stereo rig the object point depth is computed following the equation 3.5

\[
Z = \frac{bf}{x_L - x_R}
\]  

(3.5)

On the other hand, cameras in arbitrary position and orientation is a more common stereo geometry configuration. In contrast to parallel axes, this model allows a larger stereo field
of view since it does not depend on baseline size. In this configuration, the disparity is relative to the orientation of each camera (left and right). In this case, the object point is projected on the left and right coordinate system following the equations 3.6 and 3.7 respectively. From these equations, the projection of the same point on the left and image plane is computed by equations 3.8 and 3.9.

\[
\begin{pmatrix}
    x_l \\
    y_l \\
    z_l
\end{pmatrix} = \begin{pmatrix}
    \cos(\theta) & 0 & \sin(\theta) \\
    0 & 1 & 0 \\
    -\sin(\theta) & 0 & \cos(\theta)
\end{pmatrix} \begin{pmatrix}
    x - \frac{b}{2} \\
    y \\
    z
\end{pmatrix}
\]

(3.6)
3.2. DEPTH RECOVERY

\[
\begin{pmatrix}
X_r \\
Y_r \\
Z_r
\end{pmatrix}
= \begin{pmatrix}
\cos(\theta) & 0 & -\sin(\theta) \\
0 & 1 & 0 \\
\sin(\theta) & 0 & \cos(\theta)
\end{pmatrix}
\begin{pmatrix}
x + \frac{b}{2} \\
y \\
z
\end{pmatrix}
\]  

(3.7)

\[
x_l = \frac{f \cdot X_l}{Z_l}, y_l = \frac{f \cdot Y_l}{Z_l}
\]  

(3.8)

\[
x_r = \frac{f \cdot X_r}{Z_r}, y_r = \frac{f \cdot Y_r}{Z_r}
\]  

(3.9)

This subsection showed the two assemblies for a binocular stereo rig. On one hand, parallel axes configuration makes depth computation easy to compute as it depends on disparity, but the stereo field of view is restricted by baseline. On the other hand, cameras in arbitrary position and orientation configuration makes the stereo field of view an independent variable, thus covering all the angles of the object, however depth is estimated by triangulation. In the next section, the process for computing depth information, independently of the stereo configuration, is detailed.

3.2.3 Depth from Binocular Stereo

So far, the image formation process, its math representation and the stereo geometry was detailed. These aspects are fundamental to understand the theoretical basis for computing depth information of the world given a binocular stereo. This process involves several stages such as: calibration, rectification, stereo correspondence, and triangulation. Figure 3.7 shows an overview of a typical stereo vision system. Since each stage affects the 3-D recovery performance, the description of each stage is given in this section.

Calibration

As mentioned, image geometry as much as stereo geometry allow to establish a relationship between the 3-D object in the world and the image in two dimensions. The derivatives equations of the image/stereo geometry rely on internal and external camera parameters. On one hand, internal camera parameters refers to those points which describe the internal camera geometry, such as:

- Focal length: is the distance between the image plane and the center of projection.
- Principal point: it refers to the location of the origin of the image plane coordinate system.
• Lens distortion coefficients: they help to remove image distortion.

On the other hand, external or extrinsic camera parameters establish the relationship between the camera reference frame and the scene reference frame. In other words they refer to orientation and translation matrices. Typically, the calibration is carried out using a calibration pattern, commonly it consists of a grid of circles or squares. The aim of calibration process is to obtain a set of spatial points from the calibration pattern in order to infer the projection equations that links spatial and image points.

**Stereo Calibration**

Different approaches to address stereo calibration have been proposed in the literature, such as those works reported in [Horaud et al., 2000, Baek et al., 2015, Ke and Sutton, 2012]. However, a common denominator is the following process: first, the intrinsic and extrinsic parameters of each stereo camera are obtained. Then, the external parameters of stereo cameras are computed. Finally, the full-calibration process yields coplanar stereo images, thus computing depth based on a parallel axes configuration. Several tasks are involved in stereo calibration process such as: compute intrinsic parameters of each stereo camera, fulfill epipolar constraint, find correspondences between features, and triangulation. Since epipolar geometry makes stereo vision an appealing approach to infer 3-D information of the world, the epipolar constraint is described in this section.

The **epipolar geometry** is the geometry of the image planes intersection. It only depends
3.2. DEPTH RECOVERY

on the cameras internal parameters and relative pose. This geometry allows to find the relation between the object points projected on left image plane (x) and right image plane (x’), The entities involved in epipolar geometry are illustrated in Figure 3.8.

![Stereo Epipolar Geometry](https://via.placeholder.com/150)

**Figure 3.8:** Stereo Epipolar Geometry.

The terminology related to epipolar geometry is:

- Epipolar plane: is the intersection area of the field of views of each stereo camera, including the base line.
- Epipolar line: Is the intersection of an epipolar plane with the image planes.

Epipolar geometry is motivated by the scenario where only x point is known, and we are interested in finding where x’ point is constrained. Therefore, as the epipolar plane π is determined by the baseline and the ray defined by x, analogously the point x’ lies in π. From the above analysis, the point x’ lies on the epipolar line l’. Given a problem of finding correspondences, the search for the point corresponding to x is restricted to the line l’ in contrast to search on the entire image.

The fundamental matrix is the algebraic representation of epipolar geometry. This a 3x3 homogeneous matrix which satisfies:

\[ x'^T F x = 0 \]  (3.10)

for all corresponding points \( x \leftrightarrow x' \)
The fundamental matrix is highly relevant in the field of stereo algorithms, since it provides information to compute epipolar lines. As a consequence, its computation is a fundamental stage previous to a feature matching process, rectification, the computing of external parameters (stereo calibration), etc.

Rectification

Following the scheme illustrated in Figure 3.7. The next step after the stereo calibration parameters have been obtained, is the rectification. This stage aims at removing lens distortion and turning the binocular stereo rig into a model with parallel axes. This stage is motivated by the assumptions considered in a stereo system with coplanar images:

1. Depth information of an object point is computed from disparity.
2. Feature matching process take advantage of the epipolar line.

Basically, rectification is the process of re-sampling stereo images such that epipolar lines correspond to image rows. The re-sampling consist in applying 2-D projective transformations to the two images in order to match the epipolar lines. Consequently, this process helps to reduce the problem to epipolar geometry considering parallel axes.

The rectification algorithm consists of six steps. As input, two distorted stereo images are considered. The output is a pair of images with parallel x-axes.

1. Distortion removal, following any of these methods [Tordoff and Murray, 2000, Wang, 2015].
2. Find at least seven corresponding points between left and right stereo images.
3. Compute the fundamental Matrix F and find epipoles.
4. Choose a projective transformation $H'$ that maps the epipole to the point at infinity, following the method in [Hartley and Zisserman, 2003a].
5. Find the matching projective transformation $H$ that minimizes the least.squares distance, following the method in [Hartley and Zisserman, 2003b].
6. Resample left and right images according to projective transformation $H'$ and $H$ respectively.
3.2. DEPTH RECOVERY

![Diagram of image rectification](image)

**Figure 3.9:** Example of image rectification.

### Feature Matching

Feature matching is the process of finding conjugate pairs in the stereo images. Due to this stage involves a significant computational burden, it has been widely explored in the literature and named as the correspondence problem. This is stated as follows: for each point in the left image plane, find the corresponding point in the right image.

Different approaches have been proposed in the literature to tackle the correspondence problem. Most of them, strongly rely on rectified images, so that they take advantage of the assumptions considered in a stereo rig with parallel axes. Examples of the above, are those works reported in [Pollard et al., 1985, Veksler, 2003, Pilu, 1997] .

Typically, feature matching approaches without considering rectified images follow a brute force strategy, thus making this process computationally intensive. Therefore, in [Martinez-Carranza et al., 2013] hashing techniques are used to reduce the search on the entire image. Consequently, the computational burden is reduced and increasing the operation frame rate. However, it is not enough for those applications that require a real-time performance more than 30fps.

### Triangulation

Following the scheme in Figure 3.7, now that salient points of the stereo images have been matched, we are able to compute the 3-D position of each matched point, thus generating a depth map of the object. Triangulation is the process of finding the 3-space position of an object point given its image in two views and the camera matrices of those views. Several triangulation methods
have been proposed in the literature, as this is a challenging problem due to the image noise and inaccurate matching outcomes. Those methods rely on both stereo geometry with parallel axes and stereo cameras in different position and orientation. Depth computation from parallel axes has been discussed in section 3.2.1. Therefore, the triangulation methods based on uncalibrated stereo images are described in this subsection.

Basically, the triangulation process consists in finding the intersection of the ray-lights passing from the cameras center through image points, to the object point. As is illustrated in Figure 3.10. Ideally, these rays are intersected at some point, however in some cases, these points pass close each other.

The following methods assume that camera matrix $P$ and matched points $u \leftrightarrow u'$ were previously computed. The most common triangulation method is **Direct Lineal Triangulation (DLT)**, this is a suitable approach for those cases which consider two or more cameras. This method lies in the assumption that $u = Px$ where $x$ represents the object point viewed from one camera. Then homogeneous coordinates are written as $u = w(u, v, 1)$, where $(u, v)$ are the image points coordinates and $w$ is an unknown scale factor. Now denoting $p_i^T$ the $i-th$ row of the matrix $P$, this equation may be written as follows:

$$wu = p_1^T x, \quad vw = p_2^T x, \quad w = p_3^T x,$$  \hspace{1cm} (3.11)

Eliminating $w$ using the third equation yields:

$$up_3^T x = p_1^T x, \quad vp_3^T x = p_2^T x$$  \hspace{1cm} (3.12)

The above equations describe one view. Accordingly the process is repeated for the second view, thus obtaining a total of 4 linear equations in the coordinates of the $x$, which may be written in the form $Ax = 0$ for a suitable $4 \times 4$ matrix, $A$. After that, by setting $x = (x, y, z, z)^T$ the set of homogeneous are turned into no-homogeneous equations with 3 unknowns. Finally this equation system is solved by using Singular Value Decomposition.

Considering the equation 3.12, note that the result of point $x$ will not satisfy the equation exactly. There will be an error $\epsilon = up_3^T x$. In order to minimize the error, DLT process is repeated $n$ times, changing the scale factor $w$ in each iteration. Finally, $x$ is selected according to the least $\epsilon$ value. This method is called **Iterative DLT** and yields more accurate estimation of $x$.

The simplest but effective triangulation method is **Line Intersection**. As well as above
methods, this considers homogeneous coordinates $u=(u, v, 1)$, $u'=(u', v', 1)$ and the camera matrices $P_1$ and $P_2$. Therefore, the equations of rays originating at principal point or camera center point, and passing through the two matched points $u$ and $u'$, are given by:

\[
\begin{align*}
(wu, wv, w)^T &= P_1(x_i, y_i, z_i, 1)^T \quad (3.13) \\
(w'u', w'v', w')^T &= P_2(x_i, y_i, z_i, 1)^T \quad (3.14)
\end{align*}
\]

Since matched points and camera matrices are known, whereas, $x, y, z, w, w'$ are unknown, we have six equations from which the intersection point $x=(x, y, y)$ is computed. Thus obtaining the object position in 3D-space.

This section has summarized the three most used triangulation methods. However several different alternatives have been proposed in the literature, for example the works reported in [Hartley and Sturm, 1997]

\section{3.3 High Performance Computing}

Generally, High Performance Computing refers to the design of aggregating computing power to a system in a way that yields higher performance than one could deliver [Schroeder and Gibson, 2010]. Nowadays, this research field has gained an important relevance given the increasing of transistors on a silicon wafer. Electronic circuits designers have focused on aggregating more co-processing units instead of increasing the operation frequency. Thus, given place to powerful hardware platforms in terms of computational performance such as GPU’s and FPGA’s.
3.3.1 GPU’s

A GPU (Graphic Processing Unit) is similar to a CPU (Central Processing Unit) however the former contains thousand of cores [Nickolls and Dally, 2010]. The difference between them lies in the way the computational process is performed, CPU’s consist of a few cores optimized for carrying out a task in a sequential way, while a GPU contains much smaller cores designed for handling multiple tasks simultaneously. In this regard, NVIDIA is considered the leader in the design of GPU’s architectures.

NVIDIA’s computing architectures [Nvidia, 2008] consist of a set of parallel multiprocessors that are further split into many cores, each of them executes instructions from one thread at a time thus performing massive parallelism. Figure 3.11 shows the common model of a NVIDIA GPU, which consist of different memory levels, each of them with different size and speed, respecting a hierarchy memory in a such a manner that the more size the less speed achieved. Therefore, in order to perform a significant acceleration, before starting to design a parallel model, it is important to take into consideration the data-type and data-path.

![Figure 3.11: CUDA memory model.](image)

CUDA (Compute Unified Device Architecture) is used to develop software for graphic processors. This is similar to C language, but CUDA includes several specific instructions in order to communicate with the GPU. An important CUDA’s quality is that it enables to configure each GPU component, such as texture memory, shared memory, the number of threads and blocks, etc.

CUDA programming paradigm [Sanders and Kandrot, 2010] is a combination of serial and parallel executions. Figure 4.3 shows an example of this type of programming. The serial code runs in CPU also called host. Then, parallel tasks run in GPU, named as device, which consists of
3.3. HIGH PERFORMANCE COMPUTING

a grid that contains several blocks or set of threads. Parallel execution is expressed by the kernel function that is executed by the blocks in parallel. Finally, the result is returned to host.

Figure 3.12: CUDA programming scheme.

Figure 3.13 shows the internal structure of a block, that consist of a set of threads that communicate each other by shared memory, thus making possible the synchronization between them. In order to make easier the design of parallel models, blocks may be configured in one, two and three dimensions. This is very useful when we are dealing with images, because blocks are configured in two dimensions in such a way that every pixel is processed by each thread, thus dramatically accelerating the processing time.

Figure 3.13: Internal structure of a CUDA block.

3.3.2 FPGA’s

An FPGA is a programmable chip designed to implement combinational and sequential logic circuits. Basically, it consist of a matrix of configurable logic blocks which are composed of memories.
These blocks are communicated by an interconnection matrix. As much as GPU’s, these platforms have been broadly used to accelerate different applications.

The design of logic circuits is a mandatory task to develop processors well-fitted for a specific algorithm. In such a way that the pipeline, datapath, ALU’s, are specifically designed to solve specific tasks. Therefore, it is possible to design parallel schemes and improving the performance of several applications.

VHDL [Perry, 1998] is a Hardware Description Language used to write text models that describe a logic circuit such as: AND, OR, XOR, NAND, etc. This is very useful for hardware design in FPGA’s and produce the RTL (Register Transfer Level) schematic of a certain circuit.

Given the fact that working with FPGA’s the designer is not tied to deal with a specific hardware architecture, these devices play an important role in those intensive computationally applications. For example in computer vision, communications systems, security, etc.

3.4 Parametrized Algorithms

Parametrized Algorithms refer to those techniques oriented to fit a model given both reference model parameters and sample points. The most common parametrized methods in the literature are RANSAC [Schnabel et al., 2007] and Hough Transform HT [Ballard, 1981], 3D and 2D models such as cubes, planes, lines, circles, etc. are considered by these methods.

Initially, Hough Transform was proposed for estimating models in 2D-space, for example lines, and circles, by determining specific values of parameters which characterize these patterns. In the literature, there are different HT alternatives oriented to several applications. Recently, HT was improved to fit models in 3D-space, such as [Borrmann et al., 2011].

RANSAC emerged as a simple but effective solution for fitting models from scattered points. Given its capability of reducing outliers, RANSAC has been widely used in different applications such as camera calibration, feature matching, 3-D structures fitting, etc.

For the sake of clarity, the process of fitting a line given a set of points using RANSAC, is illustrated in Figure 3.14. First, two random points are selected from the given data set. Then, line model is parametrized considering the points selected. After that, a random point is selected, and the distance between the point and the parametrized model is computed, if this distance is less than a threshold, then the random point is aggregated to a set of points. Finally, this process
3.5 Summary

This chapter gave the theoretical basis to understand the proposed research and providing a founded argument in terms of work viability. First, the fundamental basis for inferring depth from a binocular stereo were presented, including calibration, local features, correspondence, and triangulation. Afterward, the two main hardware architectures (GPUs and FPGAs) used to accelerating algorithmic stages were described. Finally a brief review of the RANSAC algorithm, commonly used for modeling geometric shapes, was given at the end of the chapter. The following chapter details the proposed method for the extraction of High Level Structures in form of planes.
Chapter 4

Proposed Approach

This chapter presents the developed research for the extraction of High Level Structures in form of planes. As mentioned at the beginning of the document, the extraction of planar structures in real time, is addressed by means of the following stages: extraction of binary descriptors, feature matching, 3-D point cloud estimation, and plane extraction. The stage sequence is sorted as Figure 4.1.

![Figure 4.1: Work-flow for planes extraction.](image)

4.1 Binary Descriptors Extraction

As it has been seen in the last chapter, the extraction of visual descriptors is a fundamental task in stereo vision. Hence, diverse local features have been proposed in the literature for performing diverse applications, including 3-D recovery of the world. Keeping in mind that real-time operation is a fundamental quality for current computer systems, several visual descriptors have proposed in order to make them accurate and fast to compute.

From the above, binary descriptors play an important role in those applications with real-time constraints while maintaining results accuracy. Since this research seeks to estimate a point cloud at a high frame rate, different binary descriptors were explored in this study, from the pio-
neer BRIEF to the current A-KAZE descriptor. In [Miksik and Mikolajczyk, 2012] several binary descriptors are assessed using the recall-precision metrics [Mikolajczyk and Schmid, 2005]. This evaluation pointed out that ORB is the best binary descriptor in terms of accuracy and processing time, considering a feature matching application. From the latter argument, the ORB descriptor becomes the main core for point cloud estimation.

A brief description of the most common binary descriptors was presented previously. However, in this section, the ORB descriptor algorithm is described in more detail. As mentioned, a local feature is composed of a keypoint (point of interest in the image) detector and its descriptor. In this sense, the next paragraphs provide a brief introduction of the oFAST detector and ORB descriptors is provided.

**oFAST.** The FAST [Lowe, 2004] salient point detector is widely used due to its performance and computational properties, however FAST salient points do not have an orientation component. Therefore, in [Rublee et al., 2011] a metric of corner orientation is used to make the FAST detector suitable to describe a feature invariant to orientation. This feature detector received the name of oFAST (oriented FAST). The orientation component is calculated based on the intensity centroid [Rosin, 1999]. Since corner’s intensity is offset from its center, the vector constructed between them, may be used to induce orientation. According to Rosin [Rosin, 1999], the centroid of a patch is defined as:

\[ C = \left( \frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \]  

(4.1)

where \( m \) represents the moments of a patch, defined as:

\[ m_{pq} = \sum_{x, y} x^p y^q I(x, y) \]  

(4.2)

Then, a vector from the corner’s center to the centroid is constructed in order to compute the patch orientation:

\[ \theta = \text{atan2}(m_{01}, m_{10}) \]  

(4.3)

**ORB descriptor** The ORB descriptor emerged from the need of making BRIEF descriptor invariant to orientation. BRIEF is a descriptor that uses binary tests between pixels in a smoothed image patch. Keeping the same idea, ORB’s authors proposed to steer BRIEF according to the orientation component previously computed, thus making BRIEF invariant to orientation. More
4.1. BINARY DESCRIPTORS EXTRACTION

specifically, if \( p \) is a smoothed image patch, corresponding binary test \( \tau \) is defined by:

\[
\tau(p; x, y) := \begin{cases} 
1 & \text{if } p(x) < p(y) \\
0 & \text{otherwise}
\end{cases}
\]  

(4.4)

where \( p(x) \) is the intensity of \( p \) at a point \( x \). The feature is defined as a vector of \( n \) binary tests:

\[
f_{nd}(p) := \sum_{1<i \leq 128} 2^{i-1} \tau(p; x_i, y_i)
\]  

(4.5)

In order to construct a BRIEF descriptor that presents good performance in terms of speed, storage, efficiency, and recognition rate, it is important to take into account two elements: descriptor’s length and binary tests distribution. For the former, descriptors of 128, 256, and 512 bits proved to exhibit good discrimination performance, with 256 being the current standard size. On the other hand, many different types of distributions were considered in [Calonder et al., 2010] for selecting \( nd \) test locations. Figure 4.2 taken from [Calonder et al., 2010] shows the explored distributions, where experimental results reported that G III, a Gaussian BRIEF pattern, performs better.

As noted, no complex operations are required to compute ORB, a quality that makes the descriptor fast to calculate. Moreover, it requires a low footprint memory and the comparison between descriptors is fast to calculate because ORB representation is given by a binary vector. These characteristics, make the ORB descriptor an appealing approach for performing the first stage of the method for High Level Structure extraction. With the aim of accelerating ORB descriptor construction, this is implemented on a GPU architecture using the OpenCv library, thus dramatically reducing the processing time.
Feature extraction and description is a fundamental step for generating a point cloud. Actually, the point cloud density depends on this stage, thus impacting the overall performance of the plane extraction. The next step consists in matching the descriptors of left and right stereo images in order to provide all the elements for computing depth from stereo geometry.

4.2 Feature Matching

The last section provided an unique and short representation of each stereo image, by extracting and describing local features. This representation may be seen as a set of 2-dimensional points of each image. The following stages aim at estimating the depth information of each point, thus generating a 3-dimensional perspective of the world. In this sense, the stereo calibration parameters along with the correspondence points, provide the necessary information to infer depth from cameras geometry. The correspondence problem is tackled in this section, by describing the proposed approach for accelerating the feature matching process, considering a semi-calibrated stereo rig.

As seen in the previous section, the correspondence problem, better known as feature matching, requires an intensive computational process, thus un-fulfilling real-time constraints, especially in those cases where high resolution images are considered. Therefore, several approaches in the literature have focused on accelerating this process by reducing the search area. This is commonly achieved by using calibrated images, as search area is limited by epipolar line. However in this research, un-rectified images are considered as input, meaning that the features search is carried out along the entire image, thus highly increasing the computational cost.

Motivated by real-time constraints and the intensive computing of feature matching process, this research addresses the correspondence problem by means of designing a novel hardware implementation for feature matching, using a Graphic Processing Unit. The proposed parallel model is based on a brute force matching, since this approach guarantees that all features are compared, reducing the amount of false positives, an indispensable quality to generate an un-noisy point cloud.

The brute force matching consists in comparing a descriptor of one image against all descriptors of the target image, completing this process until all descriptors of the first image are compared. The comparison metrics rely on the feature representation, in such a way that floating-point descriptors are compared in the euclidean distance, whereas binary descriptors are compared in Hamming space. The Hamming distance of two vectors is the number of coefficients in which they refer. For instance, given two vectors \( a = 00111 \) and \( b = 11001 \), the Hamming distance between \( a \) and \( b \) is 4. Therefore, the smaller distance between vectors, the more correlated they
4.2. FEATURE MATCHING

are. Note that Hamming distance computation requires straightforward operations, making binary descriptors fast to compare.

In order to fulfill real-time constraints, CUDA’s architecture massive parallelism is leveraged to exhaustively calculate and compare the Hamming distance between every binary descriptor in the left stereo pair and all descriptors in the right stereo pair. This design exploits the CUDA memory hierarchy: shared and textured, to reduce memory traffic thus improving the performance in terms of processing time. A set of three kernels is required to fully calculate corresponding matches for the number of features determined (2000 feature as maximum). Figure 4.3 shows a data flow diagram per kernel.

Algorithm 1 is the pseudo-code of the proposed approach for feature matching, following the CUDA programming scheme. First, the left and right descriptors arrays are stored in texture memory. This is a fast read-only memory well-suited for storing data rarely updated but read often, such as the binary arrays. After that, features are matched by a CUDA parallel model split into three kernels. Finally, outliers are removed by comparing the geometric distance between matched points with the baseline. The following paragraphs explain the above in detail.
Algorithm 1: Feature Matching Process

**Input:** LeftD, RightD, baseline

**Result:** Matches

1. toSharedMemory(LeftD);
2. toSharedMemory(RightD);
3. kernel1\(\lll 32, (32, 32) \ggg\);
4. kernel2\(\lll 1024, (32, 32) \ggg\);
5. kernel3\(\lll 1, (32, 32) \ggg\);
6. deviceToHost(Matches)
7. delete those matches less than baseline

Kernel 1 aims at computing the Hamming distance between every binary descriptor extracted in the left stereo image and all descriptors extracted in the right stereo image. Firstly, an XOR operation between the first left descriptor and all descriptors of the right image is performed in parallel yielding a matrix of N*32 resided in shared memory, where N is the number of descriptors. After that, row matrix values are summed by reduction approach. Finally, the result which corresponds to the Hamming distance, is stored in the first column of the output matrix. This process is repeated until filling the N x N matrix, which contains the Hamming distances for all descriptors. The technical details are given by Figure 4.4 and the pseudo-code of Algorithm 2.

![Figure 4.4: Hardware design for computing Hamming distance (kernel 1)](image)
### Algorithm 2: kernel 1, computing Hamming distance

**Input:** A: Left Image Descriptors, B: Right Image Descriptors. In texture memory

**Result:** O Hamming Distance (A,B)

1. $i = \text{threadId.x}$;
2. $j = \text{threadId.y}$;
3. $k = \text{blockId.x}$;
4. $x = k \times \text{blockDim} + i$;
5. $y = k \times \text{blockDim} + j$;
6. $O_{xy} = A_{0y} \oplus B_{xy}$;
7. **for** $s = \text{blockDim.x}/2; s > 0; s >>= 1$ **do**
   8. $O_{xy} += O_{xy+s}$
8. **end**
9. **syncthreads();**
10. This process is repeated for all descriptors in Left Image

Kernel 2 aims at computing the minimum Hamming distance of each feature. Each of the obtained distances is compared by following a reduction approach, similar to kernel 1. This process is very fast since it is performed by every thread in parallel. The output is represented by a vector of distances, whose index represents the location of the feature. Figure 4.5 gives a further explanation in terms of the hardware architecture, and its respective pseudo-code is given by Algorithm 3.

![Figure 4.5: Computing the minimum Hamming distance (kernel 2)](image-url)
Algorithm 3: kernel 2, The minimum Hamming distance

Input: MinHam
Result: LeastHam

1 i= threadId.x;
2 j=threadId.y;
3 k=blockId.x;
4 x=k*blockDim+i;
5 y=k*blockDim+j;
6 $O_{ij} = A_{0y} \oplus B_{xy}$;
7 for $s=blockDim.x/2; s>0; s>>=1$ do
8     if MinHam$_{xy} < MinHam_{xy}+s$ then
9         LeastHam$_{xy}=MinHam_{xy}$
10     else
11         LeastHam$_{xy}=MinHam_{xy}+s$
12 end
13 end

Finally, those distances less than a threshold are discriminated in kernel 3. This threshold is chosen taking into account the suggestions reported in [Calonder et al., 2010], where authors reported that a feature is considered as matched point, as long as its Hamming distance is less than 15% of the length array. Since we are considering ORB descriptors of 256 bits, the threshold is equal to 39. As much as kernel 2, matched points are given by a vector of feature locations. This process is detailed in Figure 4.6 and the pseudo-code of the Algorithm 4.

Figure 4.6: Discriminating those Hamming distance greater than a threshold (kernel 2)
Algorithm 4: kernel 3, Discriminating distances less than a threshold

Input: \(O_{xy}\) Hamming Distances computed in kernel 1

Result: MinHam, MinHamLoc (Minimum Hamming Distances and their respective location)

1. \(t=39\) threshold 15\% of 256 bits;
2. \(i=\text{threadId.x};\)
3. \(j=\text{threadId.y};\)
4. \(k=\text{blockId.x};\)
5. \(x=k*\text{blockDim}+i;\)
6. \(y=k*\text{blockDim}+j;\)
7. \(\text{if } O_{xy} < t \text{ then}\)
8. \(\text{MinHam}_{xy} = O_{xy};\ \text{MinHamLoc} = xy\)
9. \(\text{end}\)

As a final stage, outliers are removed by comparing the distance between the matched points to the stereo baseline, known a priori. This is a simple but effective approach to reducing outliers, which fits well into our scheme, since an outliers reduction process is implicitly involved in the extraction of planar structures.

This section described the proposed parallel model for rapidly matching binary descriptors given a semi-calibrated stereo rig. As mentioned at the beginning of the section, feature matching is a fundamental task for estimating a 3-D point cloud, which is the previous step for High Level Structures extraction. The details for point cloud recovery are given in the next section.

4.3 Point Cloud Recovery

This section describes the process of inferring 3-D information of the matched points, with the aim of generating a point cloud from which planes extraction is carried out. Considering processing time constraints, in contrast to typical stereo algorithms, depth is inferred from semi-calibrated stereo images, in order to avoid a rectification stage, which means that images are unaligned. Thus two main challenges are involved in this stage:

- Distortion removal without considering epipole geometry.
- Depth computation from only extrinsic parameters.
As mentioned, several tasks are considered in the full stereo calibration process such as: obtaining calibration parameters of each stereo camera, computing the stereo parameters of the stereo system based on its geometry, and re-sampling stereo images such that epipolar lines correspond to image rows. This process enables to compute depth from disparity.

Semi-calibration, on the contrary, solely considers extrinsic stereo parameters, this is: baseline, rotation and translation matrices. In addition to these parameters, internal parameters of each stereo camera are considered by this approach.

The process for recovering a 3-D point cloud from a semi-calibrated stereo rig is breaking down in two stages: 1) distortion removal for each matched point, 2) Triangulation.

Typically, a depth map is obtained by following the process illustrated in Figure 4.7. Note that first stereo images are rectified, this process includes removing distortion for all keypoints and their alignment along the epipolar line. Then, these keypoints are matched considering a coplanar stereo configuration. Finally, depth information of each matched point is obtained from disparity. As noted, this process is computationally heavy since it involves removing distortion for every keypoint, even when solely those matched points are considered for forming the point cloud.

**Figure 4.7:** Typical process for depth estimation. 1. Distortion removal is performed for all keypoints. 2. Keypoints are matched considering a coplanar stereo rig. 3. Depth estimation is computed by disparity.
4.3. POINT CLOUD RECOVERY

In contrast to the above, with the aim of reducing the computational burden, the distortion removal is carried out by Tordoff’s method [Tordoff and Murray, 2004] only on those keypoints detected with the FAST detector and whose match was found, this is, keypoints on left image whose correspondence was found on right image, such as is depicted in Figure 4.8.

Given a set of un-distorted matched points, the final step consists in estimating their 3-dimensional perspective, by triangulation. In section 3.2.3 three triangulation methods were explained and explored to perform this stage: DLT (Direct Lineal Transformation), Iterative DLT and Line Intersection. The latter method, proved to be faster to compute while exhibiting a good performance in terms of depth accuracy. This is clearly seen in the performed experiments shown in Chapter 5. Hence line intersection is implemented to generate the 3-D point cloud of the world. For the sake of illustration, Figure 4.8 illustrates the overall process.

So far, the sections have focused on generating an accurate 3-dimensional representation of the world by means of a point cloud. This set of 3-D points is the sample data from which plane extraction is carried out. This process is the final stage of the proposed approach and it is detailed in the next section.

Figure 4.8: Proposed process for depth estimation. 1. Keypoints are matched considering un-rectified images. 2. Distortion removal is carried out only for matched keypoints. 3. Depth is computed by triangulation using line intersection method.
4.4 Plane Extraction

In previous sections, the process for generating a point cloud from a pair of semi-calibrated stereo images was detailed. Summarizing, this process makes use of binary descriptors, after these are matched by using a parallel computation model for acceleration, finally depth information for every matched point was computed. The final stage consists in extracting planes from the generated point cloud.

The task of extracting planes from a point cloud has been widely studied. Generically, this problem is known as *extraction of higher level entities*. This problem is mainly tackled by two approaches: Hough Transform [Ballard, 1981] and Random Sample Consensus [Schnabel et al., 2007]. In general, these methods were proposed with the purpose of detecting parametrized objects in two and three dimensions.

The Hough Transform, was introduced as a method of detecting complex patterns of points in an image. This method assumes the parameters which characterize an object are known. Accordingly, HT determines those values related to the parameters of the model by identifying sets of colinear points in an image, and then establishes a relationship between these points and the parameter point. Several alternatives based on this approach have been proposed in the literature, most of them are described in [Illingworth and Kittler, 1988]. More recently, this approach has been extrapolated to identify 3-D models, such as those works reported in [Knopp et al., 2010, Tarsha-Kurdi et al., 2007, Borrmann et al., 2011].

Even though HT is a robust method, it depends on complex operations, making this method unsuitable for real-time applications. On the other hand, RANSAC is a reliable alternative to HT which offers less complex operations. RANSAC is a randomized procedure that iteratively finds an accurate model for data that may contain a large number of outliers, such as the point cloud previously generated.

Before detailing the RANSAC procedure, plane equation is described. Equation 4.6 describes a planar model in 3-D space, where *a*, *b*, *c* refers to normal vector. The process for computing plane equation given three points *P*, *Q*, *R*, consist of the following stages:

\[
a(x - x_0) + b(y - y_0) + c(z - z_0) = 0
\]  
(4.6)
4.4. PLANE EXTRACTION

1. Two vectors on plane are computed, $\overrightarrow{PQ}$ and $\overrightarrow{PR}$.

$$P(1,0,2), Q(-1,1,2), R(5,0,3)$$

$$\overrightarrow{PQ} = \langle -2,1,0 \rangle$$

$$\overrightarrow{PR} = \langle 4,0,1 \rangle$$

2. Vector normal is computed by cross product between vectors $\overrightarrow{PQ}, \overrightarrow{PR}$.

$$\begin{vmatrix}
  i & j & k \\
 -2 & 1 & 0 \\
 4 & 0 & 1
\end{vmatrix} = \begin{vmatrix}
  1 & 0 & -2 \\
  0 & 1 & 4 \\
  1 & 0 & 4
\end{vmatrix} = i \cdot \begin{vmatrix}
  1 & 0 \\
  0 & 1
\end{vmatrix} - j \cdot \begin{vmatrix}
  -2 & 1 \\
  4 & 0
\end{vmatrix} + k \cdot \begin{vmatrix}
  1 & 0 \\
  4 & 0
\end{vmatrix} = i + 2j - 4k$$

3. Finally, normal vector and the point $P$ is substituted in 4.6, resulting.

$$x + 2y - 4z + 7 = 0$$

Algorithm 1 describes the full process of RANSAC for plane fitting, as pseudo-code. Firstly, the minimum number of points required to estimate the parameters of plane equation is defined. Secondly, the number of iterations is computed by means of equation 4.7 where $P$ is the probability of finding a correct solution, $w$ is the probability that a point belongs to the mathematical model (plane), and $n$ is the number computed in above step. Accordingly, three random points are selected and the plane equation is computed. After that, the distance between a random point and the plane is calculated, and those points less than a threshold are stored. Finally, this process is iterated $n$ times according to Equation 4.7.

$$\text{iterations}_{\text{min}} = \lceil \frac{\log(1 - P)}{\log(1 - w^n)} \rceil$$ (4.7)
Algorithm 5: RANSAC for planes extraction

1. MinP = 3 - minimum points to estimate a plane;
2. IT < minimum number of iterations;
3. count=0;
4. BestSolution=0;
5. Candidates=0;
6. P \triangleright Sample Points (Point Cloud);
7. while count<IT do
   8. Consensus = MinP random Points and compute plane equation parameters;
   9. Candidates = P-Consensus;
   10. while Candidates > 0 do
       11. rPoint = random Point from Candidates;
       12. if distance(rPoint, model parameters)<THRESHOLD then
           13. Consensus = Consensus + rPoint
       end
       14. Candidates= Candidates - rPoint;
   end
   15. if size(Consensus)> size(BestSolution) then
       16. BestSolution=Consensus
   end
   17. count++;
end

Finally, in order to extract several planes of a scene, the following process is carried out: first, the predominant plane is extracted, then, the points belonging to the fundamental plane are removed from the point cloud. Finally, this process is iteratively executed until sample points are 30% of the point cloud. In order to clarify the developed method, the overall process is illustrated in the following Figures. Figure 4.9 (a) illustrates the first two iterations, where red points refer to the minimum points to fit a hypothetical plane, and the points associated to the hypothetical plane are represented in green. Figure 4.9 (b) shows the third iteration and the best solution chosen by RANSAC, which represents the predominant plane. After computing the fundamental solution, their associated points are removed, and RANSAC procedure is used to compute the secondary plane, as is shown in Figure 4.10. As a final step, Convex Hull algorithm [Graham, 1972] is used to extract the plane shape, such as is depicted in Figure 4.11.
4.4. PLANE EXTRACTION

(a) Two first RANSAC iterations

(b) Plane extraction

Figure 4.9: Extraction of the predominant plane using RANSAC.
(a) Two first RANSAC iterations after removing the predominant plane

Figure 4.10: Extraction of the secondary plane using RANSAC.
4.5 Summary

This chapter provided an explanation of the entire process for extracting High Level Structures in form of planes using a semi-calibrated stereo rig. Summarizing, fast binary descriptors are extracted from each stereo image. Then these features were matched following a novel parallel approach implemented in a GPU. After that, the distortion of matched points was removed and their depth information was computed by triangulation, thus generating a point cloud. Finally from this set of 3-D points, RANSAC was implemented for plane extraction. Note that all these algorithmic stages were outlined considering to fulfill real-time constraints.

The next chapter aims at showing the potential of the proposed approach for plane extraction by assessing the performance of the proposed method in terms of depth accuracy, planes quality, and processing time. These evaluation elements are related to the requests of the application context.
Chapter 5

Experiments and Results

This chapter presents the experiments developed to assess the proposed scheme of planes extraction. As mentioned at the beginning of the document, camera calibration plays an important role in the system performance, having a direct influence in depth accuracy. Hence, before giving the details of planes extraction assessment, an evaluation of the semi-calibration parameters is provided.

Accordingly, the proposed scheme is evaluated in terms of depth recovery accuracy, quality of extracted planes and processing time, in indoor and outdoor environments. The above elements were chosen based on the constraints involved by application context. At the end of the chapter, an augmented reality application that exploits the planes extracted with the proposed approach is described, in order to demonstrate the capability of the proposed scheme.

All experiments were performed on the following systems 1) Intel Core i5-2310 at 2.90GHz, 4GB RAM, GPU GeForce GT740, 2) Intel Core i7-4720 at 2.06 GHz, GPU GeForce GTX 970M, 16GB RAM. A stereo camera with a pair of uEye LE cameras with 10cm baseline and 752x480 image resolution. Regarding algorithmic assumptions, a maximum of 2000 features for each stereo image pair are considered for feature detection process.

5.1 Evaluation of the Semi-Calibrated Stereo Rig

As mentioned, the performance of applications focused on generating a 3-D world perspective, highly depends on stereo calibration parameters. Since only orientation and translation parameters are used by the proposed scheme to infer a three-dimensional perspective, the accuracy of depth
data is compromised. Hence, an evaluation of the semi-calibrated stereo is given in this section.

Calibration parameters assessment is carried out by estimating the distance between the calibration board and the stereo camera for 320 poses. This estimation is computed by using the calibration process and compared to that information obtained by Vicon tracker system. Figure 5.1 depicts the experimental setup.

\begin{center}
\includegraphics[width=0.5\textwidth]{setup.png}
\end{center}

\textbf{Figure 5.1:} Experimental setup for the semi-calibrated stereo assessment. Blue points over the calibration board represent the vicon markers.

Extrinsic parameters of stereo calibration define location and orientation of the camera with respect to the world frame, in other words, they refer to orientation and translation matrices. According to the literature, these parameters are obtained from the fundamental matrix (F) by using different optimization methods, for instance: Singular Value Decomposition (SVD) and by means of Intrinsic Parameters (KF), both were explored in this research.

Table 5.1 shows the mean distance and its respective standard deviation for all the poses using two optimization methods, where SVD is lightly better since the mean distance obtained is closer to ground truth. Therefore, extrinsic parameters are computed from the fundamental matrix using SVD.

\begin{center}
\begin{tabular}{|c|c|c|}
\hline
Method & Mean Distance (cm) & Std. Dev. (cm) \\
\hline
Vicon & 212.42 & 61.12 \\
SVD & 182.47 & 47.98 \\
KF & 179.82 & 46.13 \\
\hline
\end{tabular}
\end{center}

\textbf{Table 5.1:} Mean distance (over 320 poses) of the calibration board with respect to stereo Camera, using two optimization methods to obtain extrinsic parameters. Information obtained by Vicon is taken as ground truth.
5.2 Depth Evaluation

The last section provided a panorama of the possible depth estimation error, considering that solely extrinsic stereo parameters are known. The obtained results point out an error of $\pm 10\text{cm}$, this value may be tolerated since the proposed approach does not aim at providing a system capable of computing high precision depth. On the contrary, this approach has the purpose of extracting High Level Structures from a 3-D point cloud while fulfilling real-time constraints. Nonetheless, depth accuracy is vital in this research, as the application context is oriented to localization/navigation and augmented reality.

This experiment aims at obtaining the depth recovery boundaries or in other words the minimum and maximum distance where the depth of the plane extracted is computed correctly. Hence, this experiment consists in computing the depth mean of the points set associated to the plane extracted. This measure is obtained for 26 poses, starting at 50 cm and up to 600 cm, considering 25 samples per pose. Figure 5.2 shows the experimental setup.

![Experimental setup of depth evaluation. Blue points over the plane represent the vicon markers.](image)

The ground truth is obtained using the VICON tracker system. Three Vicon cameras are set up to compute the distance between the plane and the origin of the vicon coordinate system, which is located at the stereo rig position. Since vicon tracker system reports the distance of each maker (See Fig 5.2), the mean over all marker distances is considered as the ground truth value.

The extraction of planes is carried out by means of three triangulation methods, DLT, Iterative DLT, and line intersection and by using the extrinsic stereo parameters obtained from two optimization methods, SVD and KF. Corresponding results are shown in Figure 5.3 graphs, where line intersection method blending with SVD optimization, reports more accurate results. Note that from the pose 19 or from 300cm, line intersection curve moves away from the ground
truth, thus indicating the maximum depth boundary. For the sake of clarity, depth results until the pose 19, using line intersection triangulation and SVD optimization, are showed in Figure 5.4 graph as well as in Table 5.2.

![Figure 5.3: Depth estimation results using three triangulation methods: DLT, Iterative DLT and line intersection. 26 poses are considering for distances from 55 cm to 600 cm](image)

![Figure 5.4: Depth estimation results based on line intersection for 19 poses, starting at 55 cm up to 300 cm](image)

<table>
<thead>
<tr>
<th>Triangulation Method</th>
<th>Error mean (cm)</th>
<th>Std. Dev. (cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Intersection</td>
<td>10.097</td>
<td>3.503</td>
</tr>
<tr>
<td>DLT</td>
<td>10.181</td>
<td>6.440</td>
</tr>
<tr>
<td>Iterative DLT</td>
<td>10.447</td>
<td>6.190</td>
</tr>
</tbody>
</table>

**Table 5.2:** Mean error per triangulation method, based on results shown in Figure 5.4.
5.2. DEPTH EVALUATION

For the sake of comparison, this experiment is repeated but using the following depth sensors: Xtion and guidance. The former make use of structured light to generate a dense point cloud. The guidance, is a combination of active and passive sensors. This means that depth information is obtained from a stereo camera and ultrasonic sensors. The obtained results are compared to our approach using linear intersection method, these are depicted in Figure 5.5.

![Figure 5.5: Comparison between commercial sensors and the proposed system, in terms of depth accuracy.](image)

Figure 5.5: Comparison between commercial sensors and the proposed system, in terms of depth accuracy.

In order to evaluate depth accuracy in outdoor environments, this experiment is repeated but just taking into consideration one triangulation method: line intersection. Similar to the indoor scene, plane depth is estimated for 5 poses from 100cm to 300cm, and considering 25 samples per pose. Instead of moving the plane with respect to the stereo camera, in this case the plane is a wall, so the camera is moved with respect to the plane, such as is shown in Figure 5.6. Distances from plane to each pose are measured by hand, in order to generate the ground truth. The graph showed in Figure 5.7 shows the results. Note that mean error is increased by 2cm compared to indoor experiment.

This is evaluation allowed to choose a reliable triangulation method under real world scenarios. In addition this assessment helps to establish a point of comparison between the proposed system and commercial depth cameras.
5.3 Recovered Area

As mentioned in related work section, planes extraction has been leveraged to perform applications in the arena of image reconstruction. In this regard, an indispensable factor to consider is the quality of planes extracted, or in other words, the percentage of recovered area given a labeled plane.

Motivated by the above, a second experiment was carried out in order to assess coverage percentages of recovered areas from extracted planes, considering their corresponding key points.
5.3. RECOVERED AREA

sets. Thus, an indoor environment with different planes at different positions and orientations was set up. These planes were manually labeled and associated key points were determined as a ground truth. Figure 5.8 depicts the different scenarios, and the planes extracted along with their representation in 3-D world.

A predetermined number of scenes samples considering from one up to five planes were acquired. First, 25 samples of a single plane scenario are captured, by applying the proposed approach for high level structure extraction, corresponding point clouds are compared to the ground truth. The same procedure is repeated for remaining scenarios. Table 5.3 shows the average percentage of recovered areas for every plane extracted considering the number of samples.

<table>
<thead>
<tr>
<th></th>
<th>Plane 1 % recovered</th>
<th>Plane 2 % recovered</th>
<th>Plane 3 % recovered</th>
<th>Plane 4 % recovered</th>
<th>Plane 5 % recovered</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene 1</td>
<td>85.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 2</td>
<td>23.4</td>
<td>58.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 3</td>
<td>35.6</td>
<td>39</td>
<td>71.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 4</td>
<td>25.6</td>
<td>24.2</td>
<td>48.9</td>
<td>52.2</td>
<td></td>
</tr>
<tr>
<td>Scene 5</td>
<td>22.6</td>
<td>26</td>
<td>52</td>
<td>63.4</td>
<td>72.2</td>
</tr>
</tbody>
</table>

Table 5.3: Average percent of area recovered in terms of the key points associated to a plane, using Line Intersection triangulation method.

The same experiment is followed for outdoor environments. Input images are captured at typical buildings, considering solar and non-solar illumination. Unlike indoors, different textured structures were chosen. Obtained results are depicted in Figure 5.10. Note that performance in
terms of area recovered is notably affected by solar illumination. In those cases with either little or much sunlight, the area recovered is highly reduced.

Table 5.4 shows the percentages of recovered area. From results, it is important to point out that the proposed approach struggles in twofold: 1) scenarios with lack or abundance of sunlight, 2) in those cases where the plane is not defined enough, for example the scene in Figure 5.9, which exhibits a dispersed point cloud.

![Figure 5.9](image)

**Figure 5.9:** Example where the proposed approach struggles to extract planes in an outdoor environment.

<table>
<thead>
<tr>
<th>% recovered</th>
<th>Scene 1</th>
<th>Scene 2</th>
<th>Scene 3</th>
<th>Scene 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>69</td>
<td>71</td>
<td>65</td>
<td></td>
</tr>
</tbody>
</table>

**Table 5.4:** Average percent of area recovered in terms of the key points associated to a plane, using Line Intersection triangulation method in outdoor environments.

![Figure 5.10](image)

**Figure 5.10:** Plane extraction for different outdoor scenes. Top: extracted planes. Bottom: Top-view 3-D representation
5.4 Process Time

As mentioned throughout the document, one of the main contributions of this research is the development of a computer vision system capable of extracting planes in real-time on the basis of stereo vision. Given the above, the proposed approach is assessed in terms of processing time, considering two computer systems, in order to report the achieved frame rate and measuring the scalability of the parallel model.

The proposed approach was implemented and assessed in terms of processing time on the following platforms: 1) Intel Core i5-2310 at 2.90GHz, 4GB RAM, GPU GeForce GT740, 2) Intel Core i7-4720 at 2.06 GHz, GPU GeForce GTX 970M, 16GB RAM. This experiment aims at providing an approximation of the frame rates achieved by the proposed framework on different processing platforms and also to show the scalability of the proposed parallel computational model. This means that the proposed framework can be implemented on a larger GPU, where more CUDA cores were available and the performance would show higher frame rates. Table 5.5 shows processing time per framework stage in milliseconds.

<table>
<thead>
<tr>
<th>Stage</th>
<th>GeForce GT740 (384 Cores)</th>
<th>GeForce GTX970M (1280 Cores)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Descriptor extraction</td>
<td>10.2</td>
<td>5.02</td>
</tr>
<tr>
<td>Feature Matching</td>
<td>21.6</td>
<td>13.5</td>
</tr>
<tr>
<td>Outliers reduction</td>
<td>0.3</td>
<td>0.1</td>
</tr>
<tr>
<td>Depth Estimation</td>
<td>0.4</td>
<td>0.1</td>
</tr>
<tr>
<td>Plane Extraction</td>
<td>3.2</td>
<td>2.1</td>
</tr>
<tr>
<td>Total time [ms]</td>
<td>35.8</td>
<td>20.8</td>
</tr>
<tr>
<td>Frame Rate</td>
<td>29fps</td>
<td>50fps</td>
</tr>
</tbody>
</table>

Table 5.5: Processing time in milliseconds of each framework stage, implemented in two different GPU’s.

Given the environment setup (Fig.5.8) 1800 features were detected on average.

5.5 Augmented Reality Example

For the sake of illustration, a derivative application of the proposed framework has already been published, and it is described in this subsection. In this example, the context of a user with augmented reality glasses is emulated, which allows interacting with the surrounding world by
projecting films, videos, social media walls, etc, on any planar structure. Therefore, the dominant plane of an indoor scene is extracted using the proposed framework, and a video is projected on the plane. In such a way that the video perspective changes according to the fitted plane.

A series of snapshots of the application with different perspectives are depicted in Figure 5.11. The extracted dominant plane is shown in Figure 5.11 (a) and (d), and Figures 5.11 (b) and 5.11(e) show the video projected over the plane and its 3D view, respectively. An accompanying video shows the entire example and the framework performance.

![Snapshots of the application with different perspectives](image)

(a) Plane extracted  
(b) Video projected  
(c) 3D-perspective

(d) Plane extracted  
(e) Video projected  
(f) 3D-perspective

**Figure 5.11:** Snapshots of the augmented reality example performed. The fundamental plane extracted, the video projection over the plane and its 3-D perspective with respect to the stereo system are shown respectively.

### 5.6 Analysis Results

The semi-calibration stereo assessment gives a preamble of its impact on the performance of the proposed approach for planes extraction. In specific, it affects the depth recovery. However, the evaluation yields an error less than 15cm. Since the proposed approach is not focusing on improving depth accuracy, this error may be handled given the application context: navigation and augmented reality.

The second experiment confirms the impact of the semi-calibration in depth recovery. The depth boundaries point out that this approach is well-suited for applications whose sensor is in constant motion (e.g., a stereo camera assembled on a mobile robot). In addition, according to the

---

1 video available at: https://drive.google.com/open?id=0B1vox39KUHSmOV9PQWJQLQVdhSkk
5.6. ANALYSIS RESULTS

comparison with commercial sensors, it may say the proposed stereo system competes with these sensors in terms of depth estimation. However, a larger field of view is obtained by commercial sensors due to they make use of active strategies such as infrared light or a wide-range ultrasonic sensors.

The plane extraction is evaluated in terms of the percentage of recovered area. As well as the above experiments, this evaluation was performed in indoors and outdoors scenes. In this regard, this was the only experiment where the environment conditions had an important influence over the results. In indoors, for example, the proposed system recovers more than 50% of the planes. However in outdoors, the recovered percentage is around 40%. This is because of illumination conditions, which decreases the performance. Nevertheless, based on the obtained results, the proposed system is suitable to be implemented in applications with environment conditions constraints.

From the carried out experiments in outdoors, raise concern about how the illumination condition was addressed. In this sense, the camera model played an indispensable role, as we make use of its respective skd, which enables to set the stereo camera up, by modifying parameters such as pixel clock, frame rate, trigger, exposure, etc. The latter help us to automatically focus the image according to illumination intensities.

Finally, processing time results make the proposed scheme an appealing approach for those applications with real-time constraints. Moreover, the achieved frame rate indicates the proposed approach is suitable to be implemented with cameras whose frame rate capture is up to 50fps.

The obtained results pointed out that outlined objectives were accomplished since the proposed approach for planes extraction offers two main qualities: real-time operation and robustness to environment conditions (indoor/outdoor). In addition, the experiment results give an overview of the future research opportunities.
Chapter 6

Conclusions and Future Work

A novel approach for real-time High Level Structures extraction using a semi-calibrated stereo was presented in this research. Several challenges are overcome in this study, mainly, real-time operation and robustness to different environment conditions such as indoors and outdoors. On one hand, real-time constraint is fulfilled by combining fast visual descriptors and parallel computing, besides a rectification process in the frame-to-frame basis is avoided, on contrary to classic stereo techniques. On the other hand, passive stereo techniques are used in order to make a computer vision system able to operate in indoors and outdoors. Thus, the main contributions of this work are twofold: i) the development of a framework which operates with a stereo rig, where only the rotation and translation parameters between the two cameras are known; ii) the design of a parallel scheme for the feature matching stage, which enables to achieve a real-time performance of 50 fps for the whole process, this is, from feature extraction to plane extraction.

Several algorithmic stages were considered in this study. First of all, in contrast to the typical stereo approaches, binary descriptors were explored for this research as they are computationally cheap while exhibiting robustness to affine transformations. In specific, the ORB descriptor as the main core for point cloud estimation. This fast descriptor offered two main qualities to the proposed work: robustness to orientation and the most important it enabled a fast parallel design for feature matching process.

Feature matching, described by the literature as a complex problem due to it requires an exhaustive process, was tackled by a parallel scheme implemented on a GPU, on the basis of CUDA programming. Considering the fact that the quantity of false positives is reduced by comparing all features, a parallel model for brute force matching was proposed in this research. This imple-
CHAPTER 6. CONCLUSIONS AND FUTURE WORK

mentation helped to reduce the computational burden, thus overcoming the bottleneck given by correspondence problem. In addition, the proposed parallel scheme proved to be scalable, meaning that the more cuda cores used, the more real-time performance achieved. This quality is crucial, considering the increment of transistors on a chip, dictated by Moore’s Law.

As opposed to classical stereo techniques, depth information was inferred from a pair of semi-calibrated images. Which means that stereo images are un-rectified but extrinsic parameters (rotation and translation) are known. Thus, a rectification step in a frame-to-frame basis is avoided, providing a vision system able to estimate an accurate depth map independently of epipolar geometry. This approach may be leveraged for computing depth from a series of images captured by a monocular camera, where only displacement information is known.

The latter steps are oriented to rapidly estimate a 3-D perspective of the world, which operates in both indoor and outdoor environments. This study go further, so the point cloud generated is leveraged to extract high level structures from the world. In such a way that the final step consist of RANSAC implementation. This parametrized algorithm is configured to fitting plane models. Accordingly, the system outcome is a representation of the world in form of planes, that can be instantiated to perform real-time applications in the arena of robotics or augmented reality.

The proposed approach was assessed in terms of processing time, depth accuracy and plane extraction. These elements were considered attending to the context application demands. Processing time evaluation was carried out for every single step, and then for the entire process. Results pointed out that, even feature matching process was accelerated via a parallel scheme, it still represents a bottleneck specially when it comes to High Definition images. However, for standard resolution, the proposed approach achieves a frame rate up to 50fps. meaning that is well-suited for real-time systems.

On the other hand, depth accuracy evaluation is vital to obtain the minimum and maximum operation values, in terms of distance. The obtained results are comparable to commercial sensors, however, the proposed system is robust to different environments. Finally, the plane extraction is evaluated in terms of the percentage of recovered area. Unlike the above experiments, the obtained results are affected by the environment, which means that a major recovered area is obtained in indoors than outdoors. Even so, it is enough to perform activities such as obstacle avoidance.

The main goal of this research is attained since the proposed system is able to extract High Level Structures in form of planes, in indoors and outdoors environments. Moreover, hardware
6.1. LIMITATIONS

platforms, GPU’s in specific, are leveraged to accelerate critical stages such as feature matching, thus achieving real-time performance. Regarding specific objectives, a three-dimensional point cloud of the world is obtained on the basis of a semi-calibrated stereo rig, from which planes are extracted. The final objective, in regards to processing time, is accomplished and even overtaken as the frame rate achieved is around 50fps. The obtained results confirm the proposed approach is suitable to be the basic component of those real-time systems oriented to solve problems regarding navigation, representation of the world, localization, among others.

Finally, due to this research started with the exploration of fast visual descriptors, an FPGA-based architecture was proposed in [de Lima et al., 2015] in order to accelerate the construction of BRIEF descriptor. Motivated by the obtained results, this architecture was adjusted for accelerating the construction stage of the ORB descriptor. The latter approach is detailed in Appendix A.

6.1 Limitations

The work limitations are threefold. i) The system highly depends on textured environments. ii) Processing time for feature matching increases with high definition images. iii) Un-robustness to not well-defined planes in the scene.

The first limitation, is related to feature extraction, since this step requires images with a scattered histogram, or in other words, images with textured surfaces where corners are intuitively calculated. The second limitation is provided by the parallel model design. As mentioned, cache memories are exploited to reduce the time cost to access data. However, as these memories are of small size, if cache memory size is exceeded by features size, then global memory is used to store the descriptors, thus increasing the processing cost of data transference. Finally, the last limitation is commonly found in outdoor environments, in specific wooded areas.

6.2 Future Work

Future work includes exploring other local features such as A-KAZE, or implementing segmentation algorithms independent to texture like superpixels. In order to overcome the feature matching bottleneck, is considered to explore hashing techniques oriented to solve the correspondence problem. Finally, the implementation of geometric-features-based methods, aside from RANSAC, is
considered for improving the plane extraction performance in terms of recovered area.
Appendix A

FPGA-based architecture for ORB descriptors construction.

This architecture was developed based on the proposed work reported in [de Lima et al., 2015]. This latter paper aims at providing a novel alternative for constructing the BRIEF descriptor such that is suitable to be implemented via an FPGA architecture. Given that the ORB descriptor has won its place as a simple but effective binary descriptor, the proposed architecture was modified for accelerating the ORB descriptor construction in such a way that a 256-bit descriptor is computed in 15 clock cycles, a significant acceleration compared to the sequential version which requires 128 clock cycles.

The acceleration of ORB descriptor lies in exploiting the arrangement of the pairwise tests proposed in [de Lima et al., 2015]. Therefore the hardware design focus on providing the data-path to construct the binary descriptor from the proposed binary test distribution. Following the idea of ORB, an orientation component was aggregated to BRIEF architecture. However, by using the arrangement, the possible binary tests are notably reduced, thus increasing noise in the descriptor. Therefore, before implementing the architecture, an algorithmic assessment of the pair test distribution was carried out.

The proposed binary test distribution is tested following the established evaluation method and datasets proposed by [Mikołajczyk and Schmid, 2005]. Basically, it consists in evaluating a feature detector and a descriptor in a feature matching scenario. Thus, a oFAST and steered BRIEF blending with the proposed binary test distribution, for feature detection and description
is defined respectively.

Keeping in mind that the aim of feature matching lies in increasing the number of correct positives while minimizing the number of false positives, [Mikolajczyk and Schmid, 2005] also proposed a metric called Recall - Precision; this criterion is based on the number of correct positives and the number of false positives obtained from an image pair, as follows:

\[
\text{recall} = \frac{\text{number of correct-positives}}{\text{total number of positives}} \quad (A.1)
\]

and

\[
1 - \text{precision} = \frac{\text{number of false-positives}}{\text{number of matches (correct or false)}} \quad (A.2)
\]

These metrics aim at measuring the rate of correct-positives, in such a way that high recall and low 1-precision, indicates a descriptor with high score. Thus, ORB descriptor is adjusted with the proposed binary test distribution and compared to other binary descriptors. The evaluation was performed using the dataset showed in Figure A.1.

Figures A.3 and A.2 depict the obtained results, note that the proposed approach behaves similar to the original ORB. These results indicate that the proposed binary test distribution is well-suited for feature matching and applications.
Figure A.2: (Part 2. Transformations: JPEG compression, illumination changes and rotation.)
Figure A.3: (Part 1. Transformations: view-point changes and blur) Recall vs 1 precision graphs for nearest neighbor strategy. Curves are obtained by tuning a threshold of distance between descriptors. In parenthesis, next to the name of each of the methods, the number of found correspondences is shown.
Once the arrangement was algorithmically assessed, the architecture is described. An overview of the proposed parallel architecture is drawn by a block diagram in Figure A.4, where oFAST points and a smoothed image are stored in a single port and a dual port RAM memory, respectively. The following steps describe the process for calculating ORB descriptors, by steered BRIEF:

1. The first corner location is read by the address control.

2. The memory address of a 4-pixels block (detailed in [de Lima et al., 2015]) is computed by the address generator block.

3. 4-pixels blocks are stored in a buffer.

4. Once the buffer is filled, binary tests are computed in parallel.

**Figure A.4:** FPGA Architecture for the construction of ORB using our approach.

This Appendix presents the proposed approach for accelerating the construction of the ORB descriptor, which is derived from this thesis work. As mentioned, for the sake of accelerating the first stage of the plane extraction method, different hardware-based alternatives were explored such as GPU and FPGA. The latter gave as a result the presented architecture.
APPENDIX A. FPGA-BASED ARCHITECTURE FOR ORB DESCRIPTORS CONSTRUCTION.
Bibliography


