



**Ana Carolina Fonseca  
Matos**

**Desenvolvimento de um sistema de visão estéreo  
com grande linha de base para a identificação de  
peões e outros alvos em estrada**

**Development of a large baseline stereo vision rig for  
pedestrian and other target detection on road**





**Ana Carolina Fonseca  
Matos**

**Desenvolvimento de um sistema de visão estéreo  
com grande linha de base para a identificação de  
peões e outros alvos em estrada**

**Development of a large baseline stereo vision rig for  
pedestrian and other target detection on road**

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Mecânica, realizada sob a orientação científica de Vitor Manuel Ferreira dos Santos, Professor Associado do Departamento de Engenharia Mecânica da Universidade de Aveiro e sob a co-orientação de António José Ribeiro Neves, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática



**O júri / The jury**

Presidente / President

**Prof. Doutor José Paulo Oliveira Santos**

Professor Auxiliar da Universidade de Aveiro

Vogais / Committee

**Doutor Miguel Armando Riem de Oliveira**

Bolseiro Pós-Doutoramento, Inesc - Porto

**Prof. Doutor Vitor Manuel Ferreira dos Santos**

Professor Associado da Universidade de Aveiro (orientador)



## **Agradecimentos / Acknowledgements**

Em primeiro lugar, gostaria de agradecer ao Professor Doutor Vitor Santos pela orientação e por todo o apoio dado na realização deste trabalho. Não menos importante, agradeço também ao Professor Doutor António Neves e ao Jorge Almeida por todos os conhecimentos passados e opiniões dadas.

Aos meus pais, agradeço o facto de me terem sempre motivado e apoiado (financeira e psicologicamente) durante todo o meu curso e em especial durante o caminho que foi a dissertação. Ao meu irmão, um especial obrigada por ter sido o 'peão' deste trabalho e por sempre se ter demonstrado disponível e entusiasmado para me ajudar, como um verdadeiro irmão mais novo. Que um dia também seja Engenheiro!

Aos meus amigos, um obrigada por tudo o que passámos nestes anos de universidade e por todas as memórias que ficam.

Por fim, ao Gonçalo, obrigada pelo apoio, pela compreensão, e por tudo o resto. Ah e não esqueço, obrigada pelo teu computador novo emprestado.





**Palavras-chave**

ADAS, ATLASCAR, Carros Autónomos, Detecção de objetos, Percepção, ROS, OpenCV, Nuvem de Pontos, Baseline, Visão Stereo, Imagem de Disparidade

**Resumo**

Os veículos autónomos são uma tendência cada vez mais crescente nos dias de hoje com os grandes fabricantes da área automóvel, e não só, concentrados em desenvolver carros autónomos. As duas maiores vantagens que se destacam para os carros autónomos são maior conforto para o condutor e maior segurança, onde este trabalho se foca. São incontáveis as vezes que um condutor, por distração ou por outra razão, não vê um objeto na estrada e colide ou um peão na estrada que é atropelado. Esta é uma das questões que um sistema de apoio á condução (ADAS) ou um carro autónomo tenta solucionar e por ser uma questão tão relevante há cada vez mais investigação nesta área. Um dos sistemas mais usados para este tipo de aplicação são câmaras digitais, que fornecem informação muito completa sobre o meio circundante, para além de sistemas como sensores LIDAR, entre outros. Uma tendência que deriva desta é o uso de sistemas stereo, sistemas com duas câmaras, e neste contexto coloca-se uma pergunta à qual este trabalho tenta responder: "qual é a distância ideal entre as câmaras num sistema stereo para deteção de objetos ou peões?". Esta tese apresenta todo o desenvolvimento de um sistema de visão stereo: desde o desenvolvimento de todo o software necessário para calcular a que distância estão peões e objetos usando duas câmaras até ao desenvolvimento de um sistema de fixação das câmaras que permita o estudo da qualidade da deteção de peões para várias baselines. Foram realizadas experiências para estudar a influência da baseline e da distância focal da lente que consistiam em gravar imagens com um peão em deslocamento a distâncias pré definidas e marcadas no chão assim como um objeto fixo, tudo em cenário exterior. A análise dos resultados foi feita comparando o valor calculado automaticamente pela aplicação com o valor medido. Conclui-se que com este sistema e com esta aplicação é possível detetar peões com exatidão razoável. No entanto, os melhores resultados foram obtidos para a baseline de 0.3m e para uma lente de 8mm.



**Keywords**

ADAS, ATLASCAR, Autonomous Cars, Object Detection, Perception, ROS, OpenCV, Point Cloud, Baseline, Stereo Vision, Disparity Image

**Abstract**

Nowadays, autonomous vehicles are an increasing trend as the major players of this sector, and not only, are focused in developing autonomous cars. The two main advantages of autonomous cars are the higher convenience for the passengers and more safety for the passengers and for the people around, which is what this thesis focus on. Sometimes, due to distraction or another reasons, the driver does not see an object on the road and crash or a pedestrian in the cross walk and the person is run over. This is one of the questions that an ADAS or an autonomous car tries to solve and due to the huge relevance of this more research have been done in this area. One of the most applied systems for ADAS are digital cameras, that provide complex information about the surrounding environment, in addition to LIDAR sensor and others. Following this trend, the use of stereo vision systems is increasing - systems with two cameras, and in this context a question comes up: "what is the ideal distance between the cameras in a stereo system for object and pedestrian detection?". This thesis shows all the development of a stereo vision system: from the development of the necessary software for calculating the objects and pedestrians distance from the setup using two cameras, to the design of a fixing system for the cameras that allows the study of stereo for different baselines. In order to study the influence of the baseline and the focal distance a pedestrian, walking through previously marked positions, and a fixed object, were recorded, in an exterior scenario. The results were analyzed by comparing the automatically calculated distance, using the application, with the real value measured. It was concluded, in the end, that the distance of pedestrians and objects can be calculated, with minimal error, using the software developed and the fixing support system. However, the best results were achieved for the 0.3m baseline and for the 8mm lens.



# Contents

<b>Contents</b>	<b>i</b>
<b>List of Figures</b>	<b>iii</b>
<b>List of Tables</b>	<b>v</b>
<b>List of Acronyms</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Objectives . . . . .	2
1.3 Thesis Organization . . . . .	3
<b>2 ADAS and Stereo Vision Systems</b>	<b>5</b>
2.1 ATLAS Project . . . . .	5
2.2 ADAS evolution . . . . .	8
2.3 Computer Vision . . . . .	11
2.4 Pedestrian and other target detection - using computer vision . . . . .	12
2.5 Fundamentals of Stereo Vision . . . . .	14
2.6 Stereo Systems . . . . .	16
2.7 Large baselines for ADAS . . . . .	17
<b>3 Stereo Rig's Development</b>	<b>21</b>
3.1 Cameras . . . . .	21
3.2 Host Adapters . . . . .	23
3.3 Synchronization . . . . .	23
3.4 Fixing and Assembling of the Cameras . . . . .	24
<b>4 Stereo Camera Application</b>	<b>27</b>
4.1 Software . . . . .	27
4.1.1 ROS . . . . .	27
4.1.2 OpenCV . . . . .	28
4.2 Application Development - Connect and Synchronize . . . . .	28
4.3 Stereo Image Processing . . . . .	31
4.3.1 Stereo Correspondence Algorithm . . . . .	31
4.3.2 Calculation of the Disparity Map . . . . .	32
4.4 Object detection . . . . .	34

4.5	ROS Application . . . . .	36
4.6	Stereo Camera Application - Global View . . . . .	38
<b>5</b>	<b>Stereo Camera Calibration</b>	<b>39</b>
5.1	Fundamentals of Stereo Calibration . . . . .	39
5.2	User Instructions for Calibration . . . . .	41
5.3	Results of Calibration Experiments . . . . .	43
<b>6</b>	<b>Experimental Results - Influence of the Baseline and Lens</b>	<b>45</b>
6.1	ADAS Context . . . . .	45
6.2	Theoretical Study . . . . .	46
6.2.1	Baseline and Lens . . . . .	47
6.2.2	Field of View . . . . .	49
6.3	Data Collection . . . . .	51
6.4	Experimental Results . . . . .	53
6.4.1	SGBM Parameters . . . . .	53
6.4.2	Evaluation of the Stereo Vision System . . . . .	56
<b>7</b>	<b>Conclusion and Future Work</b>	<b>59</b>
7.1	Future Work . . . . .	61
	<b>References</b>	<b>62</b>
	<b>Appendix</b>	<b>68</b>

# List of Figures

1.1	ATLASCAR in 2014, a 1998 Ford Escort. . . . .	2
2.1	ATLAS MV. . . . .	5
2.2	ATLAS 2000. . . . .	5
2.3	ATLASCAR in 2015, a 1998 Ford Escort. . . . .	6
2.4	i-MiEV, next ATLAS series [8]. . . . .	7
2.5	Google Self-Driving Car Project in detail [11]. . . . .	8
2.6	Land Rover Discovery Sport equipped with Bosch Stereo Video Camera [20].	10
2.7	Binocular vision in humans, which can be compared to a stereo vision system where the "eyes" of the system are the digital cameras [25]. . . . .	11
2.8	Original image and the corresponding edge maps generated by a sobel filter [27].	12
2.9	Some combined results on images using part-based detection [28]. . . . .	12
2.10	Examples of patch-based detection. (a) initial hypotheses; (b) hypothesized segmentation for correct hypotheses; (c) segmentation for false positives; (d) fitted silhouettes [29]. . . . .	13
2.11	Flow chart of a Motion-based Detection system [31]. . . . .	13
2.12	An example Multiple cameras based detection: (a) the left image of a stereo pair with the resulting depth map; (b) the segmented regions in different colors [32]. . . . .	14
2.13	Epipolar Geometry [34]. On top, OL is the center of the left camera and OR is the center of the right camera. The line OL - OR represents the baseline while the lines XL - eL and XR - eR represents the epipolar lines between the cameras and the object. Notice that the epipolar lines and the baseline are in the same plane that intersects both camera planes. On the bottom it is shown the two points of view from a stereo vision system. . . . .	15
2.14	Epipolar geometry of a stereo camera setup [35]. . . . .	15
2.15	Trinocular System used in TerraMax [16]. . . . .	17
2.16	Representation of the area covered by all sensing systems used in TerraMax [16].	18
2.17	Left: frontal stereo vision systems to detect lane markings and obstacles; right: frontal panoramic vision system to locate the leader vehicle [42]. . . . .	18
2.18	Back stereo system used to detect upcoming traffic and obstacles during backup maneuvers [42]. . . . .	18
2.19	Position and description on each sensor of Annieway [18]. . . . .	19
2.20	Position and description on each sensor of BRAiVE [19]. . . . .	19
3.1	Point Grey Flea3 GigE [43]. . . . .	21

3.2	Flea3 GigE Dimensional Diagram [43]. . . . .	22
3.3	GPIO connector description [43]. . . . .	22
3.4	Point Grey Lenses available for this study [43]. . . . .	23
3.5	Synchronization setup. . . . .	24
3.6	Stereo Camera Fixing and Assemble Project - render. . . . .	24
3.7	Aluminum profile used as rail for the fixing and assemble system. . . . .	25
3.8	Camera skate - assembling to the camera. . . . .	25
3.9	Camera skate - assemble to the rail profile. . . . .	26
3.10	Close up to the fixing and assemble system. . . . .	26
4.1	ROS simple network of processes. . . . .	28
4.2	Trigger Mode 0 ("Standard External Trigger Mode") [47]. . . . .	29
4.3	Frames from a sequence captured using <code>StereoCam</code> driver. . . . .	30
4.4	Stereo Evaluation 2015 [50]. . . . .	31
4.5	Example of two original acquired images, two rectified images and the resulting disparity map, top to bottom. . . . .	33
4.6	Different angles of visualization of the same point cloud, using PCL Visualizer. Circled in green is the closer pedestrian, in blue is the other pedestrian that is between the previous pedestrian and the light pole (circled in orange). . . . .	35
4.7	ROS Stereo Processing example [59]. . . . .	36
4.8	ROS <code>stereo_image_proc</code> diagram [59]. . . . .	37
4.9	ROS Dynamic Reconfigure. . . . .	37
4.10	Stereo Camera Application Diagram. . . . .	38
5.1	New Chessboard for stereo camera calibration. . . . .	40
5.2	Calibration setup - first experiment. . . . .	41
5.3	Examples of some of the frames used to calibrate the cameras (30cm baseline and 8mm lens). . . . .	42
5.4	Example of the calibration process (0.3m baseline and 8mm lenses): Original acquired Images (top); Detect Chessboard Corners (middle); Compute calibration parameters and Rectify images(bottom). . . . .	44
6.1	Simulator of the stopping distance [61]. . . . .	46
6.2	Influence of the distance of the obstacle with depth resolution - 8mm lens. . . . .	47
6.3	Influence of the distance of the obstacle with depth resolution - 16mm lens. . . . .	48
6.4	Influence of the disparity value with obstacle distance - 8mm lens. . . . .	48
6.5	Influence of the disparity value with obstacle distance - 16mm lens. . . . .	49
6.6	Field of View of a Stereo Vision System. . . . .	50
6.7	Data collection environment - open area at Aveiro University. . . . .	51
6.8	Positions for data collection: Red @ 10m, Yellow @ 20m, Green @ 30m, Blue @ 40m, Purple @ 50m. . . . .	52
6.9	Photo montage of the positions for data collection. . . . .	52
6.10	Example of the a disparity map considered for this ADAS system. . . . .	55



# List of Tables

2.1	ADAS evolution. . . . .	9
2.2	Most significant stereo vision cameras on the market. . . . .	16
5.1	Stereo Calibration results for the baselines and lenses under study. . . . .	43
6.1	FOV and Distance covered by the stereo vision system. . . . .	50
6.2	Qualitative analysis of the influence of the SGBM parameter is values. . . . .	53
6.3	SGBM Parameters - 8mm Lens. . . . .	55
6.4	SGBM Parameters - 16mm Lens. . . . .	56
6.5	Result is analysis of the Calculated Distance Vs. Real Distance - 8mm lens. .	57
6.6	Result's analysis of the Calculated Distance Vs. Real Distance - 16mm lens. .	58



# List of Acronyms

- ADAS** Advanced Driving Assisting System
- DARPA** Defense Advanced Research Projects Agency
- INS** Inertial Navigation System
- FPS** Frames Per Second
- GPIO** General Purpose Input/Output
- GPS** Global Positioning System
- LIDAR** Light Detection And Ranging
- PCL** Point Cloud Library
- PWM** Pulse Width Modulation
- ROS** Robot Operating System
- SGBM** Semi Global Block Matching
- VIAC** VisLab Intercontinental Autonomous Challenge



# Chapter 1

## Introduction

According to the 2014 Annual Report of "Autoridade Nacional de Segurança Rodoviária" [1] there were 30.604 road accidents with victims, in Portugal. From these, 4.595 had been run overs that resulted in 145 mortal victims and 414 serious injuries. Another important fact is that the majority of these events occurred in urban scenarios and during daylight. Therefore vehicles equipped with cameras for pedestrian detection (and other targets on road) would possibly decrease, not only the number of accidents but also the number of road accident victims. Because of that, many researchers and major companies focus their studies on Advanced Driver Assistance Systems (ADAS).

One of the main trends in ADAS is the use of stereo cameras as their price has been dropping and the image quality has been increasing.

### 1.1 Motivation

This project focuses on the study of an ideal stereo vision system with a large baseline, defined as the distance between the cameras, to complement the ATLASCAR project.

ATLASCAR, in figure 1.1, is a research project of a fully autonomous vehicle for studies related to ADAS. In the past 5 years, the ATLASCAR has evolved from a completely normal car to a vehicle with some considerable ADAS, that will be presented in section 2.1. However, the ATLASCAR still cannot detect pedestrians and other targets in a consistent way, so there is not, yet, a completely safe ADAS for this application. From a point of view that a stereo vision system could be interesting to detect pedestrians or other targets on ATLASCAR, it is necessary to focus on what would be the ideal setup for this that is where this project contributes to ATLASCAR.

A stereo vision system provides a more complete representation of the real world when compared with other techniques. Moreover, it has the advantage of being a passive sensor, i.e. there is no interference with humans or other sensors as it happens with active sensor like LIDAR. Another important fact that puts stereo cameras in the top of the most interesting 3D sensors is the decreasing of camera's prices. Thanks to the previous advantages of stereo cameras, a stereo system for pedestrians and other targets detection could be easily implemented in common cars.

Detect pedestrians or objects using a stereo vision system can be achieved through the a point cloud, the 3D representation of the world, that is based on a disparity map. The disparity map is created using two images, one from each camera, captured at the same exact

moment from two different points of view. Basically, after capturing both images, they are corrected and then a stereo correspondence algorithm is used for comparing these two rectified images generating the disparity image. For example, an object closer to the camera is represented in the disparity map with higher values than the one that is further away from the system.



Figure 1.1: ATLASCAR in 2014, a 1998 Ford Escort.

Common cameras have short baselines (10-30 cm) for a matter of portability, but, according to some authors, that limits the maximal distance where range can be correctly discriminated, at most up to a few meters. In this case, this ADAS application tries to detect pedestrians or objects at distances up to 50 meters. After a quick search some examples of research projects in ADAS that use large baseline stereo systems can be found, as shown in the next chapter. However, all these projects use different baselines and it is not clear what is the perfect baseline for this concept, and once more this is what this thesis tries to answer.

Using a stereo vision system as an ADAS in every vehicle could prevent multiples running overs each day. For example, the driver could be alerted that a kid is about to cross the road (just by analyzing the posture of the pedestrian in the point cloud), or that a lady is in the crosswalk or even if an object is in the middle of the road. Since ADAS is a context of very high relevance, it justifies a dedicated approach to develop such a rig and install it on cars for studies on safety and driver assistance, including also autonomous driving.

In order to achieve the main goal of studying the perfect baseline for this kind of application the objectives of the project are presented next.

## 1.2 Objectives

The main objectives of this master thesis are:

1. Design a structure to mechanically support and assemble, with fine-tuning possibilities, two high speed gigE cameras;

2. Develop a software application for video capture and synchronization of both sensors;
3. Implement of a calibration system for the cameras;
4. Develop an application to obtain disparity map and point clouds of scenes with high resolution;
5. Study of stereo systems with different baselines and camera lenses;
6. Develop a software application implementing one or more algorithms from the literature to detect pedestrians or other relevant targets, for ADAS.

### 1.3 Thesis Organization

In chapter "ADAS and Stereo Vision Systems" (chapter 2), this study is framed in the AT-LASCAR project and a short state of the art of pedestrian detection and stereo vision systems are presented.

"Stereo Rig's Development" (chapter 3) describes all the hardware required for the project from the equipment used and its specification to the synchronization setup and finally the fixing and assemble system specifically developed for this study.

"Stereo Camera Application" (chapter 4) is the chapter where the developed application and how stereo vision works are explained. This software basically connects two cameras and captures synchronized frames, calculates the disparity map and detect objects, based on the point cloud generated after the disparity image.

"Stereo Camera Calibration" (chapter 5) starts by explaining the Fundamentals of Stereo Calibration. Then describes how to calibrate the stereo system and presents the results of calibration experiments.

"Experimental Results" (chapter 6) is dedicated to the presentation of the results. In the firsts two sections it is presented some ADAS context and some Theoretical Study that support the following experiments. Then, the data collection procedure is explained and finally the results are shown.

Finally, the main "Conclusions and Future Work" (chapter 7) and the Appendix presents some extra information that was considered not to be necessary to the understanding of the project but still important to complement it.





## Chapter 2

# ADAS and Stereo Vision Systems

ADAS are systems that help the driver in the driving process. With these systems a semi-autonomous car can advise the driver in real time or even full-autonomous vehicles may be capable of replacing a human driver. An autonomous car should be able to perform the main tasks required to drive a vehicle such as "follow the road and keep within the correct lane, maintaining a safe distance between vehicles, regulating the vehicle's speed according to environment and road conditions, moving across lanes in order to overtake vehicles and avoid obstacles, helping to find the correct and shortest route to a destination, and the movement and parking within urban environments" [3].

However, autonomous cars (or unmanned-vehicles) can be used for many other applications like agriculture, demeaning, rescue and other dangerous applications.

### 2.1 ATLAS Project

The ATLAS project started in 2003 in the Automation and Robotics Lab at Aveiro University with the main objective of researching on ADAS.

At the beginning the project started by developing robots, such as ATLAS MV (figure 2.1) and ATLAS 2000 (figure 2.2), for autonomous driving challenges. These robots were scale models and used active and passive perception to sense their surroundings.



Figure 2.1: ATLAS MV.

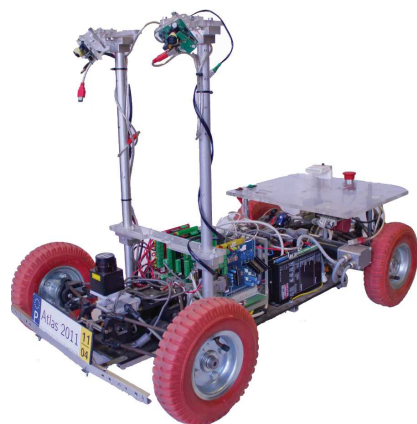


Figure 2.2: ATLAS 2000.

Due to the success of the scale series of the ATLAS project, the research group decided to focus on real road scenarios and the ATLASCAR started its development in 2010. Nowadays ATLASCAR (figure 2.3) is equipped with several sensors in order to perceive the surrounding environment:

- stereo camera;
- foveated Vision;
- 3D Laser, custom developed;
- three 2D lasers;
- Inertial Measurement Unit (IMU);
- GPS;
- low level sensors for car performance monitoring.



Figure 2.3: ATLASCAR in 2015, a 1998 Ford Escort.

All these sensors give enriched data that is then applied in different types of research. To power the sensor equipment, the car has a DC generator propelled by the engine shaft connected to an AC inverted and a UPS (Uninterruptible Power Supply) [4].

In the past 5 years the ATLASCAR has evolved from a completely normal car to a vehicle with some considerable ADAS such as an automatic steering system, an automatic pilot, detection of the navigable space, an automatic gearbox and many others.

Particularly, perception studies in ATLASCAR using vision systems started in 2012 by Tiago Talhada [2]. The main conclusion of this first study on stereo was that it is required a high computational effort making impossible to analyze each scene in real time. However the results were good and proved that stereo cameras are an interesting sensor for ADAS [4].

Projects in ATLASCAR about pedestrians detection started in 2013 by Daniel Coimbra using LIDAR to detect targets [5]. In 2013, Pedro Silva, developed an algorithm for "Virtual Pedestrian Detection using Integral Channels for ADAS" [6] and, in 2014, Rui Azevedo continued his work implementing "Sensor Fusion of Laser and Vision in Active Pedestrian Detection" [7]. Both these two projects conclusions proved that pedestrian detection is more consistent when using cameras, compared with detecting systems that only use lasers. However the algorithm developed by Pedro Silva requires too much processing time for an ADAS application. Rui Azevedo managed to reduce the processing time since the image processing was only applied in the areas considered by the laser as "possible pedestrians". The results were very impressive, but, maybe, the results could be even better if it uses a stereo vision system.

In 2016 the ATLAS project will start the development of a new ATLAS series with an electric car (figure 2.4).



Figure 2.4: i-MiEV, next ATLAS series [8].

## 2.2 ADAS evolution

Initially, researchers only used devices that required low computer processing such as radar or acoustic sensors, but as computers became more advanced, vision sensors started to be used in this field since they provide more rich description of the environment. Today, autonomous vehicles sense their surroundings with multiple techniques such as radar, LIDAR, GPS, and cameras. Advanced control systems interpret sensory information to identify appropriate navigation paths, as well as obstacles and traffic signs. The utilization of autonomous cars would mainly reduce traffic congestion and traffic collisions but also it would alleviate parking scarcity since the car would park by itself, possibly far from the desired destination.

The first real idea of an autonomous car appeared at the 1939 World's Fair and it consisted of an electric vehicle controlled by radio and powered by circuits embedded in the roadway [9]. However, the true evolution of autonomous cars started in the late 1980's thanks to some research groups willing to invest in high-risk studies while the automotive industry only became interested in this field in the late 1990's [10]. Since then, autonomous cars have evolved to a different level, mainly thanks to the DARPA Challenges (Defense Advanced Research Projects Agency) that pushed many researchers to team up and develop driver less cars. The main achievements in this area made by research groups, excluding the major automotive brands, until 2013, are summarized in table 2.1.

However, despite all the development in this area, nowadays fully autonomous cars are still mainly prototypes and demonstration systems that are developed by the some research groups.

The most famous prototype of an autonomous car is the Google Self-Driving Car (figure 2.5). It uses lasers, radars and cameras to detect pedestrians, cars and traffic signs. It is a fully automated vehicle and it is now in the final tests. However, it is not yet on sale and the Google Self-Driving Car Project Website still do not present a date for its commercialization for the public.

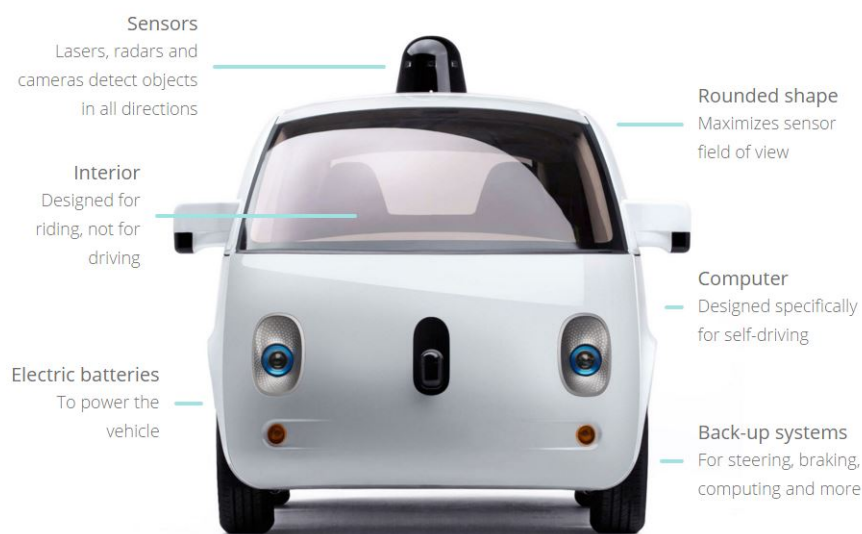














Figure 2.5: Google Self-Driving Car Project in detail [11].

Table 2.1: ADAS evolution.

Year	Project	Achievements	Car
1986	VaMoRs [12]	Tracks road markers on highway	
1994	VaMoRs-P [10]	Road and objects recognition	
1995	Navlab 5 [13]	4000km driving with 95% autonomous steering-wheel control	
1997	ParkShuttle [14]	Driverless public road transport system (Netherlands)	
1998	Argo [2]	Autonomous driving using low cost components	
2005	Stanley [15]	Champion of the 1st DARPA Event (mainly used laser-scanner and GPS)	
2005	Terra Max [16]	Asymmetrical trinocular vision system	
2007	Boss [15]	Champion of the Urban DARPA Challenge (used 2 cameras, 3 LIDAR and 1 Radar)	
2010	Google Driverless Car [17]	First legal autonomous car (the same project leader as Stanley)	
2010	VIAC [10]	Autonomous driving from Italy to China in a 100-day and 15900km journey	
2012	AnnieWay [18]	Real-world computer vision benchmarks for stereo, optical flow, visual odometry, 3D object detection and 3D tracking	
2013	BRAiVE [19]	Autonomous driving in real urban scenario	

After 2013, the automotive industry started to present real ADAS solutions to the main public and nowadays there are some examples of semi-autonomous cars that anyone, with a certain financial possibility, can buy such as:

- Infiniti Q50, released in 2013, is equipped with cameras, radar and other technology to deliver various lane-keeping, collision avoidance and cruise control features.
- 2014 Mercedes S-Class has options for autonomous steering, lane keeping, acceleration/braking, parking, accident avoidance, and driver fatigue detection, in both city traffic and highway speeds of up to 200 km/h.

In 2016 Tesla Motors plans to introduce its AutoPilot system in the Model S electric cars and Apple launched the Titan Project that consists of developing an autonomous car. In June 2015, Bosch advertised a stereo video camera that will be integrated with the braking system of the new Land Rover Discovery Sport (figure 2.6). This emergency braking systems is based solely on camera data and recognizes not only cars but also traffic signs up to 50 meters [20].



Figure 2.6: Land Rover Discovery Sport equipped with Bosch Stereo Video Camera [20].

In 2020, Nissan, GM, Mercedes-Benz, Audi, BMW, Renault, Tesla Motors and Google expect to release autonomous or at least semi-autonomous cars. Due to the secrecy policy of the automotive companies there is not much detailed information about all these projects.

However, despite all the potential advantages of autonomous vehicles and ADAS, there are some limitations related to software reliability and legislation. As it is forbidden to circulate with autonomous cars on most public streets, researchers try to create small driving aids to improve safety but never replacing the human driver [21]. As bigger companies are becoming more aware about the business possibilities in the ADAS research area more pressure is put on countries to allow testing of autonomous cars on public roads. An important achievement in this area was made in February 2016 as USA "vehicle safety regulators have said the artificial intelligence system piloting a self-driving Google car could be considered the driver under federal law" [22].

## 2.3 Computer Vision

”The goal of computer vision is to make useful decisions about real physical objects and scenes based on sensed images” [23].

Computer vision is the field of study responsible for methods for acquiring, processing, analyzing, and understanding images for decision taking. This field of study is important not only for navigation of unmanned vehicles but also for controlling processes (in industrial environment), security surveillance and medical engineering. As cameras and computers continue to become more advanced and less expensive it becomes easier to use this kind of equipment [24].

The majority of our daily activities depends on our ability to use vision. It is mainly due to the binocular vision that humans can perceive the world in 3D (figure 2.7), so it is easily understandable that an autonomous car can also use computer vision and stereo vision systems.

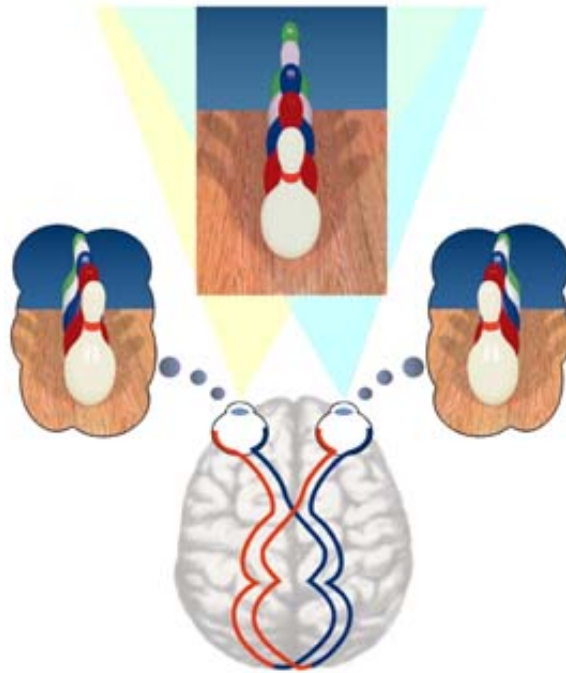


Figure 2.7: Binocular vision in humans, which can be compared to a stereo vision system where the ”eyes” of the system are the digital cameras [25].

Although others sensors, like radars, LIDAR, GPS, INS or ultrasonic sensors, can be very useful in autonomous driving they do not provide enough information. Another major problem of these sensors is the fact that they are active sensors, i.e. they send signals, light or electrons to the environment what can cause interference. By using vision systems (monocular or stereo) this problem does not exist because vision systems are passive sensors, i.e. they simply use the natural illumination of the scene to capture the surroundings, so the data acquisition is not invasive and does not alter the environment. However, in certain conditions like fog weather, night or direct sun, vision sensors are less robust. Another disadvantage of vision devices is the high computing resources required when compared to other sensors.

## 2.4 Pedestrian and other target detection - using computer vision

The detection of pedestrians or other obstacles on the road is a step forward on improving safety and to prevent accidents. However, pedestrian and target detection has another important application: intelligent video surveillance systems. Both these areas contribute to keep the pedestrian and other target detection as an active research area in computer vision.

Next, 5 widely used techniques for the detection of obstacles or pedestrians, using computer vision, are presented:

- **Holistic detection:** the whole frame is scanned by detectors trained to search if the images features inside the local search window match a certain criteria, if so a pedestrian is found. Some methods use global features like edge template (figure 2.8) while other apply local features such as histogram of oriented gradients (HOG) [26]. This technique is easily affected by background clutter and obstructions.



Figure 2.8: Original image and the corresponding edge maps generated by a sobel filter [27].

- **Part-based detection:** the human body is modeled as a collection of parts. First, a dense sample image pyramid is build in order to detect body parts at different scales. Then the guessed body parts are firstly generated by searching edgelets (short segments of lines or curves) and orientation features. These part hypotheses are then classified and joined to form the best assembly of existing pedestrian suggestions, generating the final set of bounding boxes (figure 2.9). The part detection is a difficult task [28].



Figure 2.9: Some combined results on images using part-based detection [28].



- Patch-based detection:** this approach combines detection and segmentation with Implicit Shape Model (ISM). At first, a code book of local appearance is learned during a short training process. Then, in the detecting process, extracted local features are used to match against the code book entries. The final detection results correspond to the local features that did not match the code book (figure 2.10) [29].

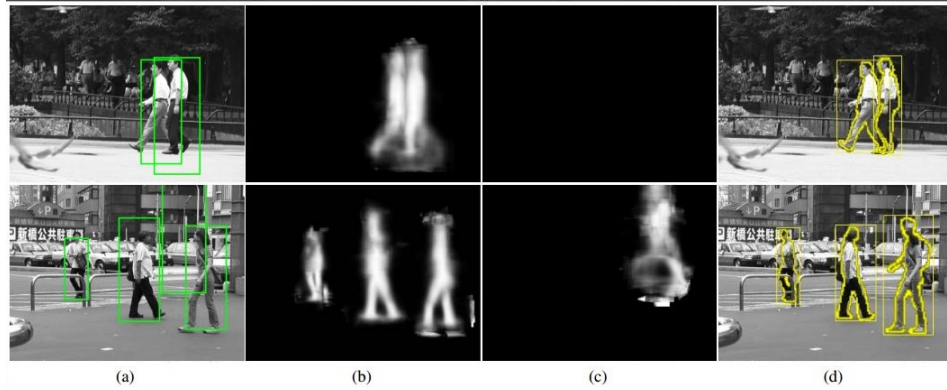


Figure 2.10: Examples of patch-based detection. (a) initial hypotheses; (b) hypothesized segmentation for correct hypotheses; (c) segmentation for false positives; (d) fitted silhouettes [29].

- Motion-based detection:** basically, this technique classifies the pixels of video streams as either background, where no motion is detected, or foreground, where motion is detected (figure 2.11). This procedure subtracts to the background the silhouettes (the connected components in the foreground) of every moving element in the scene detecting pedestrians. This method requires a fixed camera and stationary lighting conditions [30].

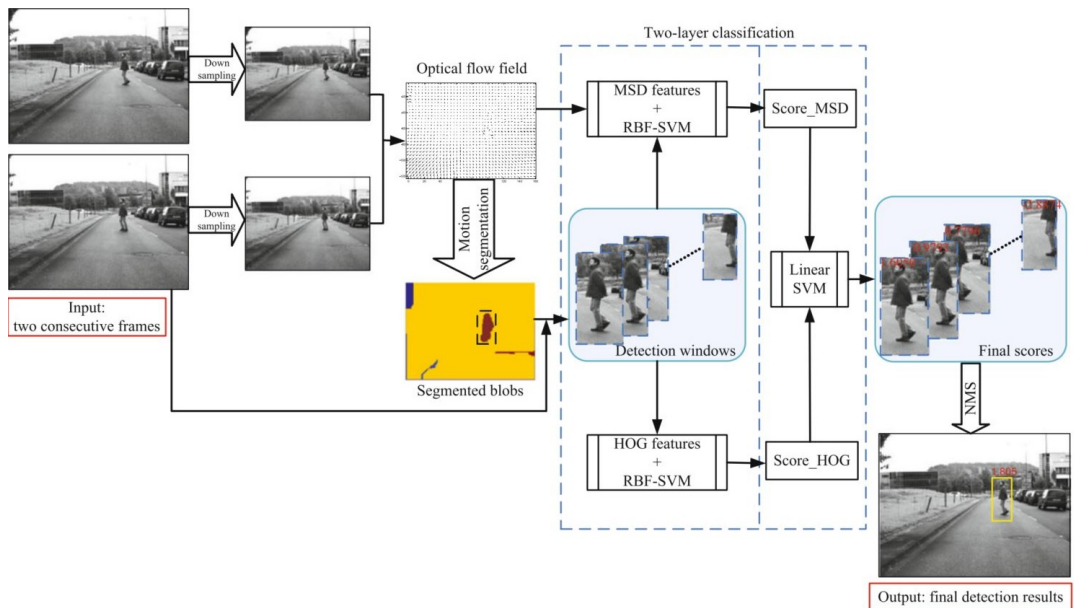


Figure 2.11: Flow chart of a Motion-based Detection system [31].

- **Multiple cameras detection:** First, synchronized images are taken from a pair of cameras. Then a region-of-interest (ROI) is detected, projecting stereo data into a polar-perspective map. After this, the map is segmented and clusters of pixels corresponding to upright objects are produced. Finally the geometric features of the 3D point cloud of each ROI are computed and the object/pedestrian is classified (figure 2.12) [32].

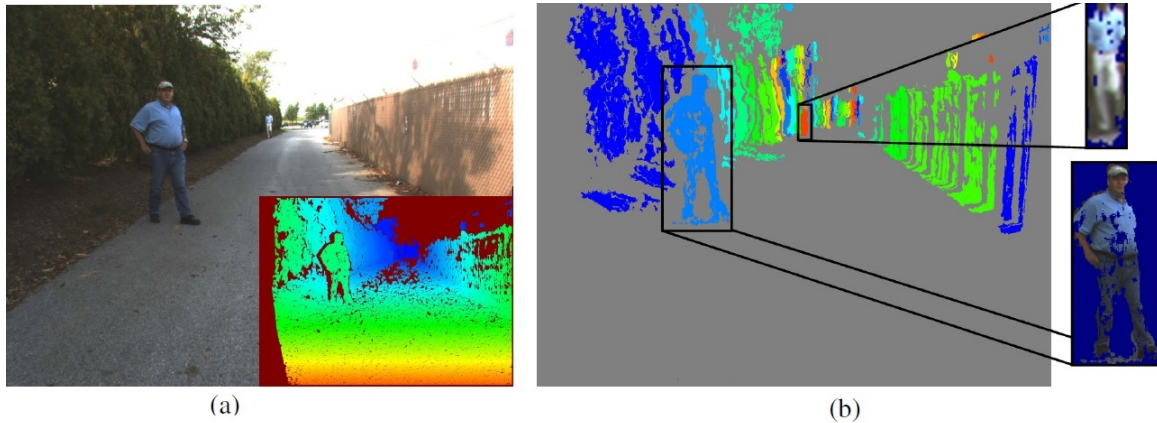


Figure 2.12: An example Multiple cameras based detection: (a) the left image of a stereo pair with the resulting depth map; (b) the segmented regions in different colors [32].

Since the objective of this project is to use a stereo camera system, it will be applied a similar technique, as the last one, to detect pedestrians and other targets.

The next section focuses on the technical facts of stereo vision.

## 2.5 Fundamentals of Stereo Vision

Before proceeding to stereo vision systems it is important to explain what is a stereo vision system and what are the processes required to it.

A stereo vision system is a set of synchronized cameras (horizontal or vertical aligned) in which the distance between each camera is named the baseline. A stereo vision system is capable of extracting 3D information from the surrounding scenario by comparing the collected information from two, or more, different points of view.

After the assembly of the setup, it is necessary to calibrate the system (as it will be explained in detail in chapter 5). The resulting parameters are then used to undistort and rectify the stereo pair of images. Basically, rectification is the process of correcting and aligning the two, or more, images so that corresponding points lie on the same image rows, i.e., the transformation causes epipolar lines (lines  $OL - X$  and  $OR - X$ , in figure 2.13) to become collinear. Rectification is commonly performed for reducing computation time during the search for correspondences since it reduces the search of correspondent points to a one-dimensional range. This constraint also reduces the number of incorrect matches [33].

The following step, after rectification, is correlation. Images from the left and right camera are used to match overlap points from one image to the other so that a correspondence can be found. For this step, an algorithm is used to measure similarity between two pixels, each one from each image, along the same epipolar line. Those pixels with highest accordance or

correlation are assigned as corresponding [33]. This will be better explained in section 4.3. The result of correlation is the disparity that corresponds to the number of pixels offset from the initial pixel that is being processed for a correspondence:  $D = (x_l - x_r)$ . (figure 2.14). All the disparity values for each correspondent pixel are then represented in a disparity map.

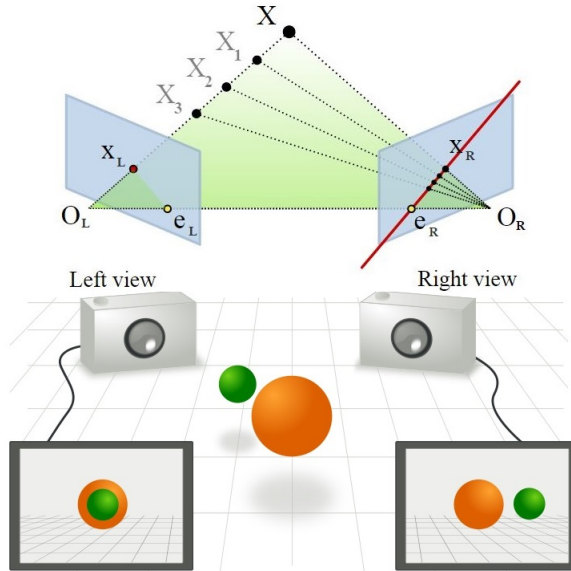


Figure 2.13: Epipolar Geometry [34]. On top,  $O_L$  is the center of the left camera and  $O_R$  is the center of the right camera. The line  $O_L - O_R$  represents the baseline while the lines  $X_L - e_L$  and  $X_R - e_R$  represents the epipolar lines between the cameras and the object. Notice that the epipolar lines and the baseline are in the same plane that intersects both camera planes. On the bottom it is shown the two points of view from a stereo vision system.

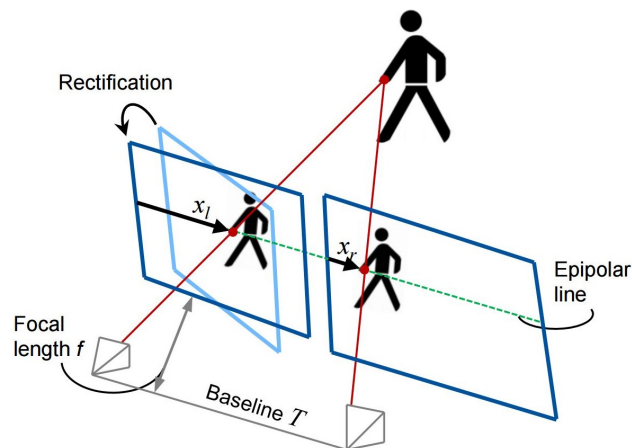


Figure 2.14: Epipolar geometry of a stereo camera setup [35].

Knowing the disparity values for each pixel and the other information about the setup it is possible to estimate the depth of the objects and re-construct the 3D scene, as presented in chapter 6.2.1.







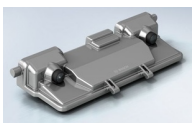
In the next section, stereo cameras systems available on the market are presented.

## 2.6 Stereo Systems

As said before, stereo vision is an ideal method for the autonomous driving problem for a variety of reasons. Visual information allows lane markings localization, traffic signs recognition and obstacle identification without requiring any modifications to road infrastructures or affecting the environment.

Nowadays there are several 3D stereo vision sensors available on the market. The most significant ones are presented in table 2.2. In order to compare and choose these kind of equipment there are four important features to pay attention to: resolution, frame rate, baseline and lens.

Table 2.2: Most significant stereo vision cameras on the market.

Model	Max Resolution	Frame Rate	Max Baseline	Photo	Notes
<b>Point Grey Research Bumblebee XB3</b> [36]	1280 x 960	16FPS	277mm		The same brand as the cameras used for this project
<b>PCI nDepth</b> [37]	752 x 480	60 FPS	60mm		It is possible to request custom baseline cameras
<b>Minoru 3D Webcam</b> [38]	800 x 600	30 FPS	60mm		On Ebay for 30\$
<b>Scorpion 3D Stinger</b> [39]	1280 x 1024	-	30 mm e 200mm		Designed especially for industrial applications
<b>VisLab 3DV-E</b> [40]	640 x 480	25 FPS	150mm		Low-cost and low-power dense stereo reconstruction system
<b>Ensenso N20</b> [41]	1280 x 930	30 FPS	100mm		Working distance up to 3m.
<b>Bosch Stereo Video Camera</b> [20]	1280 x 1024	-	120mm		Used in Land Rover Discovery Sport integrated with emergency braking system.

All the stereo cameras shown in table 2.2 have short baselines (between 30 to 300mm). This makes the system very portable and easy to use in close scenarios. However, there are many examples of stereo systems with wide baselines and with largest working distances, which are presented in the next section.

## 2.7 Large baselines for ADAS

Autonomous vehicles require the detection of pedestrians, obstacles, lane and traffic signs as it was said before. However, the perception of the largest surrounding environment possible has to happen in real time. As the driving speed increases so the stopping distance, therefore, stereo systems for autonomous cars or ADAS applications must be able to detect the environment at larger distances and for that some literature indicates that a wide baseline system is most suitable for ADAS. [16]

In table 2.1 there are four examples of vehicles wide baselines stereo systems for ADAS application. TerraMax, VIAC, AnnieWay and BRAiVE.

TerraMax was developed by VisLab in cooperation with Oshkosh Truck Corp. and Rockwell Collins. This autonomous vehicle participated and finished the 2005 DARPA Grand Challenge. Although this car did not win the competition it is a great example of an unmanned vehicle relying mainly on vision to apprehend the environment. For this project, a three-camera system (figure 2.15) was developed allowing the vehicle to sense a wide scenario. The existence of three cameras allows the vehicle to choose the best baseline (0.5, 1 and 1.5m) according to the car speed, covering a 7-50m range. "Higher speeds required greater sensing distances and thus wider baselines." [10] In this project other sensors were used: 4 monocular cameras to cover 360° around the vehicle and 3 LIDARs (figure 2.16). During the challenge TerraMax drove totally autonomously reaching speeds up to 68km/h recognizing the best path and avoiding obstacles.



Figure 2.15: Trinocular System used in TerraMax [16].

VIAC (VisLab Intercontinental Autonomous Challenge) was a trip from Italy to China (15.900km) in 2010 and it was the first intercontinental land journey completed by autonomous cars. During the challenge two identical vehicles were driving in a leader-follower way. "The first vehicle conducts experimental tests on sensing, decision, and control subsystems, and collects data throughout the whole trip. Human interventions are needed to define the route and intervene in critical situations." [42]. The second one follows the route defined by the first car needing no human intervention. If the leader is visible it uses its front vision systems to follow it, if not it uses GPS way points of the first vehicle. The main objective with this experience was to collect data to create a large set of scenarios for further studies so that in a close future autonomous vehicles can drive in the inner part of cities. For this challenge a stereo vision system was developed with a 0.8m baseline. All sensors used and their range are shown in figures 2.17 and 2.18.

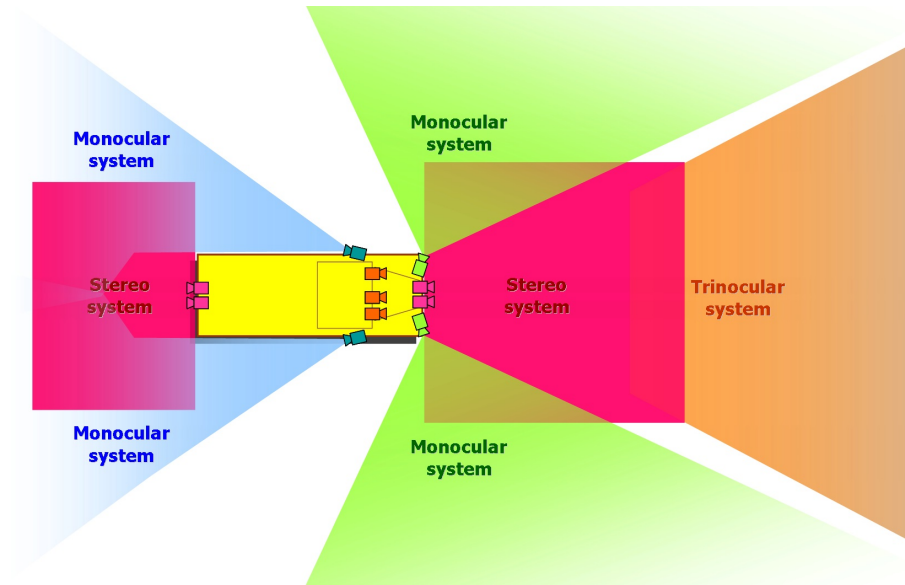


Figure 2.16: Representation of the area covered by all sensing systems used in TerraMax [16].



Figure 2.17: Left: frontal stereo vision systems to detect lane markings and obstacles; right: frontal panoramic vision system to locate the leader vehicle [42].



Figure 2.18: Back stereo system used to detect upcoming traffic and obstacles during backup maneuvers [42].

The Annieway project from Karlsruhe Institute of Technology started in 2012 with the main objective of collecting data with a common vehicle equipped with several sensors (figure 2.19). On top of the vehicle there are two grayscale cameras and two color cameras with 48cm and 54cm baselines, respectively. The authors choose these two types of cameras "since color images are very useful for tasks such as segmentation and object detection, but provide lower contrast and sensitivity compared to their grayscale counterparts, which is of key importance in stereo matching and optical flow estimation" [18]. This project also used a Velodyne to provide accurate 3D information from moving platforms, and a GPS. In the end of this experiment AnnieWay had driven through more than 40km and collecting 389 stereo and optical flow images that are available at the project webpage: The KITTI Vision Benchmark Suite. This benchmark is a great contribution to ADAS studies since it provides real data for testing.



Figure 2.19: Position and description on each sensor of Annieway [18].

BRAiVE - BRAin driVE - is another and more recent autonomous vehicle developed by VisLab Research Group incorporating all the previous knowledge of the group in this field. In 2013 it drove 2000km, in a real urban scenario, on autonomous mode 98% of the time and using the sensors shown in figure 2.20. For pedestrian detection it uses a monocular camera and 4 planes laser scanners [19]. This research project has the innovation of looking like a common vehicle since all sensor are camouflaged: this proves that every vehicle with few modifications can be autonomous.

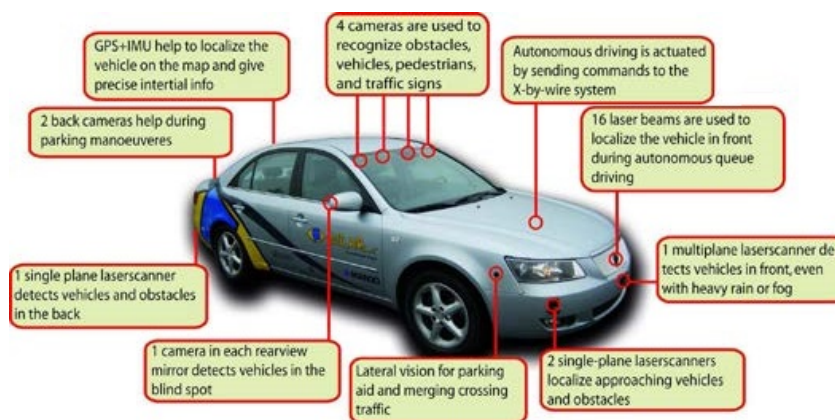


Figure 2.20: Position and description on each sensor of BRAiVE [19].





## Chapter 3

# Stereo Rig's Development

In order to achieve the goals of this thesis it is necessary to define the stereo vision system setup. This step is very important since it can affect the next steps of the project and also the results.

First it is needed to choose the cameras specifications and lenses (section 3.1). Next, depending on the chosen cameras it may be necessary the use of a host adapter to connect both cameras to the computer (section 3.2). Because the cameras must be synchronized, the system setup has to include a synchronization device (section 3.3). Finally both cameras need to be fixed and assembled to the vehicle or a testing table (section 3.4).

### 3.1 Cameras

The cameras chosen for this project were two Point Grey Flea3 GigE Cameras, model FL3-GE-28S4C-C (figure 3.1).



Figure 3.1: Point Grey Flea3 GigE [43].

These are color cameras with 2.8MP and equipped with an imaging sensor with the following specifications [43]:

- Sony ICX687 CCD, 1/1.8", 3.69  $\mu\text{m}$ ;
- Global Shutter;
- Resolution 1928 x 1448 pixels at 15 FPS.

In figure 3.2 it is represented the size of the cameras and how they connect to a computer or a switch.

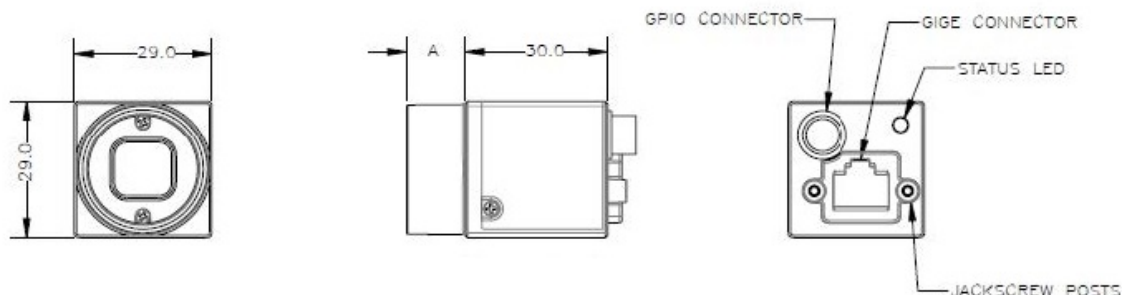


Figure 3.2: Flea3 GigE Dimensional Diagram [43].

As the name of the model and figure 3.2 indicate, these cameras are connected with the computer using GigE Vision, an interface for industrial cameras that uses the Gigabit Ethernet communication protocol. Using this global camera interface it can be transferred large images in real time (up to 125MB/s) using low cost CAT5 or CAT6 cables up to 100 meters in length without uncompromised data [44]. In this case two CAT6 cables with 4 meters length were specially made for the project.

In order to power and synchronize the cameras two GPIO (General Purpose Input/Output) cables with 4 meters length each are used. In figure 3.3 it can be seen the diagram of the connector and the description of each pin and surrounded in a blue rectangle there are the pins in use in this project.

Diagram	Color	Pin	Function	Description
	Black	1	IO	Opto-isolated input (default Trigger in)
	White	2	O1	Opto-isolated output
	Red	3	IO2	Input/Output/serial transmit (TX)
	Green	4	IO3	Input/Output/serial receive (RX)
	Brown	5	GND	Ground for bi-directional IO, $V_{EXT} +3.3$ V pins
	Blue	6	OPTO_GND	Ground for opto-isolated IO pins
	Orange	7	$V_{EXT}$	Allows the camera to be powered externally
	Yellow	8	+3.3 V	Power external circuitry up to 150 mA

Figure 3.3: GPIO connector description [43].

To effectively power the cameras, a common variable DC power supply was used at 12V connecting pin 7 to the positive output and pin 5 to the ground output. The other two pins, 1 and 6, are used for synchronization as explained in section 3.3.

The Point Grey Flea3 GigE cameras were designed with a C-mount for lenses. This way, not only the effect of the baseline in stereo vision can be studied but also the influence of the focal length that is defined by the lenses. For this study there are two types of lenses available with focal lengths of 8 and 16mm (figure 3.4).



Figure 3.4: Point Grey Lenses available for this study [43].

## 3.2 Host Adapters

Since the computer used in this study, as well as most of computers, only has one Ethernet port it is necessary to have a host adapter. For this topic there were two solutions available: a switch TP-Link SC10008P that allows Gigabit communications, and a Network Interface Card with two Ethernet ports from Point Grey equipped with an Intel 82574L controller.

The switch was used in all the experimental tests presented in chapter 6. The Network Interface Card was not used in this study because all the work was done in a laptop so it was impossible to use it. However, it will be installed in ATLASCAR for future projects.

## 3.3 Synchronization

Stereo Vision Systems require synchronization in order to create a disparity map, which means that it is necessary to use images captured at the very exact moment, otherwise the result will not represent the environment at all. However, this model do not support software trigger so, in order to synchronize the cameras, an external trigger is used.

To synchronize the cameras, a mini setup is used with the following hardware, as it is shown in figure 3.5:

- White Board;
- 2 Electronic Cables;
- Arduino Nano V3.1;
- Mini USB cable (to Power).

Using the application Arduino IDE, the arduino nano was programmed to implement Pulse Wigth Modulation (PWM), a digital square signal where the frequency is constant but the fraction of time the signal is on may vary (duty cycle). For this, the function `digitalwrite` is used to create a square wave at 15Hz with a duty cycle of 50%. Pin 13 of the Arduino was defined as the output and was connected to pin 1 of both cameras. On the other side the ground pin of the Arduino was connected to both pins 6 of both cameras. The signal send by the Arduino to the cameras is then interpreted and two images are captured by each camera at that exact moment, as it was better explained in section 4.1.

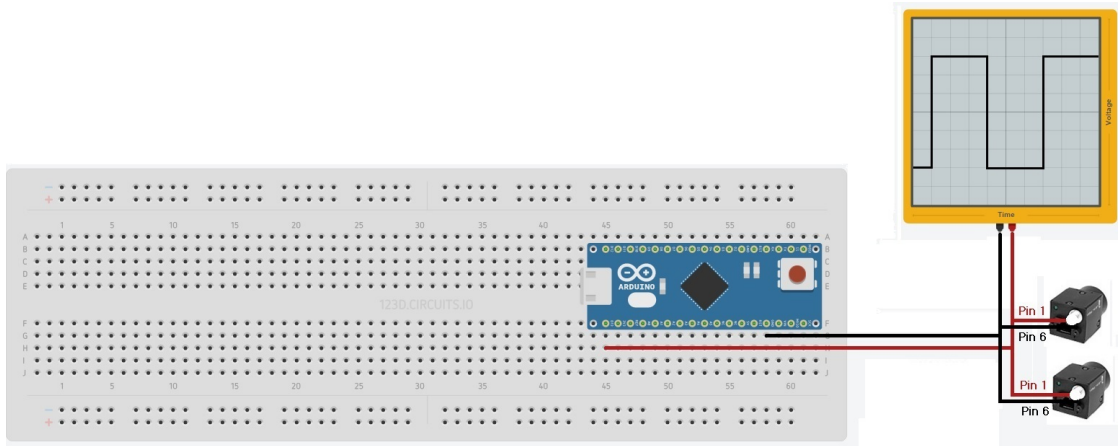


Figure 3.5: Synchronization setup.

### 3.4 Fixing and Assembling of the Cameras

Relatively to the fixing and assembling system there were some requirements:

- Precision - the axis of both camera must be parallel;
- Easy to assemble and disassemble - the camera's skate to the track;
- Firmly assembly - the cameras cannot move after assembled;
- Easy to assemble in ATLASCAR;
- Allow multiple baselines;
- Resistance and relatively low weight.

Knowing all this, the development of the fixing and assembling of the cameras can start.

Figure 3.6 presents a render of the camera skate project, designed in CATIA V5, showing how all the pieces assembled. The technical drawings are in the Appendix.

Next it is explained all the development process of the stereo system setup.

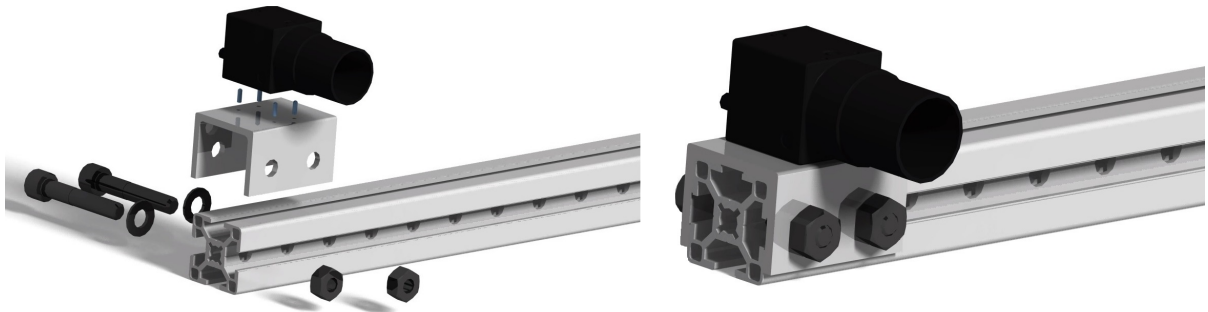


Figure 3.6: Stereo Camera Fixing and Assemble Project - render.

In figure 3.7 is a Bosch aluminum profile with 1.55 meters long and machined holes, in a CNC, allowing baselines from 0.3 to 1.5 meters with minimum interval of 2.5 centimeters

between holes. 0.3 to 1.5 meters was the chosen baseline range for the study as explains in chapter 6. This type of profile was chosen not only for its resistance to weather conditions, good material characteristics and relatively low weight but also because this type of profile allows more easy assembling to ATLASCAR that is already equipped with Bosch aluminum profile in its top. Thanks to the profile itself, it facilitates the camera skate assemble and movement when not assembled - like a track.



Figure 3.7: Aluminum profile used as rail for the fixing and assemble system.

In figure 3.8 is a real photo of how the camera is assembled in the camera skate - with 4 M2.4x4 screws and 4 M2.4 washers (for each camera), taking advantage of the threaded holes already existing in the cameras.



Figure 3.8: Camera skate - assembling to the camera.

The camera skate, figure 3.9, is made from an aluminum plate with 2mm thick. The holes were also machined in a CNC, however, the bending of the pieces were executed in a traditional press brake so the dimensional tolerances are not as small as they could be. Figure 3.9 shows how one camera skate is assembled in the Bosch profile, with 2 M6x40 screws and 2 M6 nuts. In order to give more precision to the assemble, since the bending was not as precise as it could be, it is also used 4 M6 washers in each camera skate assemble. Initially, it was designed and developed another camera skate that was 3D printed in another laboratory at the Mechanical Engineering Department. This camera skate would allow a more precise assembling and fixing because it was custom made for that specific geometry profile. However, the piece that was 3D printed revealed not resistant enough for this application due to some limitations and problems of the 3D printer.



Figure 3.9: Camera skate - assemble to the rail profile.

At last, in figure 3.10, there is a close up on the assemble of one of the cameras/skate.

With the setup defined, designed and assembled, the project can evolve to the next level, the calibration of the stereo system using the software developed in chapter 4.

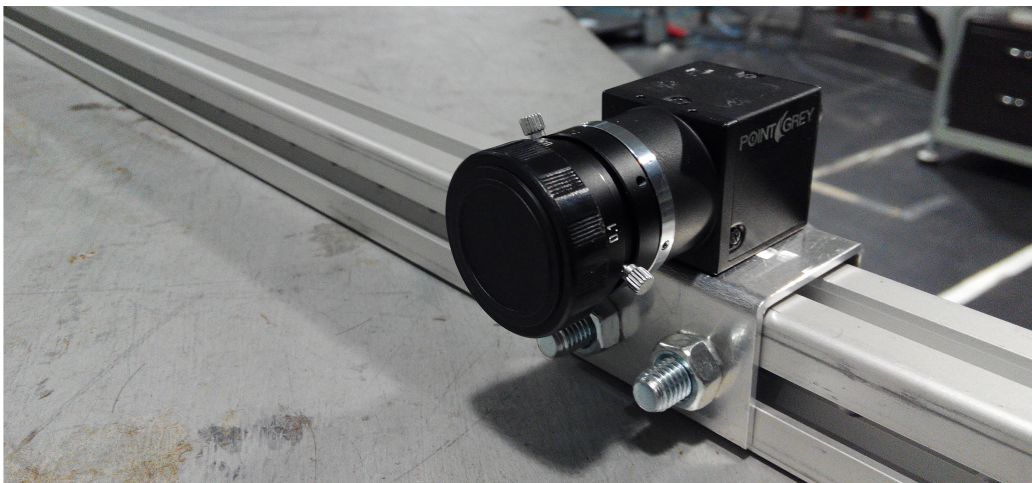


Figure 3.10: Close up to the fixing and assemble system.

## Chapter 4

# Stereo Camera Application

Initially, the application was developed in a non ROS environment and in the end it was adapted into a ROS package for it to be implemented on ATLASCAR.

In order to detect pedestrians there is an entire process that has to be done before. This process consists of the following steps:

- Camera Driver - section 4.2;
- Calibrate the stereo camera system - chapter 5;
- Rectify a pair of stereo images, compute disparity and reconstruct the 3-D point cloud - section 4.3
- Detect pedestrians and other obstacles on the road - section 4.4

The developed ROS package was named **StereoCam** and it has the main functions of first connecting and synchronizing the cameras and then publishing the acquired images and the camera's info. To process the stereo images, an already existing ROS package is used: **Stereo Image Proc**. This package subscribe the acquired images and the camera's info, rectify both images and then publishes the disparity map and the point cloud.

The non ROS application, **pgrFlea3stereo**, is capable of the same tasks as the ROS app but is also capable of detecting pedestrians and other objects on the road using algorithms from OpenCV library and from PCL - point cloud library.

This chapter presents the details of the development of the applications.

## 4.1 Software

### 4.1.1 ROS

ROS provides libraries and tools to help software developers create robot applications. Basically, it is an open source collection of software frameworks for software development that provides "the services you would expect from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management." [45].

The first ROS distribution was released in 2010 (ROS Box Turtle) and it is now in its 9th distribution (ROS Jade Turtle) released in 2015 and in use in the driver developed for this project.

Figure 4.1 outlines how the networks of processes work. A node is an executable that uses ROS to communicate with other nodes, and topics are a type of subject where nodes can publish messages or subscribe to receive messages. A package can have multiple nodes and are the main unit for organizing software in ROS. A ROS launch file can run multiple packages and nodes that can communicate between each other, which is a huge advantage of ROS. Other great advantage is the fact that the ROS community allows the exchange of stacks, packages and knowledge between the global users making of ROS a good platform to develop code.

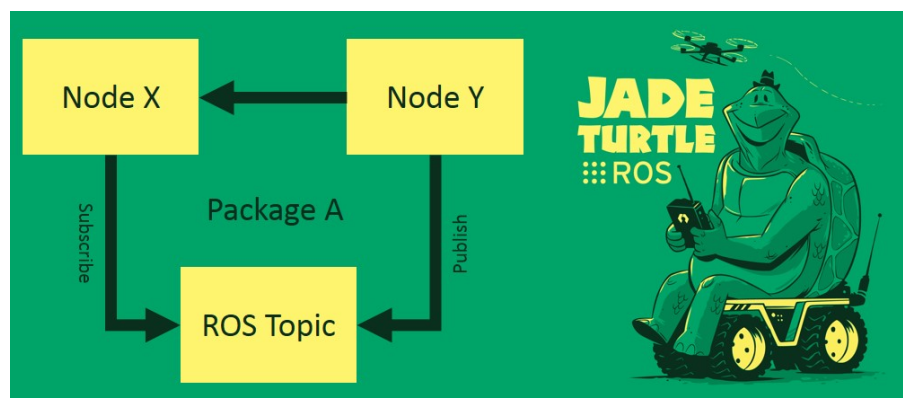


Figure 4.1: ROS simple network of processes.

#### 4.1.2 OpenCV

OpenCV - Open Source Computer Vision, is an open source computer vision and machine learning software library.

The library has more than 2500 optimized algorithms that "can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc." [46].

Thanks to all the useful algorithms that this library has developed, it is now used by companies like Google or Microsoft [46]. The library itself is used in many ROS packages and in this case it is used for calibration and to create the disparity map.

## 4.2 Application Development - Connect and Synchronize

As said before, it is necessary to develop a camera driver for this specific application to connect the two Flea3 GigE cameras and to capture and save synchronized images.

Point Grey Research developed a "FlyCapture Software Development Kit" with a complete software API library that provides many examples of programs, in C++, that help the user to understand how the Flycapture functions can be used. For the development of the camera drive the following demo-programs were used:



- `MultipleCameraEx` - to connect both cameras
- `CustomImageEx` - to change image properties (format7)
- `AsyncTriggerEx` - to synchronize both cameras
- `GigEGrabEx` - to learn how the GigE functions worked

The Stereo Camera Application allows two cameras to connect to a computer and publishes images at 15FPS (the maximum possible for the maximum resolution of cameras - 1928x1448). It was decided to work with the maximum resolution possible, despite it decreases the frame rate, since the objective is to study which baseline would be ideal for this application so, the higher resolution possible is needed to capture more detailed information about the surrounding environment. However, in future projects, if needed, the code allows easier modifications to change the resolution since other projects may prefer other resolutions.

The chosen cameras can implement different video modes. From the ones available, "Video mode 0", as named by Point Grey Research was chosen. This Video Mode does not perform pixel aggregation, i.e. binning, and uses a faster pixel clock, comparing to the others video modes available, which translates in faster frame rates. [47]

Pixel format is other important topic since it is the encoding scheme by which images are produced from raw image data. The cameras used in this thesis allows different types of pixel format and from that "Mono" was chosen. Using this pixel format the image data is monochrome, i.e. grayscale images, which allows faster frame rate when comparing to RGB or YUV, other pixels format available that are color-encoding. Color-encoding is not necessary in stereo vision systems since the algorithm used for stereo correspondence is not color sensitive and decreases the frame rate. It was also decided that the number of bits per pixel would be 8 in order to increase the frame rate, once more. [47]

For the camera's model in use the software trigger is not an option, as Point Grey Research informed. In order to synchronize the cameras an external trigger must be developed. There are many external trigger modes available differing on the type and purpose of the synchronization, all of this using the GPIO pins. The "Trigger Mode 0", as Point Grey named it, was chosen because it allows very precise synchronization and that is exactly what is needed. This is considered to be the standard external trigger mode and basically "when the camera is put into "Trigger Mode 0", the camera starts integration of the incoming light from external trigger input falling/rising edge." [47] (figure 4.2). Other interesting trigger mode possibility was mode 14 because it allows higher frame rates however, sometimes it stopped synchronizing the cameras and that is the reason it was not used.

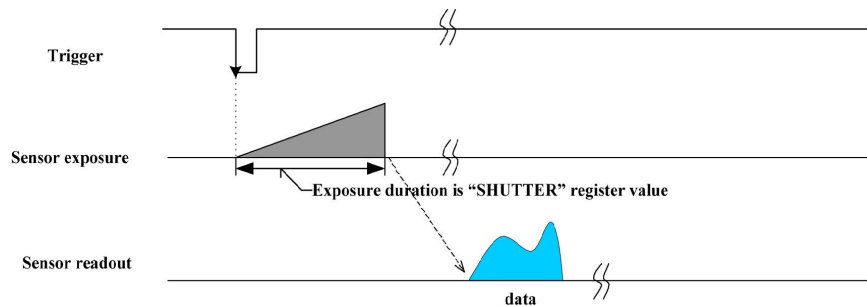


Figure 4.2: Trigger Mode 0 ("Standard External Trigger Mode") [47].

An important detail in this software application is that it verifies if the first camera connected is always the one it is defined as the "left camera". This is necessary because the calibration only works in the specific setup that was in use when the calibration was executed. So, the left camera must always be the left camera and so on. Because of this it is defined that the camera with a specific mac address would always be the camera 0 - left and the camera with other mac address would always be the camera 1 - right, since the mac address of the cameras never changes.

When running this package, as the User Instructions will explain in the Appendix, image consistency errors occur sometimes when trying to retrieve buffer. "Image consistency errors are often caused by dropped packets. Ethernet is asynchronous in nature and capable of bursting data to peak bandwidth. A sudden burst in data transfer can create a packet collision and lead to image consistency errors." [48] This issue was tried to be solved by increasing the packet size and the packet delay, as the help desk of Point Grey Research suggested, however, the problem could not be solved completely. Another possible solution was to use GigE functions but this functions revealed not to be stable enough. The practical solution was to drop an image if the retrieve buffer of the other camera was not possible. It is important to say that this error occur in an average of 10 in 200 images (5%) so the program and the final objective is not really affected by this issue.

At last, in figure 4.3, there are some synchronized frames from a sequence of images captured by the cameras. The images are still in an original format and are not rectified yet. The left image of each captured moment was captured by the left camera (camera 0) and the right one was captured by the right camera (camera 1). The sequence of the captured frames is presented left-right-down.

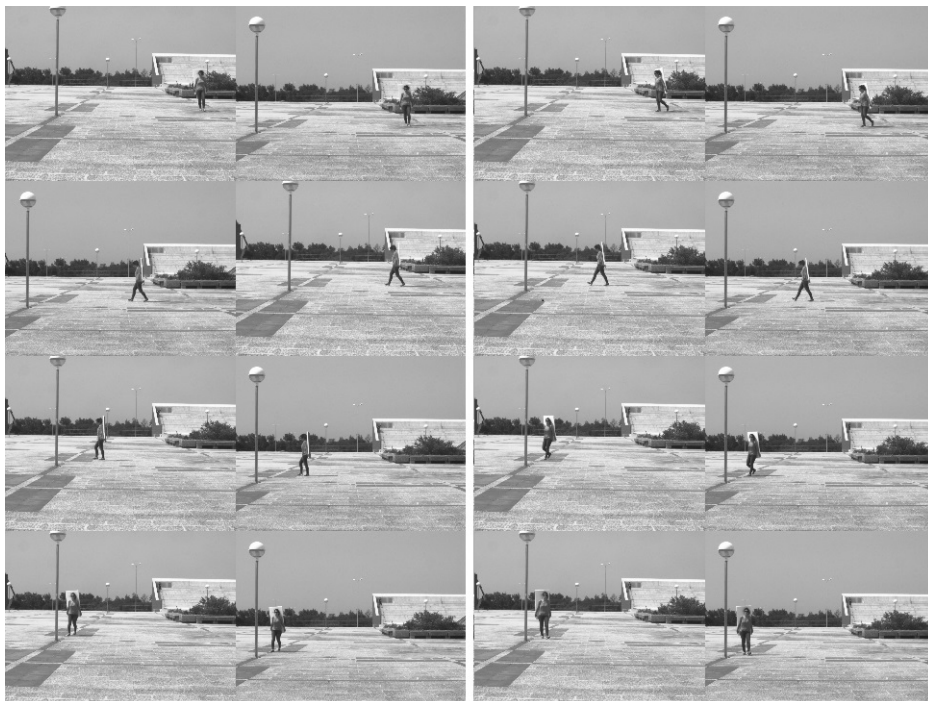


Figure 4.3: Frames from a sequence captured using StereoCam driver.

Next it is explained how stereo is calculated and how this application process stereo.

## 4.3 Stereo Image Processing

”Stereo vision is the process of recovering depth from camera images by comparing two or more views of the same scene” [49]. Or, in a more simple way, stereo image processing consists of undistorting and rectifying the pair of images, based on the calibration results (section 5), and compute disparity images from incoming stereo pairs using an algorithm.

### 4.3.1 Stereo Correspondence Algorithm

In order to choose the stereo correspondence algorithm, the following three specifications were defined:

- Open source code;
- High density stereo, for a complete disparity map;
- Computer processing time less than 2s, since targets have to be detected at a safe distance.

To choose the stereo correspondence algorithm the ”Stereo Evaluation 2015” ranking was consulted. This ranking uses the data collected by the AnnieWay project (in section 2.2 to evaluate stereo algorithms and others for ADAS - The KITTI Vision Benchmark Suite [50]). This benchmark includes 200 training scenes and 200 test scenes in an urban scenario. The stereo systems are evaluated according to some parameters, however, we are only interested in the ones that it was referred before. Figure 4.4, shows the ranking list of the stereo algorithms.

	Method	Data	Code	D1-bg	D1-fg	D1-all	Density	Time	Environment
1	<a href="#">MC-CNN-acrt</a>		<a href="#">code</a>	2.89 %	8.88 %	3.89 %	100.00 %	67 s	Nvidia GTX Titan X (CUDA, Lua/Torch7)
J. Zbontar and Y. LeCun: <a href="#">Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches</a> . Submitted to JMLR									
2	<a href="#">SPS-St</a>		<a href="#">code</a>	3.84 %	12.67 %	5.31 %	100.00 %	2 s	1 core @ 3.5 Ghz (C/C++)
K. Yamaguchi, D. McAllester and R. Urtasun: <a href="#">Efficient Joint Segmentation, Occlusion Labeling, Stereo and Flow Estimation</a> . ECCV 2014.									
3	<a href="#">OSF</a>		<a href="#">code</a>	4.54 %	12.03 %	5.79 %	100.00 %	50 min	1 core @ 2.5 Ghz (C/C++)
M. Menze and A. Geiger: <a href="#">Object Scene Flow for Autonomous Vehicles</a> . Conference on Computer Vision and Pattern Recognition (CVPR)									
4	<a href="#">MBM</a>			4.69 %	13.05 %	6.08 %	100.00 %	0.13 s	1 core @ 3.0 Ghz (C/C++)
N. Einecke and J. Eggert: <a href="#">A Multi-Block-Matching Approach for Stereo</a> . IV 2015.									
5	<a href="#">PR-Sceneflow</a>		<a href="#">code</a>	4.74 %	13.74 %	6.24 %	100.00 %	150 s	4 core @ 3.0 Ghz (Matlab + C/C++)
C. Vogel, K. Schindler and S. Roth: <a href="#">Piecewise Rigid Scene Flow</a> . ICCV 2013.									
6	<a href="#">AABM</a>			4.88 %	16.07 %	6.74 %	100.00 %	0.08 s	1 core @ 3.0 Ghz (C/C++)
N. Einecke and J. Eggert: <a href="#">Stereo Image Warping for Improved Depth Estimation of Road Surfaces</a> . IV 2013.									
7	<a href="#">SGM+C+NL</a>		<a href="#">code</a>	5.15 %	15.29 %	6.84 %	100.00 %	4.5 min	1 core @ 2.5 Ghz (C/C++)
H. Hirschmüller: <a href="#">Stereo Processing by Semiglobal Matching and Mutual Information</a> . PAMI 2008. D. Sun, S. Roth and M. Black: <a href="#">A Quantitative Analysis of Current Practices in Optical Flow Estimation and the Principles Behind Them</a> .									
8	<a href="#">SGM+LDOF</a>		<a href="#">code</a>	5.15 %	15.29 %	6.84 %	100.00 %	86 s	1 core @ 2.5 Ghz (C/C++)
H. Hirschmüller: <a href="#">Stereo Processing by Semiglobal Matching and Mutual Information</a> . PAMI 2008. T. Brox and J. Malik: <a href="#">Large Displacement Optical Flow: Descriptor Matching in Variational Motion Estimation</a> . PAMI 2011.									
9	<a href="#">SGM+SE</a>			5.15 %	15.29 %	6.84 %	100.00 %	45 min	16 core @ 3.2 Ghz (C/C++)
H. Hirschmüller: <a href="#">Stereo Processing by Semiglobal Matching and Mutual Information</a> . PAMI 2008. M. Hornacek, A. Fitzgibbon and C. Rother: <a href="#">SphereFlow: 6 DoF Scene Flow from RGB-D Pairs</a> . CVPR 2014.									
10	<a href="#">SNCC</a>			5.36 %	16.05 %	7.14 %	100.00 %	0.08 s	1 core @ 3.0 Ghz (C/C++)
N. Einecke and J. Eggert: <a href="#">A Two-Stage Correlation Method for Stereoscopic Depth Estimation</a> . DICTA 2010.									
11	<a href="#">SGM+CNN</a>			5.82 %	16.34 %	7.57 %	99.31 %	10 s	1 core @ 2.5 Ghz (C/C++)
Anonymous submission									
12	<a href="#">ELAS</a>		<a href="#">code</a>	7.86 %	19.04 %	9.72 %	92.35 %	0.3 s	1 core @ 2.5 Ghz (C/C++)
A. Geiger, M. Roser and R. Urtasun: <a href="#">Efficient Large-Scale Stereo Matching</a> . ACCV 2010.									
13	<a href="#">REAF</a>		<a href="#">code</a>	8.43 %	18.51 %	10.11 %	100.00 %	1.1 s	1 core @ 2.5 Ghz (C/C++)
C. Ciela: <a href="#">Recursive Edge-Aware Filters for Stereo Matching</a> . The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)									
14	<a href="#">OCV-SGBM</a>		<a href="#">code</a>	8.92 %	20.59 %	10.86 %	90.41 %	1.1 s	1 core @ 2.5 Ghz (C/C++)
H. Hirschmüller: <a href="#">Stereo processing by semiglobal matching and mutual information</a> . PAMI 2008.									

Figure 4.4: Stereo Evaluation 2015 [50].

The chosen algorithm was the OpenCV SGBM (Semi Global -Block Matching). However, others algorithms were considered to be used in the beginning such as ELAS and REAF. Although ELAS has better evaluation results the code uses deprecated functions and the ROS package is not available anymore in ROS repository. On the other side, REAF, although the code is available, in fact it is but only after payment.

The OpenCV SGBM algorithm is a block matching technique where the sum of absolute distances (SAD) is used to find correspondences, along the same epipolar lines. In order words, SAD is the difference between a pixel in the original block and the corresponding pixel in the block used for comparison. There are three steps to the block matching technique that OpenCV uses: pre-filtering, correspondence search, and post filtering [35]. This algorithm allows the user to change and choose the some parameters for a better disparity map [51]:

- `minDisparity` - minimum possible disparity value (usually it is 0).
- `numDisparities` - it must be divisible by 16. In practice the bigger the value the better the disparity image at close range.
- `SADWindowSize` - matched block size (it must be an odd number  $\geq 1$ ) - SAD windows are used as a scoring method for each pixel in the image based on its surrounding neighbors.
- `P1` and `P2` - Affects the disparity smoothness.
- `preFilterCap` - truncation value for the prefiltered image pixels. The algorithm first computes x-derivative at each pixel and clips its value by `[-preFilterCap, preFilterCap]` interval.
- `uniquenessRatio` - margin in percentage by which the best (minimum) computed cost function value should "win" the second best value to consider the found match correct.
- `disp12MaxDiff` - maximum allowed difference in the left right disparity check
- `speckleWindowSize` - maximum size of smooth disparity regions (speckles)
- `speckleRange` - maximum disparity variation within each connected component.
- `fullDP` - if true it runs the full-scale two-pass dynamic programming algorithm. By default, it is set to false.

These parameters are not intuitive to define, it requires some time to understand in what each one of them affects the disparity map.

After choosing the algorithm for stereo it is time to use it as we will see next.

### 4.3.2 Calculation of the Disparity Map

For stereo image processing the OpenCV example `stereo_match` was adapted in order to update the Camera Driver Application (in non ROS environment). This sample uses two synchronized images and the extrinsic and intrinsic parameters, for image rectification and undistortion.

First, the code was adapted so it could read multiple images, previously saved, from a

folder or to function in real time, save the disparity images into a file and to use only the SGBM algorithm. The disparity images are visualized using OpenCV.

Next, a quick study on the influence of the SGBM parameters was done, using this application. However, a more complete study on the influence of the SGBM parameters was done using the ROS application and the `dynamic_reconfigure` explained in section 4.5. It is important to highlight that in other scenarios it may be needed to adjust the parameters again.

Figure 4.5 presents an example of two pair of stereo images in the acquired format, rectified and the disparity map created for this project, using the ROS app. However, the results using the non ROS application are the equivalent. The ground does not appear in the disparity map since it is removed by increasing the `uniquenessRatio` parameter, explained before in this section.

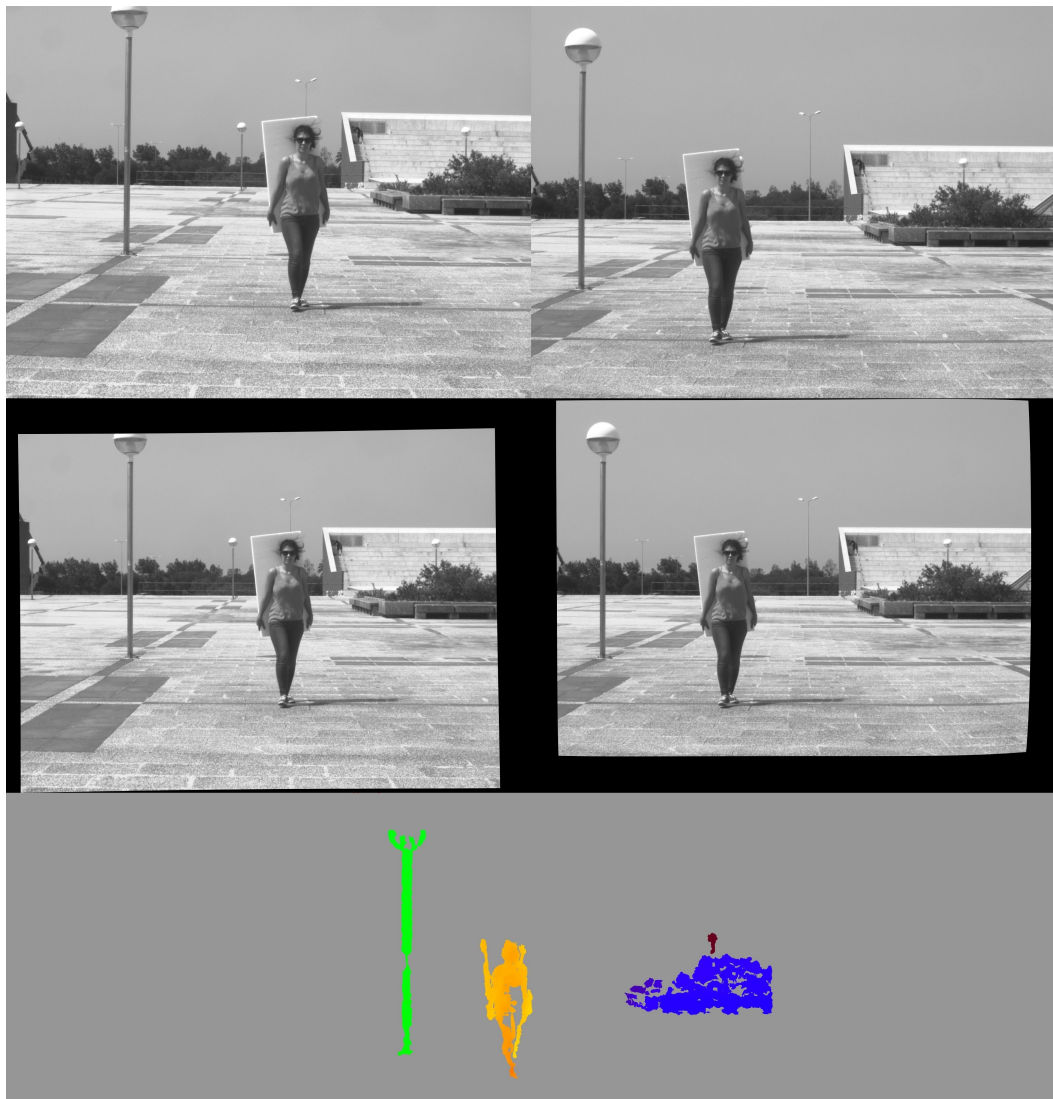


Figure 4.5: Example of two original acquired images, two rectified images and the resulting disparity map, top to bottom.

## 4.4 Object detection

After the disparity image processing part completed the next step is to try to detect objects on the road.

The code was based on the example `opencv_reproject_image_pcl`, by Martin Peris [52], which was useful to understand how the point cloud can be created and how it can be visualized. Then, it became necessary to analyze the point cloud and for that some tutorials available in the Point Cloud website [53] were used.

In order to create the point cloud, the function `reprojectImageTo3D`, from the OpenCV library, is used and then the 3D coordinates are inserted into the point cloud structure. The point cloud created has the field `XYZ` since every point of the point cloud has 3 coordinates (X, Y and Z).

The point clouds created using the Stereo Camera Application, usually, had more than 2 million points and noise. Because of that it is necessary to apply filters, from the Point Cloud Library, in order to reduce processing time and to reduce noise in the point cloud. As it can be seen in the diagram in figure 4.10, first it is used a `Pass Through Filter` [54] to cut off values that are outside the range defined. In this case points with distances bigger than 50m from the cameras are removed. Then the `Statistical Outlier Removal Filter` [55], that removes isolated points. The last one is the `Voxel Grid Filter` [56] that is a downsample filter, which means that it reduces the density of the point cloud.

Finally the point cloud is divided in clusters and since there are less points to process, the computational cost of cluster search is reduced. This clusters (regions) are parts of the filtered point cloud that are divided according to some parameters, in order to try to locate objects in the road. In this case, the `Euclidean Cluster Extraction` algorithm, from PCL, is used and it is necessary to define three parameters [57]:

- **Cluster Tolerance:** tolerance (in mm) between points. If this value is too small values objects can be divided in several clusters, if is too big different objects may be in the same cluster [58];
- **Minimum Cluster Size:** minimum number of points that a cluster can have;
- **Maximum Cluster Size:** maximum number of points that a cluster can have;
- **Search Method:** in this case `KdTree` - to find the K nearest neighbors points within the radius defined by the parameters above.

The result of the `Euclidean Cluster Extraction` are two clusters. Since the objective for this project is to know at what distance the pedestrians or obstacles are, a function from PCL is used (`Centroid`). The result is the coordinate of the geometric center of the point cloud that represents the coordinate of the center of mass of each object. This calculation is fully automatic and, for easier comparison, every coordinate of each centroid are saved in a text file.

Figure 4.6 presents an example of a point cloud calculated from an acquired frame, not filtered. The ground does not appear in the disparity map and in the point cloud since it had been removed by increasing the `uniqueness_ratio` parameter of the stereo correspondence algorithm. After filtering this point cloud only three clusters can be seen: the light pole and the 2 pedestrians that are passing in front of the cameras.

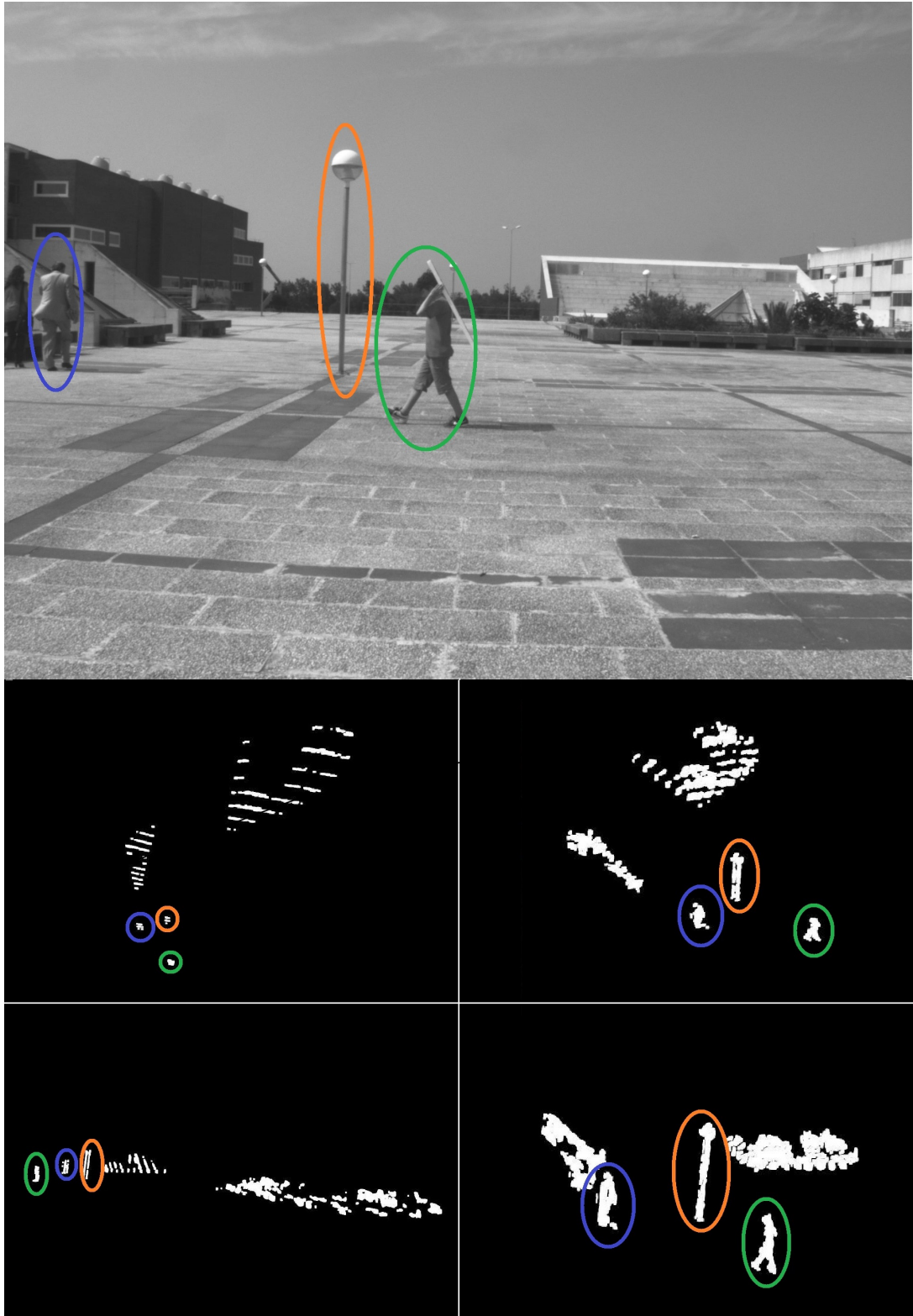


Figure 4.6: Different angles of visualization of the same point cloud, using PCL Visualizer. Circled in green is the closer pedestrian, in blue is the other pedestrian that is between the previous pedestrian and the light pole (circled in orange).

## 4.5 ROS Application

In ROS repository there is already a package for Point Grey Cameras - `pointgrey_camera_driver`, however, it was specially designed for another camera models (Bumblebee XB3 and Chamaleon). Because of that, it was decided to develop an application in non ROS environment and after adapt it into a new package for this cameras.

In general, both applications are very similar. The only differences are related to the structure of ros nodes. Related to ROS loop rate, it is important to say that it also affects the frame rate so its value was optimized in order to maximize the frame rate. The other higher difference is that ROS has a package for image visualization (image view). So for the stereo camera app for ROS the captured images are not visualized using OpenCV but instead the `image_view` package for ROS it is used.

For image stereo processing in ROS an already existing ROS package was chosen - `stereo_image_proc` that implements the SGBM algorithm. Basically it subscribes the `image_raw` and the `camera_info` from each camera and then, it performs rectification (the transformation process used to project two-or-more images onto a common image plane). After rectifying the stereo cameras it publishes the disparity image (figure 4.7) and the point cloud can be visualized in `rviz`. Figure 4.8 shows an example of how this package works. To visualize the rectified images and the disparity map it is also used `image_view`.

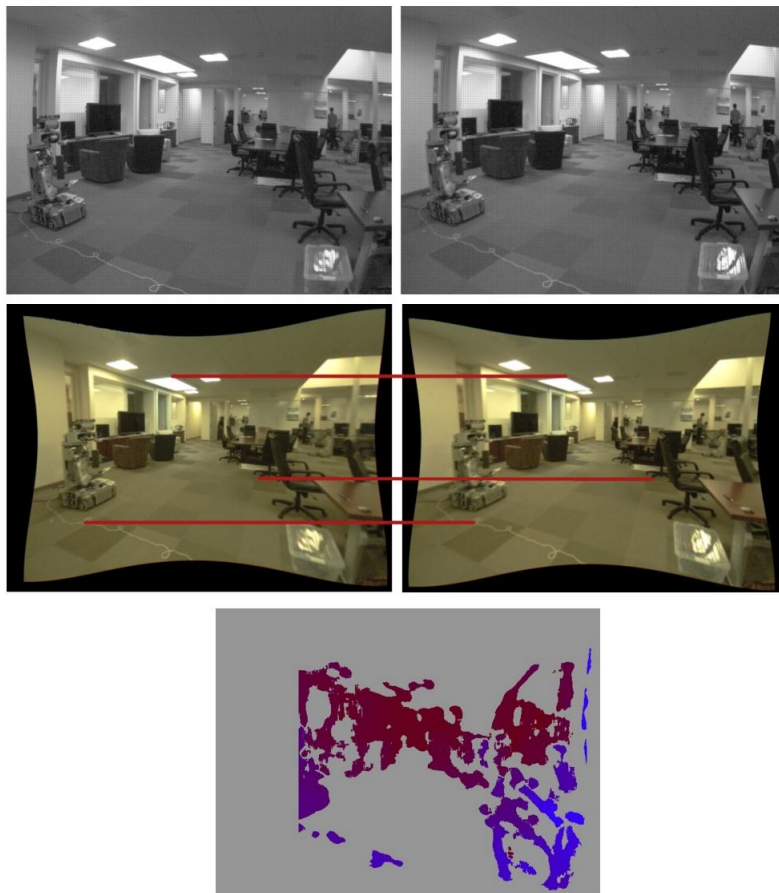


Figure 4.7: ROS Stereo Processing example [59].



A huge advantage of using ROS `stereo_image_proc` is that it allows the use of the `dynamic_reconfigure` to define and study the stereo parameters. In figure 4.9, it is shown how this application looks like. Basically it provides a standard way to expose a subset of a node's parameters to external reconfiguration which is very useful when working with hardware drivers. The use of the `dynamic_reconfigure` was a great help in discovering what would be the better stereo algorithm parameters since the values can be changed at the same time as previously saved images are processed. This way the disparity image can be visualized continuously changing according to the value set for the parameter, so it was quicker to understand their influence.

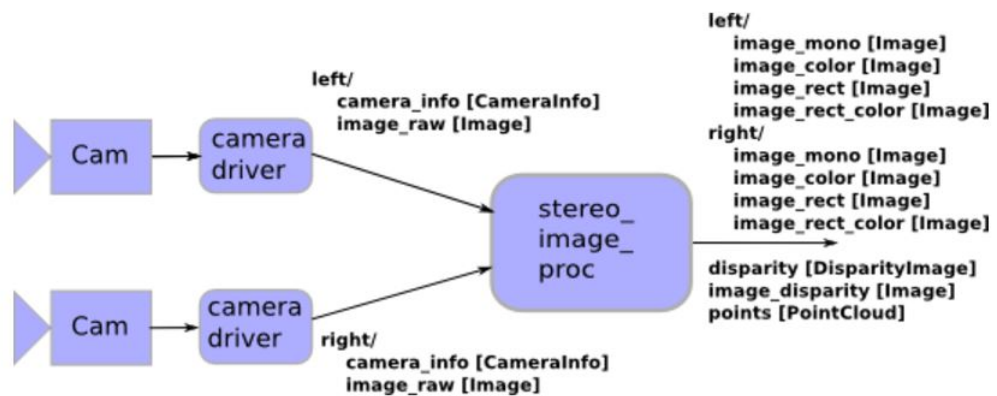


Figure 4.8: ROS `stereo_image_proc` diagram [59].

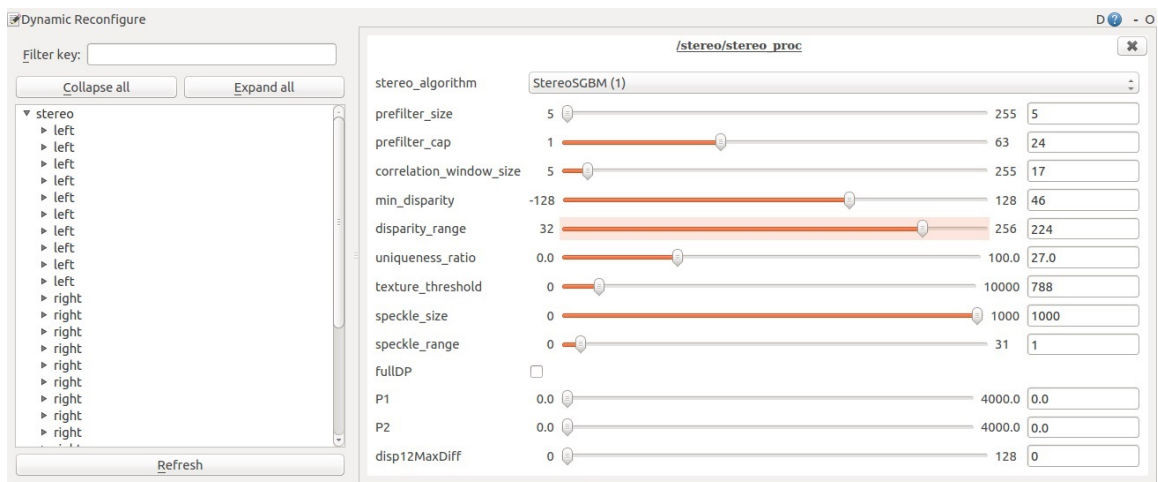


Figure 4.9: ROS Dynamic Reconfigure.

## 4.6 Stereo Camera Application - Global View

One of the main objectives when developing an application is for it to be user friendly. Because of this, all the user has to do to start any of the applications explained before is to compile and run one file (see Appendix). Eventually in different conditions the user may have to change some configurations such as camera settings, SGBM parameters or PCL filters.

Figure 4.10 shows a diagram of the Stereo Camera Application for an easier understanding of this app.

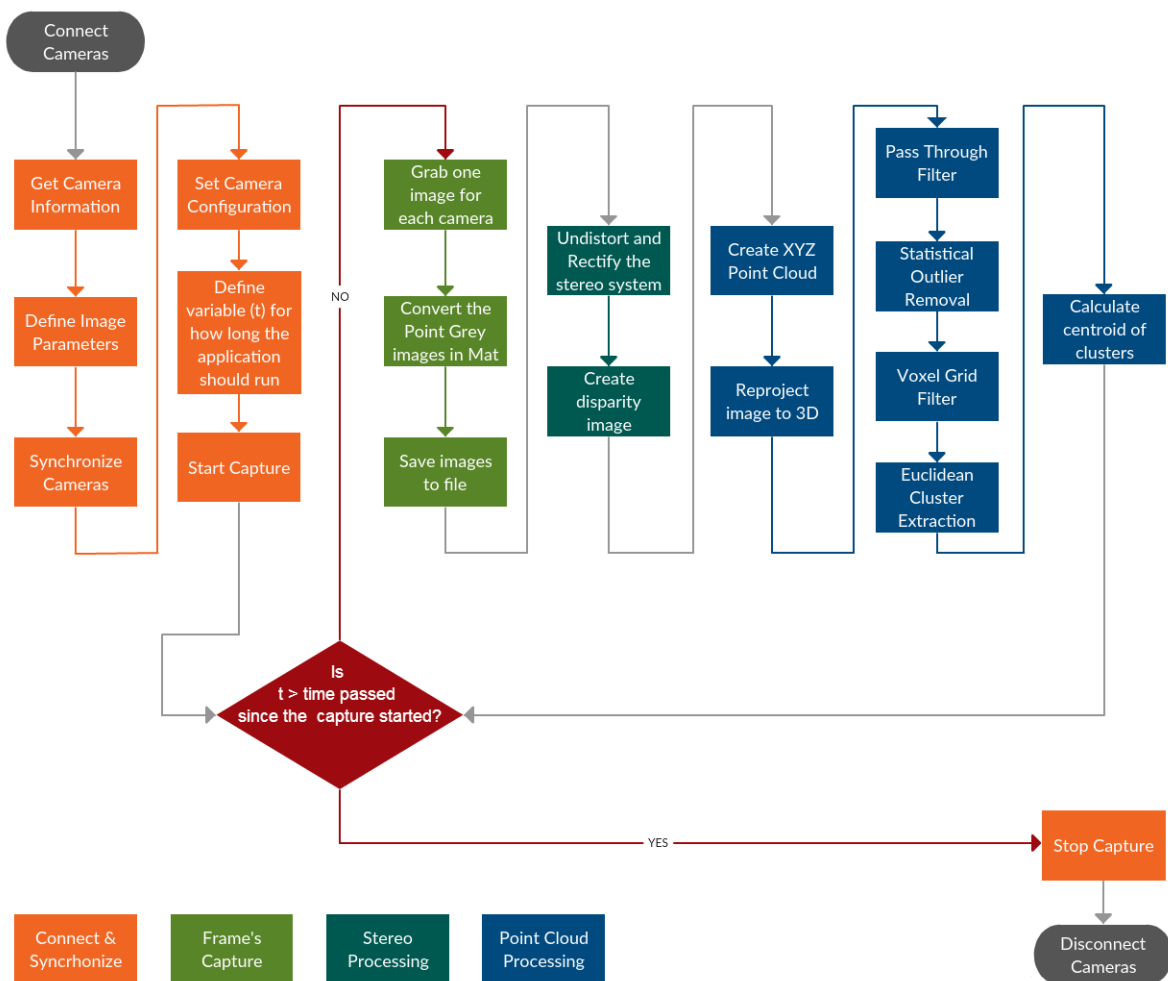


Figure 4.10: Stereo Camera Application Diagram.

## Chapter 5

# Stereo Camera Calibration

Camera calibration is a necessary step in stereo image processing as explained before.

The process of calibration consists of determining the intrinsic and extrinsic parameters of the stereo camera system that represent the geometrical relationship between the two cameras in space and each camera itself. With this information, the pair of images can be undistorted and rectified, which corrects any distortion of the lenses, any rotation on the optical axis of the two cameras (that must be parallel) and row-align the image planes.

Without stereo camera calibration there is no disparity image or reconstructed scene in 3D and the cameras must be calibrated every time anything in the camera setup is changed such as lenses, baseline, focus length or lenses aperture.

In this chapter it will be explained what is calibration, how it is done and the results of calibration for this setup.

### 5.1 Fundamentals of Stereo Calibration

The intrinsic parameters are represented by the camera matrix of each camera and the distortion matrix, of each camera as well. The camera matrix  $A$

$$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (5.1)$$

contains the parameters  $c_x$  and  $c_y$  that represent the image center, which is determined by the optical axis of the lens, and it is typically different from the center of the CCD sensor array. The parameters  $f_x$  and  $f_y$  represent horizontal and vertical scale factors, respectively that combine the lens focal length with the physical size of the individual elements on the sensor array. The distortion matrix include the 8 distortion coefficients (D1 [8x1] and (D2 [8x1]) that correct the "fish eye" effect that some lens have. Summing up, the intrinsic camera parameters are used to link the pixel coordinates of an image with the corresponding coordinates in the camera reference frame [51].

On the other side, the extrinsic parameters define the location and orientation of the camera reference frame comparing to a world reference frame [51]:

- **Rotation matrix** - between the first and second camera coordinate system (R [3x3]);
- **Translation vector** between the coordinate systems of the cameras (T [3x1]);

- **Rectification transform** - (rotation matrix) for each camera ( $R1 [3 \times 3]$  and  $R2 [3 \times 3]$ );
- **Projection matrix** - in the new (rectified) coordinate systems for each camera ( $P1 [3 \times 4]$  and  $P2 [3 \times 4]$ );
- **Disparity-to-depth** - mapping matrix ( $Q [4 \times 4]$ ).

All these parameters are extremely important since the disparity map and the 3D reconstruction require to know the object coordinated with respect to the pixel position in both cameras.

In order to calibrate the camera, any characterized object can be used as a calibration target, as long as its 3-D world coordinates can be known in reference to the camera [33]. There are many different calibration methods like using a box covered with markers or a plane grid with symmetrical or asymmetrical grids. In this case it was chosen to use the chessboard method since it provides good calibration results and it is easy to implement. Thanks to this it is the most used method for camera calibration and there are many example codes, toolboxes and help guides available.

For this method it is only needed the stereo vision setup and a printed chessboard since their squares corners are very easy to find using computer vision algorithms. In order to find out the position of each corner it is necessary to know the number of horizontal and vertical squares in the chessboard and the size of each square (the coordinate system of the target). With this information the equations that contain the intrinsic parameters of each cameras can be obtained and, as a side benefit, the position and orientation of the camera, with respect to target (camera pose), are found (extrinsic parameters).

Since the objective is to detect pedestrians or objects at distances up to 50 meters, the calibration area must be representative of this so the calibration of the stereo cameras is outside the laboratory at multiple distances. Because of that it is needed to use a big sized chessboard and in this case with the size of 90x80cm and 9x7 white and black squares with 10.5cm width.

In order to obtain better results the chessboard already existing in LAR was improved. The frame was rebuilt and it is now much lighter than before, which makes it easier to hold the chessboard during calibration tests. Also, for this purpose, two pieces to hold the chessboard were placed. Another improvement that was made was to put styrofoam inside the frame so the printed chessboard sheet does not flow with the wind since the results are much better if the calibration pattern is affixed to a flat surface. The new chessboard used for the calibration tests is presented figure 5.1.

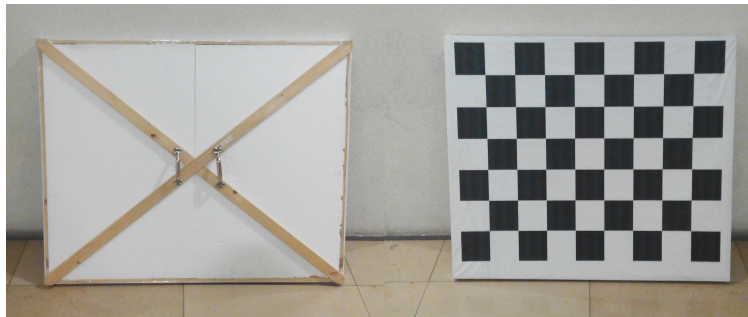


Figure 5.1: New Chessboard for stereo camera calibration.

For stereo calibration a new application was developed, based on an example available at the OpenCV library [51]. This program uses the images saved by the Stereo Camera Application and tries to detect the chessboard corners. Then it uses the function `stereoCalibrate` to calculate the extrinsic and intrinsic parameters and export them to an OpenCV format file that the stereo application will use after. In order to validate if the calibration results are satisfactory, the program shows some rectified images. However, this is only for a validation step since the Stereo Camera Application rectify the new images in real time. The ROS application was not used for stereo calibration because it is an interactive application which is not useful when calibrating cameras up to 50 meters.

## 5.2 User Instructions for Calibration

First of all, it is necessary to have the setup ready (as explained in figure 5.2). Then it was necessary to collect and save the images, for that it can be used the ROS package developed - `StereoCam`, with a specific launch file called `stereoCam_for_calibration.launch` or the application in non ROS environment - `pgrFlea3connect`. Both these programs will only connect both cameras, capture images and save them to a folder.

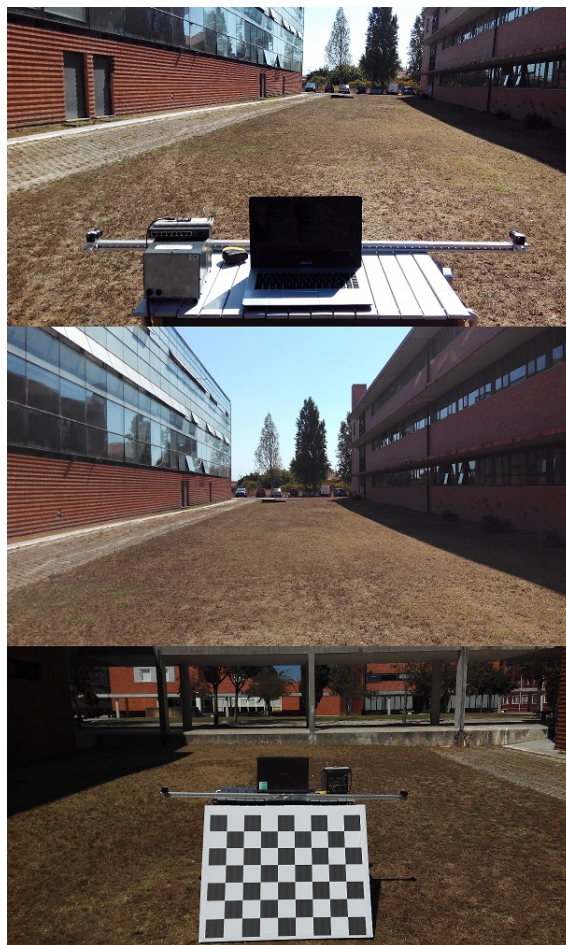


Figure 5.2: Calibration setup - first experiment.

To calibrate the stereo system, multiple pairs of images of the calibration pattern from different angles and positions and at different distances must be taken (in this case up to 50 meters). Another important thing is that the entire pattern must be visible in each image otherwise the pair of image cannot be used for calibration. Figure 5.3 presents some of the frames used to calibrate the cameras for the baseline of 0.3m and 8mm lens.

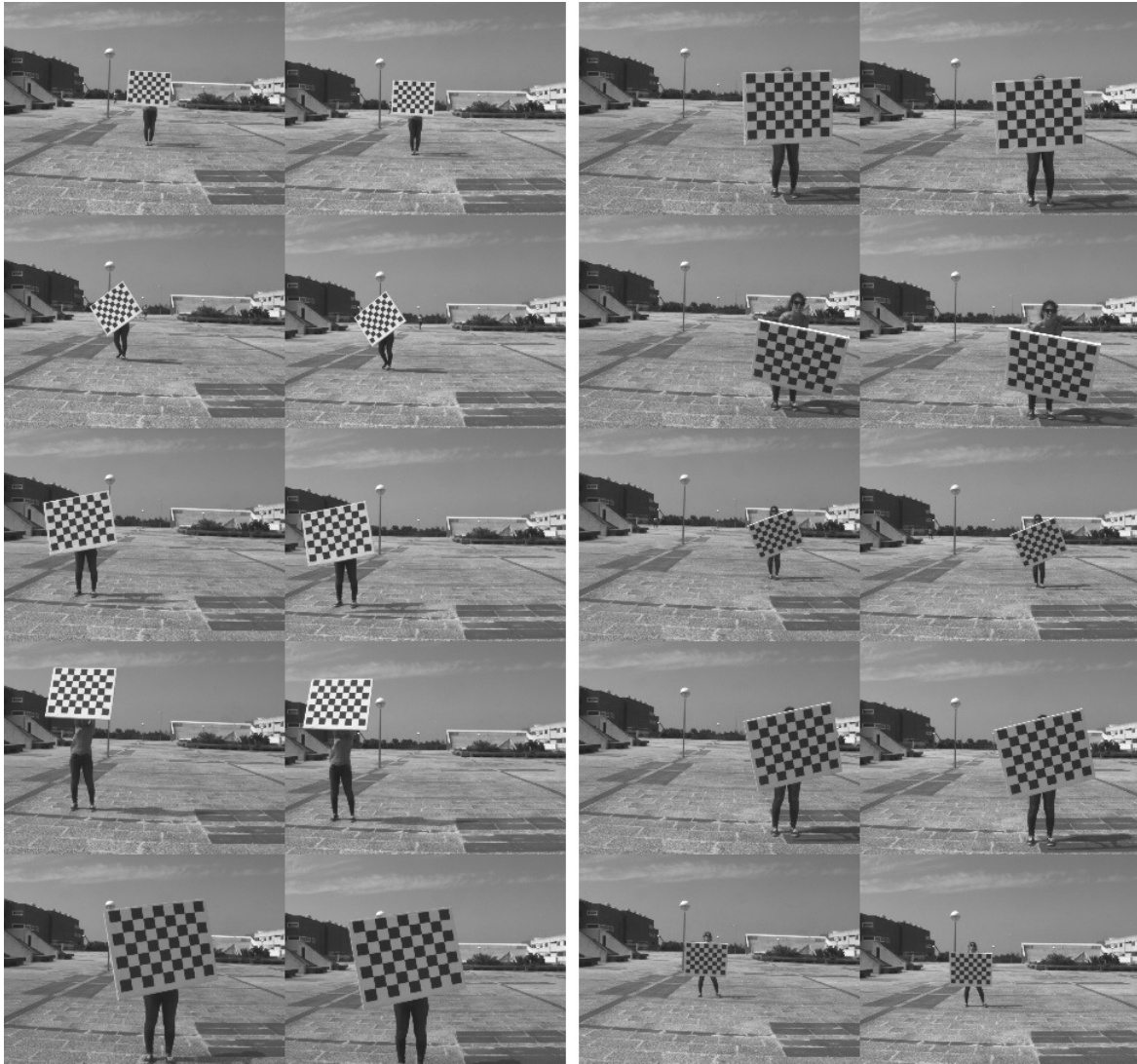


Figure 5.3: Examples of some of the frames used to calibrate the cameras (30cm baseline and 8mm lens).

After the data collection it is mandatory to actually calibrate the cameras and for that there is the software developed, `stereoCamCalibration`. This program require the path of the pictures that are considered for calibration and for data saving. In order to get good results it should be used several frames.

In the end, the program shows the images with the detected corners, the rectified images and the average re-projection error (RMS) presented in the next section. For better results, the average re-projection error must be as close to zero as possible, so it was defined that all

pictures with RMS bigger than 1 should be removed and re-run the calibration application with the remaining frames [60].

The calibration process must be repeated after any setup changes such as baseline, lens, focal aperture or others.

Finally, for the stereo camera application to run, either the one in ROS environment or the other, it is just needed to indicate the path of the calibration files to the chosen main program.

### 5.3 Results of Calibration Experiments

For each calibration test the user instructions for calibration, explained in section 5.2, were followed and more than 500 images was shot to choose about 100 of them in the end. All the frames must contain the complete chessboard and the chessboard must be in different orientations and at different positions and distances from the setup, as said before.

Two calibration tests was performed during the development of this study. Figure 5.2 presents the calibration setup of one of this tests. In the first test, the focus was on testing the camera driver and understand how the calibration works and what would influence the calibration results. For that, the stereo camera was calibrated for 13 baselines, each of them for the 2 lens available. From this first calibration test was concluded that the baseline and lens do not affect the result of the calibration but the number of photos with the chessboard in different positions and at different distances from the stereo camera do.

In the last test the cameras were calibrated for both lenses (8 and 16mm focal length) and for 5 different baselines that had been previously chosen for the study (chapter 6). In table 5.1, the average re-projection error, for the second calibration test, is presented.

Table 5.1: Stereo Calibration results for the baselines and lenses under study.

Baseline	(cm)	Lens	Average Reprojection Error (px)
B1	30	L8	0.172
B1	30	L16	0.279
B6	80	L8	0.171
B6	80	L16	0.3
B8	100	L8	0.199
B8	100	L16	0.293
B10	120	L8	0.268
B10	120	L16	0.254
B13	150	L8	0.242
B13	150	L16	0.254

As it can be seen in table 5.1, the maximum error is 0.3px and the medium value is 0.243px which represents a good calibration result ( $RMS \leq 1px$  [60]). Figure 5.4 shows an example of an original set of images, the same images with chessboard corners highlighted and the rectified and undistorted set of images. As it can be observed, in the two bottom images, the same chessboard corner is aligned in both images and the objects that are not curve in the

real world are not curve in the last two (corrected) images.

”With a calibrated system such as this, there is a known equation relating the disparity of the scenes capture by the two images of the calibrated system to depth of objects to the cameras” [31]. So it is time to study which baseline/lens are the best solution for pedestrian detection in the next chapter.

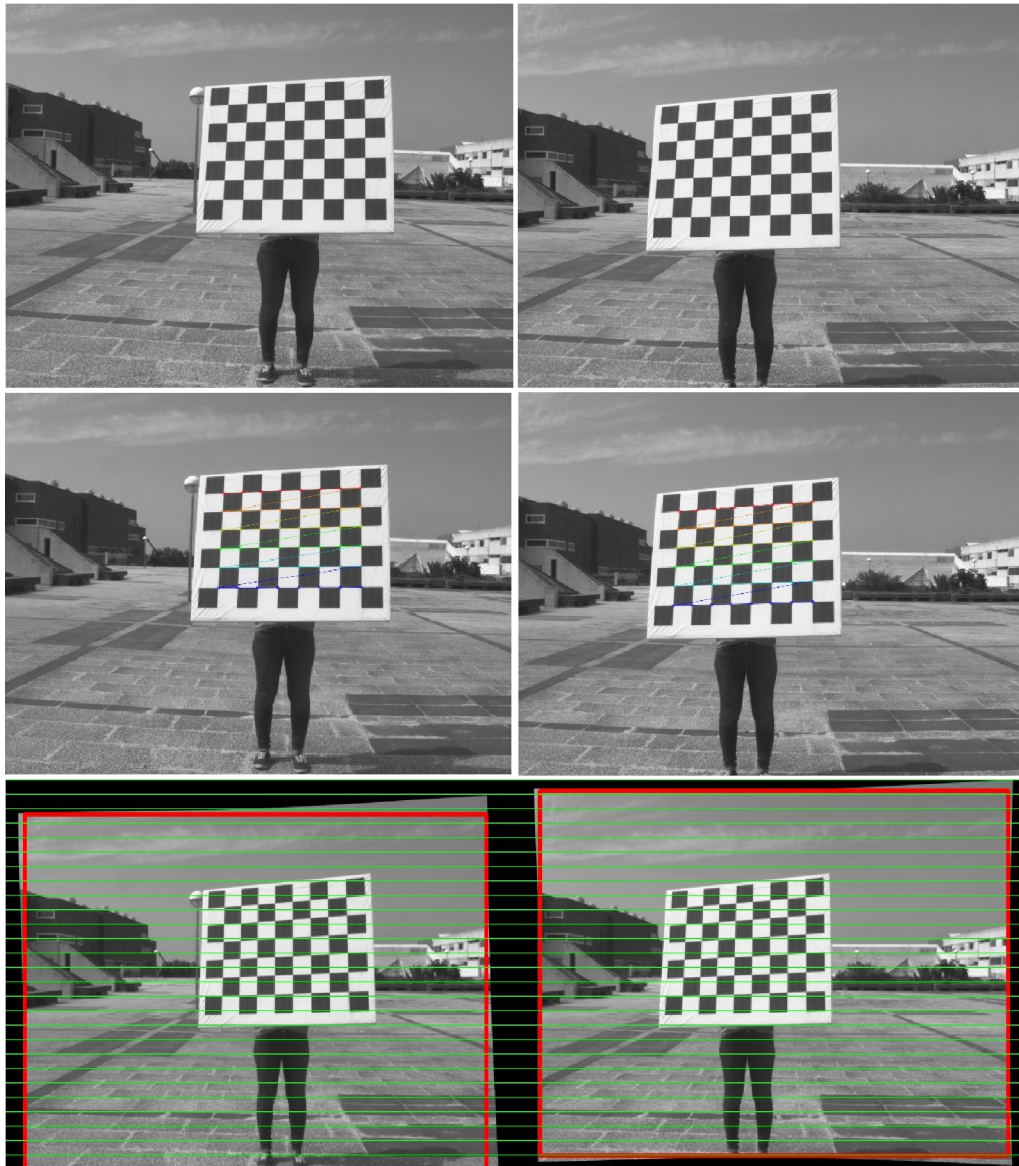


Figure 5.4: Example of the calibration process (0.3m baseline and 8mm lenses): Original acquired Images (top); Detect Chessboard Corners (middle); Compute calibration parameters and Rectify images(bottom).



## Chapter 6

# Experimental Results - Influence of the Baseline and Lens

To study the ideal stereo vision system for pedestrian detection there are many variables such as baseline, lens, calibration process and results, stereo correspondence algorithm, computational time cost and other factors related to quality/cost, which is very important for commercialization.

### 6.1 ADAS Context

In order to qualify and quantify the results of the stereo vision system it is necessary to think about an hypothetical operating procedure on an ADAS for pedestrian detection:

- The driver might not see the pedestrian - the reason why this ADAS is needed;
- The ADAS is always searching for pedestrians (the computational time cost should be around 1,5 seconds);
- If the ADAS detect a pedestrian or other target on the road it advises the driver;
- If the driver stops, okay;
- If the driver do not stop, the ADAS initiate an emergency brake system.

In this project it is only developed the application to detect pedestrians, so the three last steps will be possibly achieved in future projects. However, it must be considered, now, the global application since all this operating process requires time and this time translates in meters driven. So, the system must be capable of detecting pedestrians and other targets on time for the vehicle to be able to safely stop, with future projects.

On the website "Prevenção Rodoviária Portuguesa" [61] (Portuguese Road Safety) there is a simulation that shows how many meters a vehicle takes to stop after seeing and obstacle on the road - figure 6.1.

The simulator, on figure 6.1, takes in consideration the velocity of the car (35km/h in the figure), the reaction time, considered to be 1.5 seconds for computational process and driver reaction and weather (rain in the figure) and road paving. For this specific conditions, the

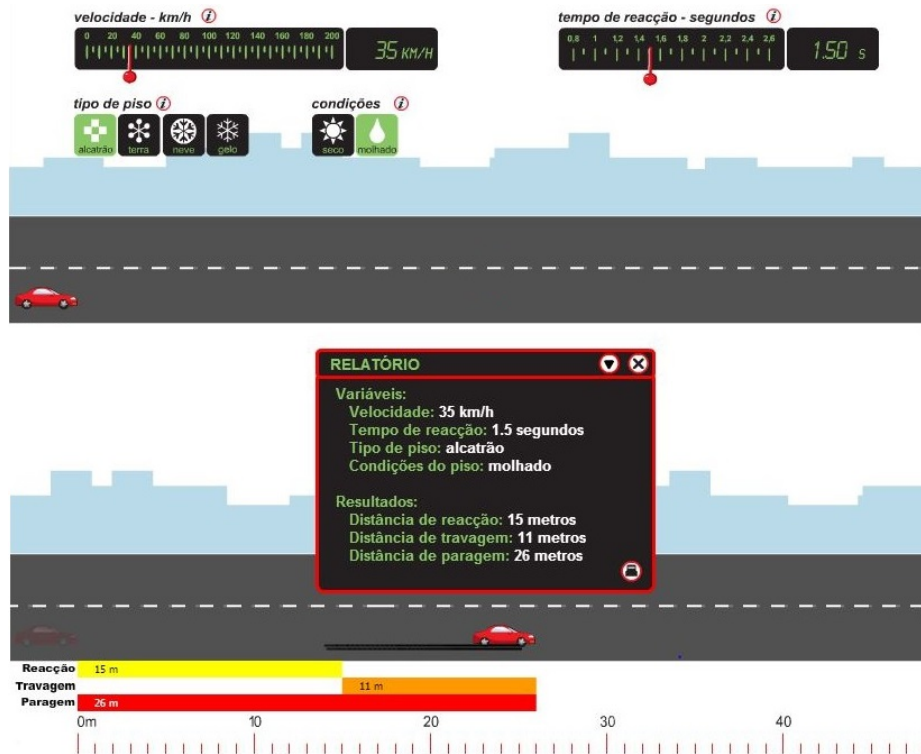


Figure 6.1: Simulator of the stopping distance [61].

simulator indicates that 26 meters are needed to safely stop the vehicle. Imagining that there is a pedestrian on the road in this conditions, the ADAS is useful if it detects the pedestrian at more than 26 meters, otherwise the car would not be capable of safely stop. Next it is presented the simulator results for the two urban velocity "extreme situations".

- Velocity: 50km/h = 13,89 m/s
- Stopping Distant (Rain): 43 m
- Stopping Distant (Sun): 32 m
- Velocity: 20km/h=5.55 m/s
- Stopping Distant (Rain): 11 m
- Stopping Distant (Sun): 10 m

After looking at the previous data, it is defined that an ADAS for pedestrian and object detection in urban scenarios must be capable of detecting pedestrians between 10 and 50 meters away from the vehicle.

## 6.2 Theoretical Study

Before start to capture frames and calculate stereo it is necessary to understand the possible influence of the baseline and the lens, based on theoretical equations and also what the field of view can influence as well.

### 6.2.1 Baseline and Lens

In order to estimate what would be the ideal baseline and lens for the distance gap (10 to 50 m) a theoretical study was performed based on the following equation for depth resolution, "the accuracy with which a stereo vision system can estimate changes in the depth of a surface" [62]:

$$\Delta Z = \frac{Z^2 S}{fB} [33] \quad (6.1)$$

wherein,  $\Delta Z$  is the depth Resolution (m),  $f$  is the focal length (m),  $B$  is the Baseline (m),  $Z$  is the distance from the camera (m) and finally  $S$  is the size of one pixel on the sensor (m) ( $3.69 \mu\text{m}$ , for the cameras used).

Figures 6.2 and 6.3 present two charts that show the influence of the distance of the obstacle with depth resolution for different baselines and for the two lens available for the study.

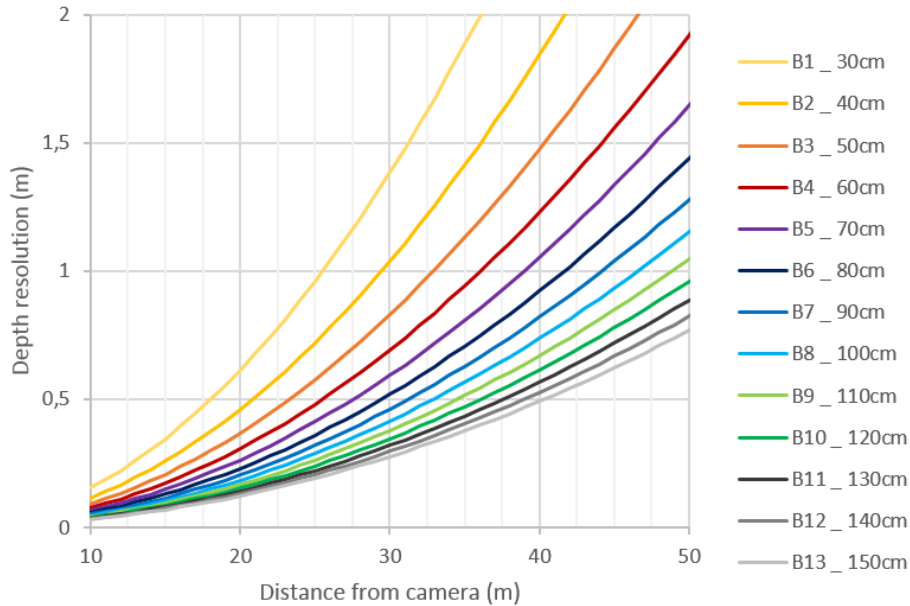


Figure 6.2: Influence of the distance of the obstacle with depth resolution - 8mm lens.

By analyzing the graphics before it can be understood that the higher the distance from the camera, the higher the depth resolution (the uncertainty of the measure) and that higher resolution can be achieved when using a wide baseline system for closer objects. It can also be verified that bigger focal lengths have lower depth resolutions when comparing to the same baseline and distance from the camera. The importance of depth resolution is that, for example, an object at 30m afar from the setup could be detected at 31.5m or at 28.5m when using a baseline of 0.3m and the 8mm lens, which could turn fatal. This is very important to analyze since our system must be the most exact possible. For this kind of system it is considered that an error larger than 1.5m is to high. Based on this last affirmation and on the charts presented, it seems that baselines less than 0.8m length are not adequate. However it is still necessary to perform the experimental tests.

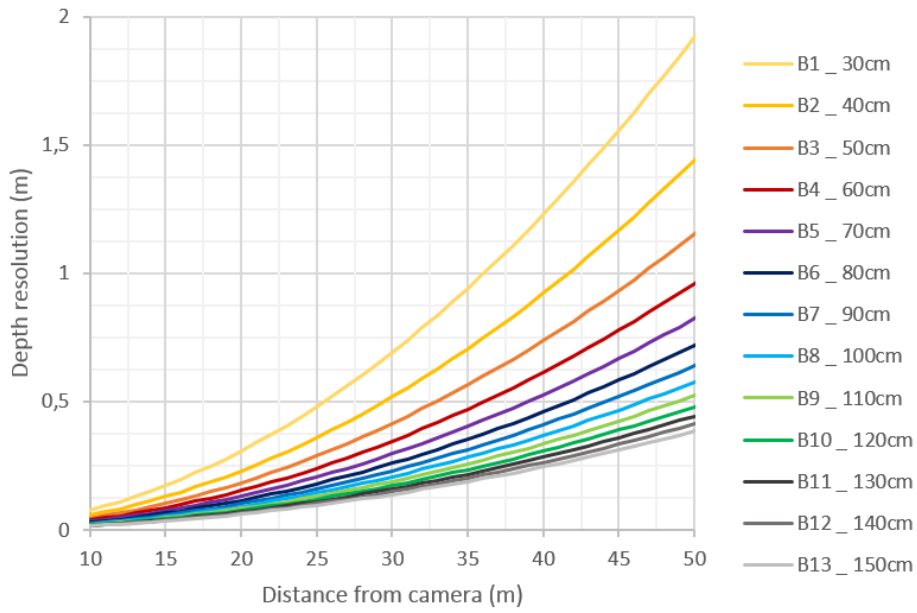


Figure 6.3: Influence of the distance of the obstacle with depth resolution - 16mm lens.

For the graphics in figures 6.4 and 6.5 another equation is used:

$$Z = \frac{fB}{dS} [33] \tag{6.2}$$

wherein,  $Z$  is the distance from the camera (m),  $f$  is the focal length (m),  $B$  is the Baseline (m),  $d$  is the disparity value (px) and  $S$  is the size of one pixel on the sensor (m/px) (3.69  $\mu$ m).

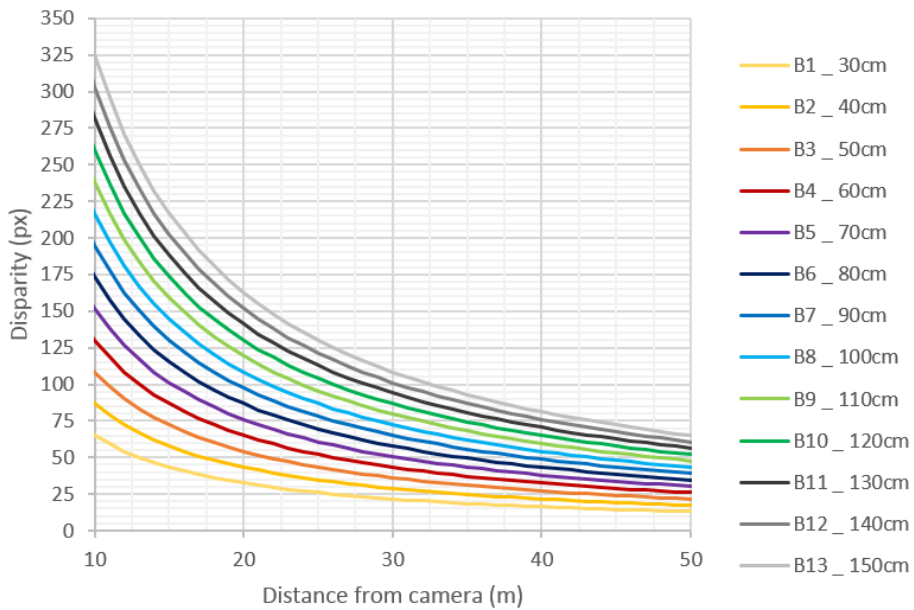


Figure 6.4: Influence of the disparity value with obstacle distance - 8mm lens.

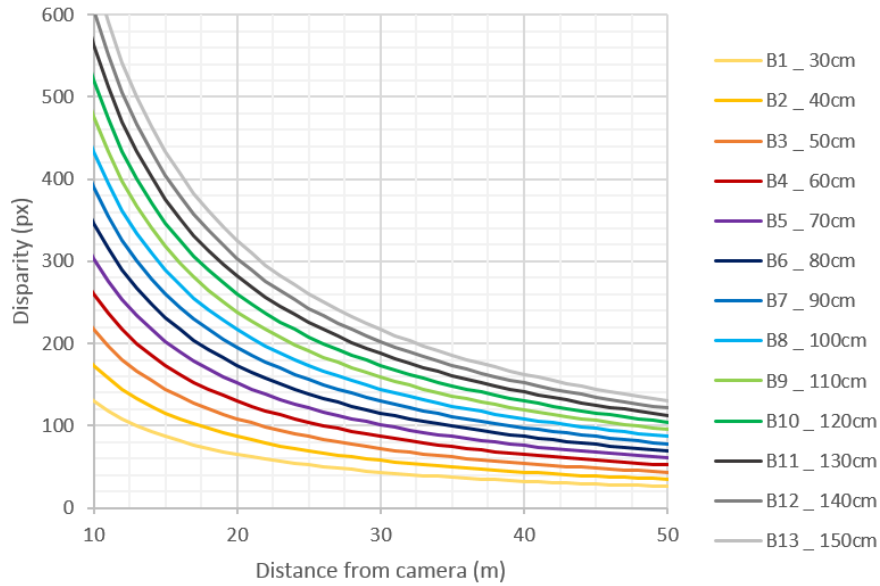


Figure 6.5: Influence of the disparity value with obstacle distance - 16mm lens.

Looking at the graphics in figures 6.4 and 6.5, it is shown that the bigger the baseline the higher the disparity values, for the same distance from the camera, as it is predictable. This means that the same object, at the same distance from the camera, is represented with higher disparity values, in the disparity map, as the baseline increases. For example, for a baseline of 0.3m and 8mm lens, an object at 10m from the camera is represented with a disparity value of 65px. In the same situation, with a baseline of 1.5m the disparity value is 325px. When comparing focal lengths, it can be noticed that an object detected at the same distance and with the same baseline have higher disparity values with the 16mm lens than with the 8mm.

If the disparity value is higher, the stereo image is probably more accurate, at a further range, because it means that the stereo system can distinguish better the objects more afar from the cameras [33]. However, if the objects are too close from the camera or if the baseline is too large it results in not overlapped objects between the images and therefore correlation suffers when viewing close objects. Decreasing the baseline between cameras fixes this problem, for closer objects, despite the loss of definition at far objects [33].

## 6.2.2 Field of View

Another important question to analyze is the FOV (Field Of View) in angles (equation 6.3) and the actual distance covered by image (equation 6.4) in meters.

$$FOV = 2 * atan\left(\frac{S * P}{2 * f}\right)[33] \quad (6.3)$$

$$distance \ covered = 2 * sin\left(\frac{FOV}{2}\right) * Z[33] \quad (6.4)$$

In equations 6.3 and 6.4, S is the size of one pixel (3.69  $\mu\text{m}$ , for this cameras), P is the number of active pixels (in this sensor - 1932H x 1452V [63]), f is the focal length of the cameras (m) and Z is the distance from the camera (m). For a better understanding, there is figure 6.6.

In order to calculate the FOV and the distance covered for horizontal and vertical alignment it is just necessary to consider the horizontal or the vertical values of P. In this case the horizontal FOV is the most important one so the distance covered is presented in table 6.1.

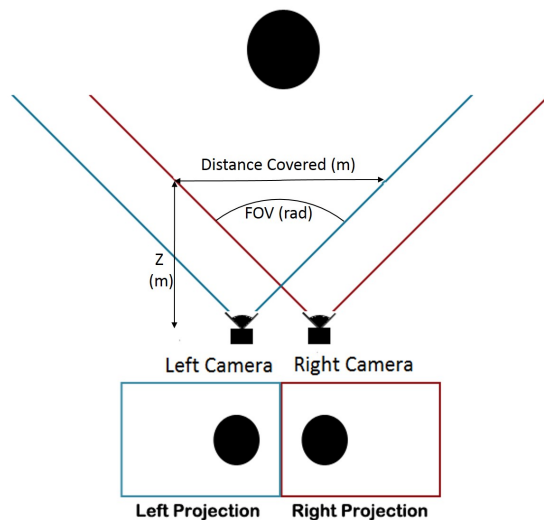


Figure 6.6: Field of View of a Stereo Vision System.

Table 6.1: FOV and Distance covered by the stereo vision system.

	Lens	
	8mm	16mm
Horizontal FOV (rad)	0.838	0.438
Distance Covered @ 10m (m)	8.140	4.349
Distance Covered @ 20m (m)	16.280	8.698
Distance Covered @ 30m (m)	24.420	13.047
Distance Covered @ 40m (m)	32.560	17.396
Distance Covered @ 50m (m)	40.699	21.745

As it can be seen by the equations 6.3 and 6.4 and table 6.1, the horizontal FOV and the distance covered does not depend on the baseline but only on the lens. For example, an object at 10m from the cameras can only be detected if it is inside the "horizontal range" at that distance (8.140m for 8mm lens and 4.349m for 16mm lens). This fact will influence the distances possible of study and maybe influence the choice of the ideal solution since this project consists in detecting pedestrians and the system must be able of detect them even if they are in the sidewalk.

After analyzing this theoretical data it was considered that the most interesting baselines for pedestrian detection in ADAS context are from 80 to 120 cm; wherein the 80, 100 and 1.2m baselines were chosen. It was considered that baselines less than 0.8m appear to result in depth resolutions too high for ADAS and baselines bigger than 1.2m turn the detection of objects at close range difficult or even impossible. However, it will also be studied the 0.3m and the 1.5m baselines to know the results at the limit of the system setup. Both lens influence will be studied.

### 6.3 Data Collection

In order to collect data to evaluate the influence of the different baselines and lens in the stereo vision system an experimental test was performed at Aveiro University (figure 6.7).



Figure 6.7: Data collection environment - open area at Aveiro University.

The main big concern about the testing environment was to be an open area with more than 50 meters wide. The objective of this experiment is to evaluate the precision of the target detection comparing to the real target distance for different baselines and lens. For this, some positions were measured and marked. The experimental scenario is not a crowd scene since the objective of this thesis is not to test pedestrian detection but to understand which baseline/lens would be better, i.e. more precise, for it.

To start the data collection the user must follow the user instructions for the application (in the Appendix) and the user instruction for calibration detailed (in section 5.2). The calibration must be repeated after every change of setup (baseline, lens, lens aperture or focus) and before starting collecting data. In order to work later on the data collected, bags containing the original acquired images and camera info were recorded using the "rosbag" package for the ROS app to be used and also images in png format for the non ROS application.

Figure 6.8 shows the positions measured and marked for data collection. The left and right positions are 2.5m apart from the middle (related to the system setup) - vertical alignment. The parallel positions are separated by 10 meters apart from the next one, from 10 to 50 meters - horizontal alignment. Figure 6.9 presents a photo montage of a pedestrian in every marked position. During the experiment, a pedestrian walks through every marked point. After the calculated distance of the pedestrian, using the Stereo Camera Application, will be compared with this real distances measured (10m, 20m, 30m, 40m and 50m).

In the next section the results of this study are presented.



Figure 6.8: Positions for data collection: Red @ 10m, Yellow @ 20m, Green @ 30m, Blue @ 40m, Purple @ 50m.



Figure 6.9: Photo montage of the positions for data collection.



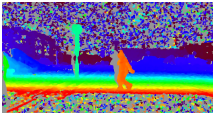

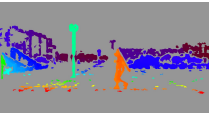
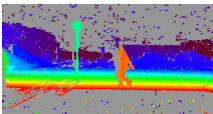
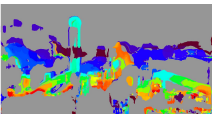
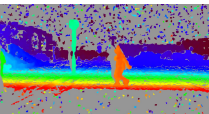
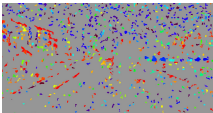
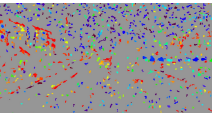
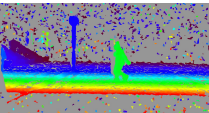
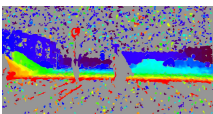
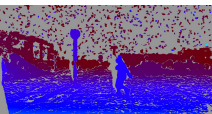
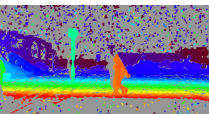
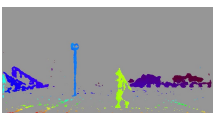
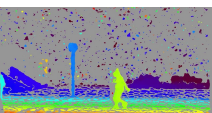
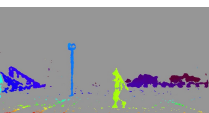
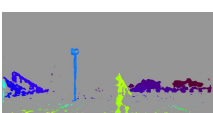
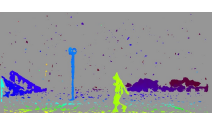
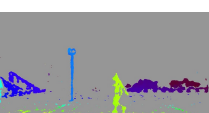
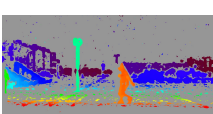
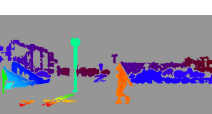
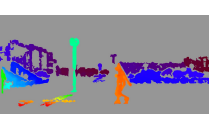

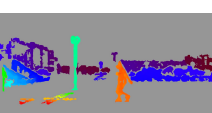
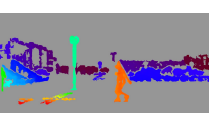
## 6.4 Experimental Results

### 6.4.1 SGBM Parameters

Using the rosbags and the frames previously saved, and after running the calibration program, it is now necessary to try to find the best parameters for the stereo correspondence algorithm. For that, the `dynamic_reconfigure` and the `stereo_image_proc` package were used.

The search for the "ideal" parameters is a subjective process since there is no objective evaluation at all. Table 6.2 presents the influence of the SGBM parameters.

Table 6.2: Qualitative analysis of the influence of the SGBM parameter is values.

Parameter	Lower Value	Larger Value	Chosen Value
Uniqueness Ratio			
SAD Window Size			
Min Disparity			
Number Of Disparities			
P1			
P2			
Speckle Window Size			
Speckle Range			
Pre Filter Cap disp12MaxDiff Full DP	no significant influence on the disparity map		

Starting from the first parameter on table 6.2, **Uniqueness Ratio**: if the value is too high there is almost nothing in the disparity image because the majority of "points", in the disparity image, are interpreted as no correct match. If the value is too low there is much noise in the figure. However, by increasing this value until a certain number removes repeating pattern from the disparity image such as ground, sky, walls, ceiling, ..., for that lower values between 40 and 10 (from 0 to 1000) were chosen.

Looking at the effect of the **SAD Window Size** parameter, if the value is too high the block matching size is too big and the disparity map gets distorted. Because of that, lower values were chosen in order to smooth the image - values chosen between 20 and 13 (from 5 to 255).

The next parameter is **Min Disparity** - this parameter allows values from -128 to 128 (pixels) and normally it is defined 0. However, if the parameter value is close to extreme values the disparity image is only random noise. Basically, if it is defined with a negative value close to 0 the disparity map can show far away objects, instead, when it is defined with positive values, close to 0, far away objects do not appear but closer objects from the camera do. Since the range of interest of this thesis is from 10 to 50 meters this parameter was defined with values between 10 and 109 because, within this values, the disparity map can contain objects inside the range defined for object detection and still removes other objects outside the range (such as buildings).

The **Number of Disparities** affects the precision of the disparity map. The bigger the value the better the disparity at close range, however it loses precision for far away objects but if it is too low the detection of close objects is impossible. Because of this for this parameter were chosen values between 64 and 256 (from 32 to 256).

The influence on stereo of the next 2 parameters are very similar. The higher the value the higher the smooth of planar surfaces but also the more noise in planar surfaces (such as sky or ground). Since the ground was already "removed" by increasing the uniqueness ratio the lower value possible for this 2 parameters - 0 (from 0 to 4000) was chosen. **P2** has smaller influence when comparing to **P1**.

The **Speckle Window Size** parameter functions like an isolated points filter and because of that the higher value possible (1000 was chosen, for all baselines). The **Speckle Range** has no significant influence however, if it is 0, there is nothing in the disparity map so it was defined as 1 for all baselines as well. The effect of the parameter Speckle Window Size is very similar to this one, however, less impactful.

About the 3 last parameters (**Pre Filter Cap**, **disp12MaxDiff** and **Full DP**), it was not noticed a significant difference, in the chosen scenario, so the chosen values for the Pre Filter Cap was 1 and for the others the value was 0.

The ideal disparity map for this work was considered to be something like it is presented on figure 6.10. It was tried to decrease noise in the image and to remove the ground and the sky since the objective is to detect obstacles from 10 to 50 meters. Also, this way computational time is saved from the point cloud processing because there is less points to process.

In tables 6.3 and 6.4 are shown the final SGBM parameters used to evaluate the influence of the baseline and the focal lens in object detection. It is important to say that the values do not follow any rule since some of them affect others and the parameters might be needed to be altered for different scenarios.

After the SGBM parameters have been defined the study concentrated on how to evaluate the stereo system.

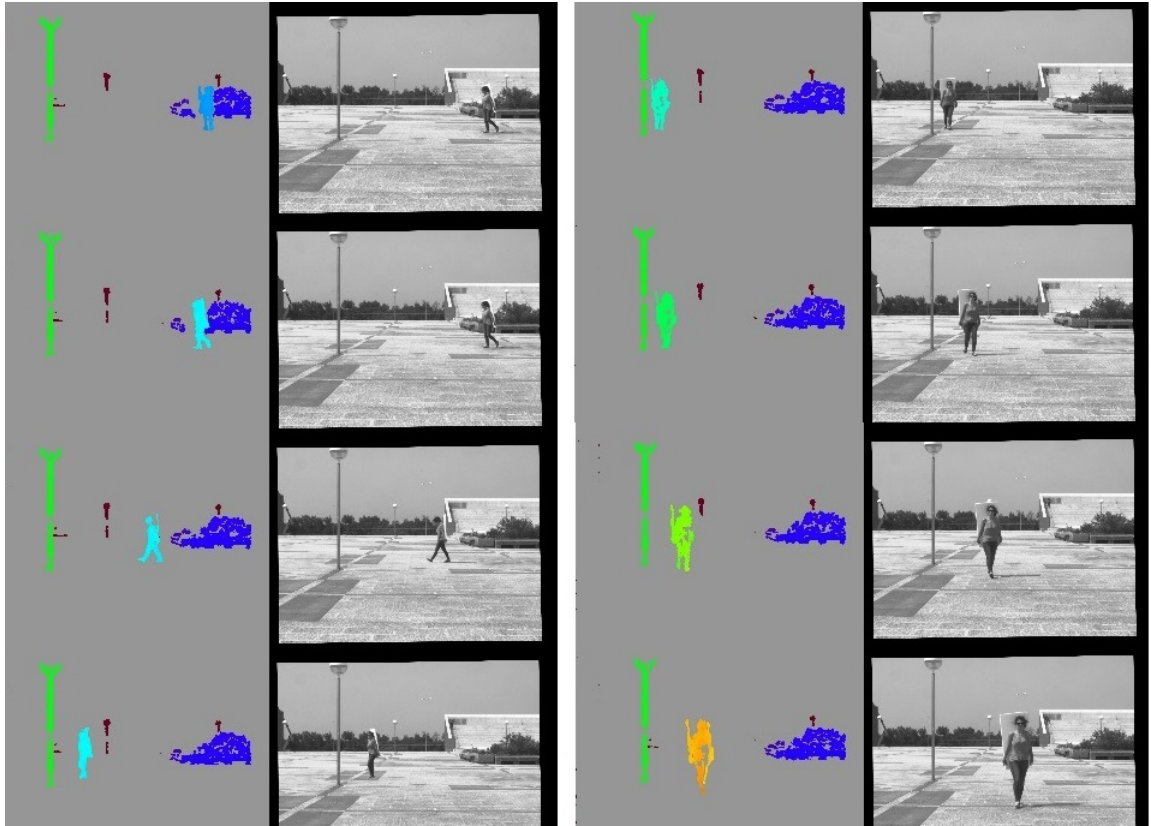


Figure 6.10: Example of the a disparity map considered for this ADAS system.

Table 6.3: SGBM Parameters - 8mm Lens.

SGBM Parameters	BASELINE (mm)				
	30	80	100	120	150
Pre Filter Cap	1	1	1	1	1
Uniqueness Ratio	40	16	10	14	10
SAD Window Size	19	19	19	19	13
Min Disparity	10	29	35	43	52
Number Of Disparities	64	144	192	256	256
Speckle Window Size	1000	1000	1000	1000	1000
Speckle Range	1	1	1	1	1
P1	0	0	0	0	0
P2	0	0	0	0	0
disp12MaxDiff	0	0	0	0	0
Full DP	0	0	0	0	0

Table 6.4: SGBM Parameters - 16mm Lens.

SGBM Parameters	BASELINE (mm)				
	30	80	100	120	150
Pre Filter Cap	1	1	1	1	1
Uniqueness Ratio	35	27	27	27	27
SAD Window Size	17	17	17	17	20
Min Disparity	24	46	75	75	109
Number Of Disparities	112	272	192	256	256
Speckle Window Size	1000	1000	1000	1000	1000
Speckle Range	1	1	1	1	1
P1	0	0	0	0	0
P2	0	0	0	0	0
disp12MaxDiff	0	0	0	0	0
Full DP	0	0	0	0	0

#### 6.4.2 Evaluation of the Stereo Vision System

The evaluation of this stereo vision system consisted of comparing the real distance (measured with a common tape measure) and the distance automatically calculated using the Stereo Camera Application.

As explained before in 6.3, the experimental test consisted of a pedestrian walking through multiple positions previously marked in the ground, as presented in figure 6.9. During the test, frames and bags were saved in order to later calibrate the system and study the influence of baseline and lens in the precision of the detection by comparing the real distance measured with the Z coordinate of the centroid's clusters of each target found by the application. Photos for calibration were taken after each change in the stereo vision system such as baseline and lens.

After running the program for all baselines and lens (0.3m, 0.8m, 1.0m, 1.2m and 1.5m baselines and 8mm and 16mm lens) a text file was created with all the Z coordinates of each cluster (pedestrians or light pole). The results are shown in tables 6.5 (for 8mm lens) and 6.6 (for 16mm lens). In those tables there is presented the calculated distance, the difference between the calculated distance and the measured one, the percentage of error when comparing to the previously measured distance and the theoretical resolution calculated using the equation 6.1. The distance from the cameras of the light pole was 16.2m and this is the only fixed object detected.

As seen in table 6.5 it is not possible to detect objects at 10m from the cameras using a baseline of 1.5m and the 8mm lens. When using the 16mm lens it is not possible to detect objects at 10m, with any baseline. Actually the 16mm lens, for the 1.2m and 1.5m baselines the application cannot even detect objects at 16.2m and for the 1.5m baseline the minimum distance to detect objects is 30m, as it can be seen in table 6.6.

Comparing the percentage of error, for the same baseline and lens, as the distance of the object from the camera increases the theoretical resolution increases. However, the results do not show that (in both tables), in fact, the percentage of error increases and decreases as the distance of the object gets bigger.

Looking at the percentage of error for different baselines but for the same lens, it can be noticed that the theoretical resolution decreases as the baseline increases. However, once

more, the results do not show any relation since in the major cases the percentage of error increases and decreases as the baseline increases.

Comparing both tables (6.5 and 6.6) and the percentage of error for the 8mm lens and the 16mm the theoretical resolution presents lower values for the 16mm lens yet, the percentage of error calculated is bigger for the 16mm lens which contradicts, once more, the theoretical studies in section 6.2.1.

Analyzing the difference of the calculated distance it is observed that for the 0.3m baseline, independently the lens, the difference is lower than the resolution. For the other studied baselines, when using the 16mm lens, the calculated difference is higher than the resolution, with the exception of the 1.5m baseline. For the other studied baselines, but using the 8mm lens, the calculated difference is usually lower than the resolution or at least close to it.

Looking at the results from an ADAS point of view and since it was considered in section 6.2.1 that differences between real and calculated distances higher than 1.5m were too high for this kind of application. Thanks to this, the 1.0m and 1.2m baselines, using 16mm lens, exceed this limit for any distance.

For objects at 10m, 16.2m and 40m, the 0.300m baseline using the 8mm lens shown the lower error. At 20m and 50m the most exact baseline/lens was the 120m baseline with the 8mm lens. For objects at 30m the better results were using the 16mm lens. Comparing both lens, the studied baselines using the 8mm lens proved to achieved better results globally.

Table 6.5: Result is analysis of the Calculated Distance Vs. Real Distance - 8mm lens.

	Baseline (m)	Real Distance (m)					
		10	16.2	20	30	40	50
<b>0.300</b>	Estimated Distance (m)	10.038	16.214	20.174	30.207	40.106	49.751
	Difference (m)	0.038	0.014	0.174	0.207	0.106	0.249
	% error	0.380	0.086	0.870	0.690	0.265	0.498
	Resolution (m)	0.154	0.394	0.615	1.384	2.460	3.844
<b>0.800</b>	Estimated Distance (m)	9.913	16.177	19.758	29.948	39.562	50.631
	Difference (m)	0.087	0.023	0.242	0.052	0.438	0.631
	% error	0.870	0.142	1.210	0.173	1.095	1.262
	Resolution (m)	0.058	0.148	0.231	0.519	0.923	1.441
<b>1.000</b>	Estimated Distance (m)	10.126	16.534	20.441	30.505	40.583	50.987
	Difference (m)	0.126	0.334	0.441	0.505	0.583	0.987
	% error	1.260	2.062	2.205	1.683	1.458	1.974
	Resolution (m)	0.046	0.118	0.185	0.415	0.738	1.153
<b>1.200</b>	Estimated Distance (m)	10.128	16.310	20.030	30.093	40.249	50.043
	Difference (m)	0.128	0.110	0.030	0.093	0.249	0.043
	% error	1.280	0.679	0.150	0.310	0.623	0.086
	Resolution (m)	0.037	0.098	0.154	0.346	0.615	0.961
<b>1.500</b>	Estimated Distance (m)		16.115	19.905	29.695	39.700	49.359
	Difference (m)		0.085	0.095	0.305	0.300	0.641
	% error		0.525	0.475	1.017	0.750	1.282
	Resolution (m)		0.079	0.123	0.277	0.492	0.769

Table 6.6: Result's analysis of the Calculated Distance Vs. Real Distance - 16mm lens.

Baseline (m)		Real Distance (m)				
		10	16.2	20	30	40
<b>0.3</b>	Estimated Distance (m)	16.483	20.233	29.979	40.189	49.433
	Difference (m)	0.283	0.233	0.021	0.189	0.567
	% error	1.747	1.165	0.070	0.473	1.134
	Resolution (m)	0.197	0.308	0.692	1.230	1.922
<b>0.8</b>	Estimated Distance (m)	17.019	20.968	31.215	41.574	52.133
	Difference (m)	0.819	0.968	1.215	1.574	2.133
	% error	5.056	4.840	4.050	3.935	4.266
	Resolution (m)	0.074	0.115	0.259	0.461	0.721
<b>1.0</b>	Estimated Distance (m)	17.936	21.658	32.193	42.808	52.001
	Difference (m)	1.736	1.658	2.193	2.808	2.001
	% error	10.716	8.290	7.310	7.020	4.002
	Resolution (m)	0.059	0.092	0.208	0.369	0.577
<b>1.2</b>	Estimated Distance (m)		21.428	32.184	42.653	53.483
	Difference (m)		1.428	2.184	2.653	3.483
	% error		7.140	7.280	6.633	6.966
	Resolution (m)		0.077	0.173	0.308	0.480
<b>1.5</b>	Estimated Distance (m)			30.689	40.202	50.070
	Difference (m)			0.689	0.202	0.070
	% error			2.297	0.505	0.140
	Resolution (m)			0.138	0.246	0.384

## Chapter 7

# Conclusion and Future Work

Two new and fully integrated stereo camera driver were developed and successfully implemented in ROS environment and in non ROS environment. The Stereo Camera Application connects and synchronize two Point Grey Flea 3 GigE cameras, captures synchronized frames, correct them (using the files resulting from the calibration process) and finally creates the disparity image and the correspondent point cloud. In the end, the Z coordinate (distance from the camera) of the centroid of the objects found inside the cameras range is obtained. Also, a structure to mechanically support and assemble the stereo vision system was designed, developed and assembled.

Applying the user instructions for the stereo camera application (in non ROS environment) and the user instructions for stereo camera calibration objects can be detect and it is possible to know how far they are from the cameras. This way, if the cameras fixing system was assembled in the ATLASCAR pedestrian and object can be detected when driving.

However the major objective of this work, besides developing the driver and the fixing system, was to study the influence of baseline and lens on the precision of pedestrian and other target detection. In order to achieve this goal, synchronized frames from the stereo vision system were recorded to calibrate the system and for the study. The frames for the study were taken with a pedestrian in previously marked positions (10m, 20m, 30m, 40m and 50m) - that corresponded to the limit distances considered safe for a car to stop when driving between 20 and 50km/h (common velocity in urban scenarios). Since the measure of this distances were taken with a normal tape measure the measure has some uncertainty associated with as well as the pedestrian positions since a person does not walk exactly on the measured mark. For ground truth a light pole in the scenario was considered as a fixed object.

Looking at the results in tables 6.5 and 6.6 it can be conclude that it is possible to detect objects on road, though this results contradicts the theoretical studies.

For an ADAS application, the results show that the most exact detection is achieved using a 0.3m baseline with the 8mm lens, i.e. using the 8mm lens the calculated Z coordinates of the objects are closer to the real distance measured. However, it had been predicted that the 16mm lens would be ideal and also a baseline about 1.0m.

One reason for this could be the fact that when using the 16mm lens, the distance of the objects in the camera CCD sensor is lower than when using the 8mm lens which induce loss of resolution and can potentially increase error in the calculated distance.

Other possible reason for the difference between what was expected and the results col-

lected is that the stereo camera depth estimates are affected by alignment and calibration errors of stereo camera parameters since that the calibration process is never ideal and the theoretical equations does not consider this factors. However the calibration results (table 5.1) - evaluated by the average re-projection error are very good since the ideal average re-projection error is lower than 1px and the higher value is 0.3px [60].

From another point of view, the SGBM parameters are subjective hence using another parameter values the results could be different, but the change would not be significant. Also, the disparity accuracy depends on the matching accuracy of the stereo correspondence algorithm and the one that was used (SGBM) has some limitations and associated errors.

Nevertheless, the pedestrian, as a person, does not walk precisely at the defined distances which translates in an random associated error. However, looking at the error associated with the calculated distance for the light pole (the distance of a fixed object) it is proved that the random error associated with the fact that the pedestrian position is not exact is not the cause of the contradictory results. Because of this it can also be concluded that the theoretical study is not enough for evaluating the depth resolution of practical stereo camera measurement systems as it is seen in the experimental part of this study.

Although the results being different than it was predicted it can be said that this stereo system can detect objects, with reasonably precision, using the 8mm lens, for any baseline, and the 16mm lens for a 0.3m baseline. According to the results, other baselines with the 16mm lens should not be used because the limit of 1.5m error is exceeded and for baselines bigger than 1.2m the detecting pedestrians closer to 20m from the cameras which is not possible, which is not adequate to this kind of application.

Looking at the Bosch Stereo Video Camera (in table 2.2) this conclusion makes sense. With a baseline of 12cm this camera can detect objects in an ADAS context and its integrated in an emergency braking system in Land Rover Discovery Sport. This proves that nowadays maybe it is not necessary to have a huge baseline stereo camera for object detection in an ADAS context which translates in more portable solutions. For example, in some years from now, with a simple modification on cars, they could have this same equipment in the front glass, like a "Via Verde" device.



## 7.1 Future Work

For future work it would be interesting to repeat the experimental study but, instead of measuring the real distance with a normal tape measure, a laser could be used to measure the precise distance of the objects. This way the error associated with the measurement of the real distance would be severely reduced and the study could be more precise.

Focusing on a stereo vision system with only two cameras, like the one presented in this project, a dynamic fixing system with variable baseline could be designed and developed in order to adjust the baseline depending on car velocity. However, the results of this thesis show that the baseline does not influence much the precision of the stereo, yet, new and more deep studies could prove the opposite.

Concerning the Stereo Camera Application the point cloud visualization and processing could be integrated in the ROS application, that was not completed in this project. This upgrade would facilitate the use of the Stereo Camera Application in the ATLAS project. Another software improvement could be the optimization of the drivers because despite it captures frames at 15Hz, the processing time cost slows the application which reduces the real stereo frequency to 1 frame every 1.5s. The use of a more advanced processing equipment would make a huge difference in the processing time cost. Another possible solution could be the capture of lower resolution frames: for example, using half the resolution that was used reduces the time needed to create the disparity map in about 2.5 times.

In order to improve the quality of the disparity map and consequently the detection of objects another stereo correspondence algorithm could be tested or even developed for this specific purpose. The same could be done with the point cloud processing since it could be tried to use other algorithms, specifically developed for human detection, and other filters. This thesis did not focus only on the detection of humans but also other targets since, in real urban scenarios, there are more obstacles besides pedestrians. In the future it would be interesting if an evolution of this system in the new ATLAS series could be seen detecting obstacles and pedestrians and alerting or even stop the car if the driver does not react to the advise.

# References

- [1] Autoridade Nacional Segurança Rodoviária. Relatório Anual - Ano 2014 - Vítimas a 30 dias. [serial on the Internet]. 2014 [cited September 20 2015]. Available from: [http://www.ansr.pt/Estatisticas/RelatoriosDeSinistralidade/Documents/2014/RELAT%C3%93RIO%20ANUAL%20-%20V%C3%8DTIMAS%20A%2030%20DIAS/Rel\\_2014\\_30dias.pdf](http://www.ansr.pt/Estatisticas/RelatoriosDeSinistralidade/Documents/2014/RELAT%C3%93RIO%20ANUAL%20-%20V%C3%8DTIMAS%20A%2030%20DIAS/Rel_2014_30dias.pdf)
- [2] Talhada, T.: "Perceção 3D utilizando uma câmara stereo", Master's Thesis, Universidade de Aveiro (2012).
- [3] Bertozzi, M., Binelli, E., Broggi, a., & Rose, M. D. Stereo Vision-based approaches for Pedestrian Detection. Computer Vision and Pattern Recognition - Workshops, 2005 - CVPR Workshops - IEEE Computer Society Conference. 2005 [cited May 9 2015]. Available from: <http://doi.org/10.1109/CVPR.2005.534>
- [4] Santos, V. and all. ATLASCAR-technologies for a computer assisted driving system on board a common automobile; Intelligent Transportation Systems (ITSC) - 2010 13th International IEEE Conference on. 2010 [cited July 16 2015]; pp1421-1427.
- [5] Silva, D.: "Detecção e Segmentação de Alvos em Ambiente de Estrada usando LIDAR", Master's Thesis, Universidade de Aveiro (2013).
- [6] Silva, P.: "Visual Pedestrian Detection using Integral Channels for ADAS", Master's Thesis, Universidade de Aveiro (2013).
- [7] Azevedo, R.: "Sensor Fusion of LASER and Vision in Active Pedestrian Detection", Master's Thesis, Universidade de Aveiro (2014).
- [8] MBP Automóveis Portugal S.A.. iMieEV Ligue-se ao Futuro [Image]. [updated 2015; cited July 16 2015]. Available from: [http://www.mitsubishi-motors.pt/uploadedImages/Contents/Models/i-MiEV\\_MY12/i-MiEV\\_MY12i-MiEV/360\\_View/360-view/i-MiEV%20MY15\\_0f4aa833-aa41-40a3-aa54-c35304bce685\\_6dab5e75-54b2-46bc-bc2b-728bcb18e7ec\\_2070\\_01.png](http://www.mitsubishi-motors.pt/uploadedImages/Contents/Models/i-MiEV_MY12/i-MiEV_MY12i-MiEV/360_View/360-view/i-MiEV%20MY15_0f4aa833-aa41-40a3-aa54-c35304bce685_6dab5e75-54b2-46bc-bc2b-728bcb18e7ec_2070_01.png)
- [9] O'Toole, R. Gridlock: Why We're Stuck in Traffic and What to Do About It. The Journal of transport and Land Use. 2010 4(1) pp 83-85. Available from: <https://www.jtlu.org/index.php/jtlu/article/viewFile/199/159>
- [10] Bertozzi, M., Broggi, A., Fascioli, A. The Evolution of Vision-based Unmanned Ground Vehicles: the VisLab Perspective. [cited May 9 2015] Available from: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.73.7759&rep=rep1&type=pdf>

- [11] Google. Google Self-Driving Car Project. [Web Page] [cited January 31 2016] Available from: <https://www.google.com/selfdrivingcar/>
- [12] Universitat Munchen. VaMoRs. [Image] [cited July 16 2015] Available from: [http://www.unibw.de/lrt8/institut/mitarbeiter/prof\\_wuensche/vamors/pic1\\_preview](http://www.unibw.de/lrt8/institut/mitarbeiter/prof_wuensche/vamors/pic1_preview)
- [13] Carnegie Mellon University - School of Computer Science. Navlab 5 [Image] [cited July 16 2015] Available from: [http://www.cs.cmu.edu/staszal/stuff/robothalloffame2007/Navlab5/navlab5\\_2.jpg](http://www.cs.cmu.edu/staszal/stuff/robothalloffame2007/Navlab5/navlab5_2.jpg)
- [14] ParkShuttle Pilot Project. [Image] [updated August 07 2009; cited July 16 2015]. Available from: <http://faculty.washington.edu/jbs/itrans/parkshut.htm>
- [15] Urmsen, C. & all. Autonomous Driving in Urban Environments: Boss and the Urban Challenge. *Journal of Field Robotics*. 2008 [cited July 16 2015]; 25(8):425-466. Available from: <http://www.interscience.wiley.com>
- [16] Broggi, A., Cappalunga, A., Caraffi, C., Cattani, S., Ghidoni, S., Grisleri, P., ... Beck, J. The passive sensing suite of the TerraMax autonomous vehicle. *IEEE Intelligent Vehicles Symposium*. 2008 [cited July 16 2015]; pp769-774. Available from: <http://doi.org/10.1109/IVS.2008.4621208>
- [17] AutoCarHire Travellers Blog. Google's autonomous vehicle got its driving license. [Image] [updated May 11 2012; cited 16 July 2015]. Available from: <http://www.autocarhire.com/travelblog/wp-content/uploads/2012/05/Google-car.jpg>
- [18] Geiger, A., Lenz, P., & Urtasun, R. Are we ready for autonomous driving? The KITTI Vision Benchmark Suite. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2012 [cited July 16 2015]; pp3354-3361. Available from: <http://doi.org/10.1109/CVPR.2012.6248074>
- [19] Grisleri, P., & Fedriga, I. The BRAiVE Autonomous Ground Vehicle Platform. *Intelligent Autonomous Vehicles*. Lecce, Italy: International Federation of Automatic Control; 2010 [cited July 16 2015]; 7(1) pp497-502. Available from: <http://doi.org/10.3182/20100906-3-IT-2019.00086>
- [20] Bosch. Bosch in Portugal - Mobility Solutions [Web Page] [updated June 26 2015; cited August 20 2015]. Available from: [http://www.bosch.pt/en/pt/newsroom\\_11/news\\_10/news-detail-page\\_69312.php](http://www.bosch.pt/en/pt/newsroom_11/news_10/news-detail-page_69312.php)
- [21] Fossati, A., Schonmann, P., and Fua, P. (2011). Real-time Vehicle Tracking for Driving Assistance. *Machine Vision and Applications*. 2009 [cited July 16 2015]; 22(2) pp439-448. Available from: <http://infoscience.epfl.ch/record/149699/files/FossatiSF10.pdf>
- [22] Reuters. Exclusive: In boost to self-driving cars, U.S. tells Google computers can qualify as drivers. [Web Page] [updated February 10 2016; cited February 14 2016]. Available from: <http://www.reuters.com/article/us-alphabet-autos-selfdriving-exclusive-idUSKCN0VJ00H>
- [23] Shapiro, L., Stockman, G. *Computer Vision*. Prentice Hall. (2000). [cited July 16 2015]

- [24] Williamson, T. "A high-performance stereo vision system for obstacle detection". PhD Thesis, Robotics Institute Carnegie Mellon University Pittsburgh, PA 15213. (1998). Available from: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.39.736&rep=rep1&type=pdf>
- [25] Optometrist Network. [Image] [updated May 11 2012; cited 16 July 2015]. Available from: <http://www.vision3d.com/images/bb.jpeg>
- [26] Dalal, N., Trigg, B.. Histograms of oriented gradients for human detection. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). 2005 [cited July 16 2015]; pp886-893. Available from: <https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>
- [27] Papageorgiou, C., Poggio, T.. A Trainable Pedestrian Detection system. International Journal of Computer Vision (IJCV). 2000 [cited July 16 2015]; pp 15-33. Available from: <http://www.ai.mit.edu/courses/6.899/papers/icip99.published.pdf>
- [28] Mikolajczyk, K. and Schmid, C. and Zisserman, A. Human detection based on a probabilistic assembly of robust part detectors. The European Conference on Computer Vision (ECCV). 2005 [cited July 16 2015]; 3021/2004; pp69-82. Available from: [http://nichol.as/iss/mikolajczyk2004\\_eccv\\_probabilistic\\_assembly.pdf](http://nichol.as/iss/mikolajczyk2004_eccv_probabilistic_assembly.pdf)
- [29] Leibe, B., Seemann, E., Schiele, B.. Pedestrian detection in crowded scenes. IEEE Conference on Computer Vision and Pattern Recognition(CVPR). 2005 [cited July 16 2015]; pp878-885. Available from: <http://10.1109/CVPR.2005.272>
- [30] Barnich, O., Jodogne, S., Van Droogenbroeck, M.. Robust analysis of silhouettes by morphological size distributions. Advanced Concepts for Intelligent Vision Systems(ACIVS). 2006 [cited July 16 2015]; pp734-745. Available from: <http://orbi.ulg.ac.be/bitstream/2268/1378/1/Barnich2006Robust.pdf>
- [31] Zhang, S., Klein, D., Bauckhage, C., Cremers, A.. Fast moving pedestrian detection based on motion segmentation and new motion features. 2015. [updated April 17 2015; cited July 16 2015]. Available from: [http://link.springer.com/article/10.1007/978-3-319-25711-2\\_25](http://link.springer.com/article/10.1007/978-3-319-25711-2_25)
- [32] Bajracharya, M., Moghaddam, B., Howard, A., Brennan, S., & Matthies, L. H. (2009). Results from a Real-time Stereo-based Pedestrian Detection System on a Moving Vehicle. IEEE ICRA Workshop on People Detection and Tracking. [cited July 16 2015] 28(11-12), pp1466-1485. Available from: <http://doi.org/10.1177/0278364909341884>
- [33] Short, N. J.. 3-D Point Cloud Generation from Rigid and Flexible Stereo Vision Systems. Master's Thesis. 2009.
- [34] Wikipedia. Epipolar Geometry. [Web Page] [updated November 3 2015; cited January 13 2016] Available from: [http://en.wikipedia.org/wiki/Epipolar\\_geometry](http://en.wikipedia.org/wiki/Epipolar_geometry)
- [35] Banz, C., Hesselbarth, S., Flatt, H., Blume, H., PirschReal, P.. Time Stereo Vision System using Semi-Global Matching Disparity Estimation: Architecture and FPGA-Implementation. Embedded Computer Systems (SAMOS). 2010 [cited July 16 2015], pp93-101. Available from: <http://doi.org/10.1109/ICSAMOS.2010.5642077>

- [36] Point Grey - Bumblebee Stereo Vision Camera Systems [Datasheet] [updated June 2012; cited April 15 2015]. Available from: <http://www.ptgrey.com/support/downloads/10132>
- [37] Focus Robotics. PCI nDepth Vision System. [Web Page] [cited April 15 2015]. Available from: <http://www.focusrobotics.com/products/systems.html>
- [38] Minoru 3D. Minoru 3D Webcam. [Web Page] [updated 2015; cited April 15 2015]. Available from: <http://www.minoru3d.com/>
- [39] Tordivel AS. Scorpion 3D Stinger. [Datasheet] [updated 2015; cited April 15 2015]. Available from: <http://www.tordivel.no/scorpion/pdf/scorpion%208/PD-2011-0002%20Scorpion%203D%20Stinger%20Camera.pdf>
- [40] VISLAB. 3DV-E system. [Web Page] [cited April 15 2015]. Available from: <http://vislab.it/products/3dv-e-system/>
- [41] IDS. Stereo cameras for 3D vision and robot vision applications!. [Web Page] [cited April 15 2015] Available from: <https://en.ids-imaging.com/ensenso-stereo-3d-camera.html>
- [42] Bertozzi, M., Bombini, L., & Broggi, A.. The VisLab Intercontinental Autonomous Challenge: 13,000 km, 3 months, no driver. 2010 [cited July 16 2015] Available from: <http://www.ce.unipr.it/people/cattani/publications-pdf/itswc2010.pdf>
- [43] Point Grey. Point Grey Flea3 GigE, GigE Digital Camera - Technical Reference. 2013 [cite January 18 2015]. Version 7.0. Available from: <http://www.ptgrey.com/support/downloads/10119/>
- [44] Vision Online - Global Association for Vision Information. GigE Vision - True Plug and Play Connectivity. [Web Page] [cited July 20 2015]. Available from: <http://www.visiononline.org/vision-standards-details.cfm?type=5>
- [45] ROS. Ros Introduction. [Web Page] [updated May 22 2014; cited March 7 2015]. Available from: <http://wiki.ros.org/ROS/Introduction>
- [46] OpenCV. OpenCV - About. [Web Page] [cited March 7 2015]. Available from: <http://opencv.org/about.html>
- [47] Point Grey. Register Reference for Point Grey Digital Cameras. 2015 [cited April 21 2015]. Version 3.1. Available from: <http://www.ptgrey.com/support/downloads/10130>
- [48] Point Grey. Technical Application Notes. 2014 [cited April 21 2015]. Available from: <https://www.ptgrey.com/tan/10351>
- [49] MathWorks. Computer Vision System Toolbox. [Web Page] [cited March 7 2015]. Available from: <http://www.mathworks.com/products/computer-vision/>
- [50] The KITTI Vision Benchmark Suite. Stereo Evaluation 2015. [Web Page] [cited May 23 2015]. Available from: [http://www.cvlibs.net/datasets/kitti/eval\\_scene\\_flow.php?benchmark=stereo](http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=stereo)

- [51] Bradski, G., Kaehler, A.. Learning OpenCV - Computer Vision with the OpenCV library. O'Reilly. Available from: <http://www.cse.iitk.ac.in/users/vision/dipakmj/papers/OReilly%20Learning%20openCV.pdf>
- [52] Peris, Martin. Martin Peris' Blog - 3D RECONSTRUCTION WITH OPENCV AND POINT CLOUD LIBRARY. [Web Page] [updated January 6 2012; cited April 13 2015] Available from: <http://blog.martinperis.com/2012/01/3d-reconstruction-with-opencv-and-point.html>
- [53] PCL - PointCloudLibrary. Documentation. [Web Page] [cited December 2 2015]. Available from: <http://pointclouds.org/documentation/tutorials/>
- [54] PCL Documentation. Downsampling a PointCloud using a VoxelGrid filter [Web Page] [cited December 2 2015]. Available from: <http://pointclouds.org/documentation/tutorials/passthrough.php#passthrough>
- [55] PCL. Removing outliers using a Statistical Outlier Removal filter. [Web Page] [cited December 2 2015]. Available from: [http://pointclouds.org/documentation/tutorials/statistical\\_outlier.php#statistical-outlier-removal](http://pointclouds.org/documentation/tutorials/statistical_outlier.php#statistical-outlier-removal)
- [56] PCL. Downsampling a PointCloud using a VoxelGrid filter [Web Page] [cited December 2 2015]. Available from: [http://pointclouds.org/documentation/tutorials/voxel\\_grid.php#voxelgrid](http://pointclouds.org/documentation/tutorials/voxel_grid.php#voxelgrid)
- [57] PCL. Euclidean Cluster Extraction. [Web Page] [cited December 2 2015]. Available from: [http://pointclouds.org/documentation/tutorials/cluster\\_extraction.php#cluster-extraction](http://pointclouds.org/documentation/tutorials/cluster_extraction.php#cluster-extraction)
- [58] PCL. How to use a KdTree to search. [Web Page] [cited December 2 2015]. Available from: [http://pointclouds.org/documentation/tutorials/kdtree\\_search.php](http://pointclouds.org/documentation/tutorials/kdtree_search.php)
- [59] ROS. stereo\_image\_proc - Package Summary. [Web Page] [update July 30 2015. cited August 8 2015]. Available from: [http://wiki.ros.org/stereo\\_image\\_proc](http://wiki.ros.org/stereo_image_proc)
- [60] Kyto, M., Nuutinen, M., Aalto, P.. Method for measuring stereo camera depth accuracy based on stereoscopic vision. University School of Science and Technology. Available from: [http://www.helsinki.fi/~msjnuuti/pdf/EI\\_2011\\_kyto\\_preprint.pdf](http://www.helsinki.fi/~msjnuuti/pdf/EI_2011_kyto_preprint.pdf)
- [61] Prevenção Rodoviária Portuguesa. Simulador. [Web Page] [cited April 13 2015] Available from: <http://www.velocidade.prp.pt/default.aspx?Page=4031&quad=0>
- [62] National Instruments. What to Expect from a Stereo Vision System. [Web Page] [update June 2013; cited April 13 2015] Available from: <http://zone.ni.com/reference/en-XX/help/372916P-01/nivisionconcepts dita/guid-10d358bd-3dcd-4ccd-a73c-672e48aed39a/>
- [63] Sony. ICX687ALA/ICX687AQA - CCD Image Sensors for Industrial Cameras. [Datasheet] [cited September 20 2015] Available from: [http://www.sony.net/Products/SC-HP/cx\\_news\\_archives/img/pdf/vol\\_67/icx687ala\\_aqa.pdf](http://www.sony.net/Products/SC-HP/cx_news_archives/img/pdf/vol_67/icx687ala_aqa.pdf)



# Appendix

## User Instructions for Stereo Camera Application

In order for the computer to really start detecting people and objects, using the software developed in this project, he has to follow the next steps:

1. Set up the setup explained in chapter 3
2. Verify if the arduino is sending the right signal with an oscilloscope. If not upload the synchronizing software to the arduino.
3. Turn the power on at 12V
4. Verify if the computer recognize the cameras. The first time the user connect the cameras it may be necessary to IP reconfigure. Point Grey Research provides a simple way to do it with the software **GigE Configurator** for Windows)
5. Calibrate the stereo system (follow the User Instructions for calibration presented next).
6. If the cameras are calibrated you can:
  - Compile **stereoCam** package and run lauchfile: **stereoCam.launch** - to use the Stereo Camera Application for ROS
  - Compile **pgrFlea3stereo** program and run **stereoCam** executable - to use the Stereo Camera Application non ROS