Master's Degree in

Mechatronics Engineering

STRUCTURED LIGHT ASSISTED REAL TIME STEREO PHOTOGRAMMETRY FOR ROBOTICS AND AUTOMATION

Novel implementation of stereo matching

Supervisor University of Trento
Prof. Daniele Fontanelli
Prof. Nicola Conci

Supervisor Aalto University
Prof. Juho Kannala

Student
Losi Jacopo 206605

ACADEMIC YEAR 2019/2020

Abstract

In this Master's thesis project a novel implementation of a stereo matching based method is proposed. Moreover, an exhaustive analysis of the state-of-the-art algorithms in that field is outlined. Specifically, both standard and deep learning based methods have been extensively investigated, thus to provide useful insights for the designed implementation. Regarding the developed work, it is basically structured in the following manner.
At first a research phase has been carried out, hence to simply and rapidly test the thought strategy. Subsequently, a first implementation of the algorithm has been designed and tested using data available from the Middlebury 2014 dataset, which is one of the most exploited dataset in the computer vision area. At this stage, numerous tests have been completed and consequently various changes to the algorithm pipeline have been made, in order to improve the final result.
Finally, after that exhaustive researching phase the actual method has been designed and tested using real environment images obtained from the stereo device developed by the company, in which this work has been produced.
Fundamental element of the project is indeed that stereo device. As a matter of fact, the designed algorithm in based on the data produced by the cameras that constitute it. Specifically, the main function of the system designed by LaDiMo is to make the built stereo matching based procedure simultaneously faster and accurate. As a matter of fact, one of the main prerogative of the project was to create an algorithm that has to prove potential real-time results.
This has been, in fact, achieved by applying one of the two methods created. Specifically, it is a lightweight implementation, which strongly exploits the information coming from the LaDiMo device, thus to provide accurate results, keeping the computational time short.
At the end of this Master's thesis images showing the main outcomes obtained are proposed. Moreover, a discussion regarding the further improvements that are going to be added to the project is stated. In fact, the method implemented, being not optimized only demonstrate a potential real-time implementation, which would be certainly achieved through an efficient refactoring of the main pipeline.

Acknowledgements

I wish to thank in primis LaDiMo oy., which allows me to use its resources for the developing of this Master's thesis project, especially the CEO Jouni Halme and the CTO Jorma Palmén and all my colleges, from which I have learned a lot. Moreover, an important thanks goes to the supervisors, who accurately follow and revised this work, professor Juho Kannala, professor Daniele Fontanelli and professor Nicola Conci. I thank, then, the company advisor MsC. Sami Ruuskanen, whose help and support allow me to accurately complete this project.
Moreover, I wish to thank the EIT Digital Master School, which developed such a relevant academic program, continuously supporting the students and giving them the possibility to work closely with important start-ups and companies in Europe.

Family and friends

An important thanks goes, then, to my family, which always supports on both private and academic life, without with all of this would not be possible.
Then, I wish to thank my friends, who helped me in difficult moments and always encourage me in all my decisions.

To my parents.
For their support, encouragement and patient.

Espoo, July 15, 2020

Jacopo Losi

Abbreviations and Acronyms

API	Application programming interface
BSD	Berkeley Software Distribution
CNN	Convolutional Neural Network
DOE	Diffractive Optical Element
DP	Dynamic Programming
DSI	Disparity space image
DSLR	Digital single-lens reflex
IDE	Integrated development environment
LiDAR	Light detection and ranging
MRF	Markov Random Field. Set of random variables having Markov property, described by undirected graph.
MI	Mutual Information
NCC	Normalized Cross Correlation
PKR	Peak Ratio
SAD	Sum-of-absolute-difference
SGM	Semi-Global Matching. Stereo matching algorithm developed by Heiko Hirschmüller
SGBM	Semi-Global Box Matching. Area-based version of the previous algorithm. The main changing stands in the aggregation cost phase that here is performed over a certain subregion for every pixel and not over the entire image dimensions.
SSD	Sum-of-squared-difference. Intensity based matching cost technique

Contents

Chapter 1

Introduction

The work developed in this Master's thesis is focused in a novel implementation of a stereo matching based procedure, which exploits, on top of a pair of stereo rectified images of the analysed scene, a sparse cloud of points generated by a multi-cameras device. The device, designed by LaDiMo, the company in which this thesis has been developed, was actually employed to accelerate a standard stereo matching pipeline, thus to achieve real-time performance in 3D reconstruction.
Therefore, the main goal of this Master's thesis stand in the designing of an algorithm that allows to detect the object of an analysed scene, by increasing the density of an initial sparse cloud of point and exploiting a pair of rectified RGB stereo images. As aforementioned, the main task of the utilization of the sparse 3D point cloud is to accelerate the stereo matching procedure, which would be otherwise slow and computationally expensive.
The final scopes of this algorithm can be theoretically multiple. As a matter of fact, real-time object detection is fundamental in autonomous driving applications. The built depth estimation process can effectively facilitate warehouse managing operations. Moreover the 3D scene reconstruction provided by the algorithm is crucial when dealing with automation and robotics, especially for indoor implementations.
Hence the actual final application in which the thought method can be employed are different. However, the pivotal goal of this thesis is the design of an algorithm that provides an accurate dense 3D cloud of point, which describes the analysed scene. Then, this would be employed for specific tasks, depending on the particular use case.

1.1 Problem statement

Dense and accurate disparity maps are the key factor for obtaining correct depth estimations for many computer vision applications such as autonomous driving, 3D reconstruction, object detection and robotics. Therefore, stereo matching and disparity estimation can be identified as fundamental problems in the current developments of computer vision [1].
Multiple methods for disparity estimation has been developed for many years [2]. Older strategies are focused on local-based or global-based methods. On the contrary, deep learning based strategies applied to local or global methods has been recently proposed.
The latter approach aims to a precise local correspondence exploiting deep learning and applying Semi-Global Matching (SGM) as the regularization step of the pipeline. Therefore, deep learning techniques such as FlowNet and DispNet [1] are used as the end-to-end part of the pipeline. According to the current benchmark database ranks for stereo matching algorithms, e.g. the one published in the KITTI website[1], the state-of-the-art implementations are based on deep learning methods. However, these strategies lack in accuracy compared to the standard pipelines when dealing with scenes different from the ones used for training the network. Thus, this is probably due to the differences between real environment and the training database as underlined in [1] and [3].
As aforementioned, the state-of-the-art methods to recover dense disparity maps from stereo pairs are focused on deep Convolutional Neural Networks (CNNs) trained end-to-end [4]. Most of these techniques, which will be subsequently described, exploit as regularization phase the SGM method.
The Hirschmüller's algorithm [5] is, in fact, used in both standard and deep learning based techniques because, considering the available benchmark data, its approach is still among the best performing in terms of computational cost and accuracy. For this reason, it is the preferred trade-off for most real-time applications, which are based on both the standard, local and global, and deep learning based strategies.
Therefore, as previously anticipated, the main problem, which this project is focused on, is the 3D reconstruction of a specific scene, obtained by depth data estimation. The algorithm pipeline is based on a pair of stereo images, necessary for the disparity calculation, which is then related to the depth of the specific pixel. Additionally, crucial input to the method is the sparse cloud of point, which is exploited to accelerate the stereo matching procedure.
At the end of the pipeline, a 3D denser point cloud is basically obtained. That

[1] `http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=stereo`

cloud of point should be, thus, able to describe the object in the scene, making its identification simple.
Considering the stereo device designed by the company and hence the data available as input to the whole process, it has been decided since the beginning to develop the algorithm on standard method based strategy. Moreover, the choice of do not employing a deep learning based approach was also due to the so called *domain shifting* [3]. This is defined as a dropping in accuracy when running the algorithm on a scene that present data different from the one used for training, which are typically synthetic data. Hence, the accessible device, the data that it can provide and the requirement of being able to apply the algorithm on whatever environment bring to the decision of basing the implementation on a standard approach.
Therefore, some initial tests have been carried out to obtain a rough estimation of the feasibility of the choice done. Actually, the main aspect that has been researched was the potential improvement, primarily in terms of computational time, that the employment of the sparse point cloud can give to the whole pipeline. This idea is similar to what has been proposed by Poggi et al. in their ***Guided Stereo Matching*** [3], which is based on a deep learning based method, though. Differently from the designed approach, they developed a method capable of reducing the lack in accuracy when shifting to unseen environments, by exploiting a set of sparse accurate depth measurements, which can be obtained from an type of source, such as LiDAR.
In this thesis work the problem of estimating a dense and reliable 3D point cloud, starting from a pair of rectified stereo images and a set of accurate depth measurements, has been tackled in a different manner. Specifically, two main strategies have been designed and analysed. A so called SGM-based method, which strongly follows the steps of the standard Semi-Global Matching algorithm to recover the disparity information. Then, a novel approach has been created. It primarily employs the information that are available from the input point cloud. The peculiar feature of this method stands in the calculation of the local derivatives among neighboring points of the cloud. This can be defined as the key characteristic of the algorithm, which will be fully described in the following chapters.

1.2 Structure of the Thesis

Therefore, the two aforementioned stereo matching strategies and the whole presented work, which is going to be widely discussed in the next chapters, is now generally set out, thus to provide to the reader a generic understanding of the entire thesis project.
First of all, in Chapter 2 a theoretical background is delineated.

It will cover the relevant topics necessary to provide a complete understanding of the designed project. Specifically, a wide analysis will involve the bases of most of the stereo matching methods. Additionally, epipolar geometry and the meaning of *rectification* are outlined, providing the relevant mathematical formulas, thus to support their explanation and correlation with the algorithms developed.
Moreover, a substantial part of the background is dedicated to the description of the standard and the deep learning based stereo matching algorithms, which have been precisely analysed through an exhaustive literature review.
Then, at the end of Chapter 2 a section concerning the image processing techniques is proposed, being some of those techniques exploited and tested during the pre and post-processing phases of the project. Additionally, a section of that chapter covers the most used edge detection and segmentation algorithms, which are going to be of particular interest for the future improvements of this project.
After the background chapter, the actual work is deeply outlined. Chapter 3 is related to the hardware and software environment employed.
Specifically, the general information on the datasets exploited are set out and their specific use in the different phases of the project described.
Moreover, the LaDiMo device used for recovering the real environment images and the initial point cloud is briefly described. In particular, its main functionalities and its hardware setup are stated, thus to provide a comprehensible explanation of the main features of its working pipeline, without addressing sensible details, closely related to the company's domestic policy.
Then the methods implemented in the different approaches designed are displayed in Chapter 4. Specifically, the multiple strategies designed are generally outlined, together with well-known algorithm employed in the algorithm pipeline. Moreover, the specific pre and post-processing procedures utilized are proposed and their benefits highlighted.
After that the actual implementation of the algorithms will take place. The structure of the different approaches designed is fully introduced and explained in detail. Thus, Chapter 5 includes the details about the initial testing phase carried out and the motivations that brought to implement that. Then, the three main algorithms designed and tested are proposed, explaining their specific features and their formulation.
At the end of that, in Chapter 6 the results achieved are shown and commented. Therefore, the evaluation of the entire project in accomplished, displaying the images obtained and proposing a comparison between the designed method and the standard Semi-Global Block Matching algorithm available in the OpenCV library.
The final discussion over the whole presented work is then outlined in Chapter 7. The main scope of that is to provide comments on the specific implementation of the algorithm and on the achieved results, which should provide an accomplish-

ment of the initial goal set. Some general thoughts about the completed work are presented and generic ideas for further improvements are expressed.
Thus, the entire work concludes with a summary of the thesis, provided in Chapter 8.

Chapter 2

Theoretical Background and Related Work

In Chapter 1 a general overview of the project has been proposed and the main purposes of the developed work have been outlined.
In this chapter an exhaustive revision of the theoretical basis of most of the stereo-matching methods is presented. Then, epipolar geometry, camera calibration and disparity estimation algorithms are specifically described. Starting from the necessary mathematical fundamentals, the discussion moves on the disparity estimation algorithms. Then, the chapter focus on the main benefits and drawbacks of standard and novel approaches in depths computation. Comparison between photogrammetry based and deep learning based algorithms is proposed, to provide a clear explanation of the decisions implemented. At the end a relevant discussion about the main image processing techniques is carried out, thus to analyse some of the algorithms that have been used in the pre and post-processing phases of the project.
Taking into account the standard algorithms for stereo correspondence, they are mainly based on photogrammetry related operations that fully exploit the mathematical principles of stereo geometry. They can be conventionally classified into two general categories, local and global approaches [2]. Specifically, the local-based methods tend to estimate the disparity image through the analysis of the corresponding pixels between the left and right views of the scene, concerning a particular area. Then, the matching cost related to all the potential matches defined for that area is evaluated and employed in the exact disparity estimation. In order to recover from low accuracy proper of the previous strategies, global-based methods try to calculate the disparity values by minimizing an energy function. In this context, Semi-Global Matching (SGM) combines strong factors of global and local approaches allowing to obtain a good trade-off between computational cost and accuracy.

Technically speaking, SGM applies a pixelwise, Mutual Information (MI) based matching cost for analysing pixel intensity value differences in the input images [5]. Moreover, pixelwise matching is enhanced with a smoothness constraint, which leads to a global cost function. Then, post-processing techniques are applied to remove outliers and filter the image.
Referring to the analysis performed by Scharstein and Szeliski [2], SGM carries out four main steps, as well as most of the stereo matching algorithms. These are defined as: *matching cost computation*, *cost aggregation*, *disparity computation* and *disparity refinement*.
Considering the former, it is usually based on absolute, squared or sampling insensitive difference between pixel intensities [5]. Although those methods allow to reach a reliable accuracy, they are sensitive to radiometric difference. Thus, cost based on image gradients or window-based methods, such as Rank and Census transform [6], became an optimal choice. Furthermore, Mutual Information results as a good trade-off for dealing with complex radiometric relationships between images.
In the second phase, cost aggregation accumulates the matching costs coming from multiple directions and along the considered disparity levels. Subsequently, disparity evaluation is defined for each pixel, as the one with the lowest cost. This is the approach typically used in local methods. Global algorithms, contrarily, used to get rid of the aggregation step and define a global energy function. Over that function, pixel similarity and disparity smoothness are enforced with different strategies. In this latter case, the best disparity is identified finding the minimum of the cost function. This is achieved with multiple techniques such as: Dynamic Programming (DP) [7], Belief Propagation [8] or Graph Cuts [9].
Disparity refinement tends to differ more among the different methods. Post-processing techniques such as filtering, outlier removal and left-right consistency check are applied in general.
As aforementioned, among the top-ranked stereo matching algorithms, SGM results to be the best performing in terms of computational time and accuracy. Its benefits stand in the hierarchical computation of the matching cost, which mainly exploits Mutual Information. Cost aggregation is achieved considering a global energy function and a pathwise pixel optimization. The final disparity is chosen with a winner takes all strategy. Disparity refinement is completed by consistency check between left and right disparity images.
Therefore, this brief introduction presents the general concepts of most of the standard stereo matching algorithms. However, an analysis of the basis of stereo correspondence is equally important in this context, thus to highlight its relevance for multiple applications such as: autonomous driving, robotics, object detection and 3D reconstruction.

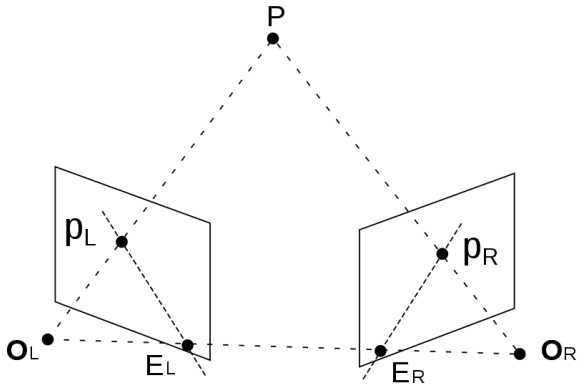


Figure 2.1: Epipolar geometry - Credits to ZooFari

2.1 Stereo matching and pixel correspondence

First of all, stereo matching is defined as the process of estimating a 3D model of a scene, starting from two or more images. Therefore, the matching pixels between the images are found and their 2D positions are converted into 3D depths. This procedure and the definition of a dense depth map, generally obtained by assigning the relative depth to the input pixels, is going to be properly described in the following sections.
Nonetheless, essential is the concept of disparity, defined as the amount of horizontal motion between two properly configured images of a stereo pair. It is inversely proportional to the distance from the observer, i.e. the camera. Although these ideas are relatively simple to understand, the challenging task within this process stands in establishing dense and accurate inter-image correspondences [10].
As already underlined, stereo matching is one of the most widely studied topic in computer vision since years and it continues to be one of the most active research in that field. In fact, modelling of human visual systems, robotic navigation and manipulation, autonomous driving [3] and 3D model building are only some of the possible applications.
Therefore, it worth to explain the fundamental principles of stereo matching, such as epipolar geometry, rectification and disparity map, in order to provide a comprehensive outline of the techniques and the algorithms handled in this project.

2.1.1 Concepts of stereo geometry

In this section, a generic introduction to the main concepts of stereo geometry is proposed. This will be, then, further extended in the following sections, where the different aspects of epipolar geometry and pixel correspondence will be specifically

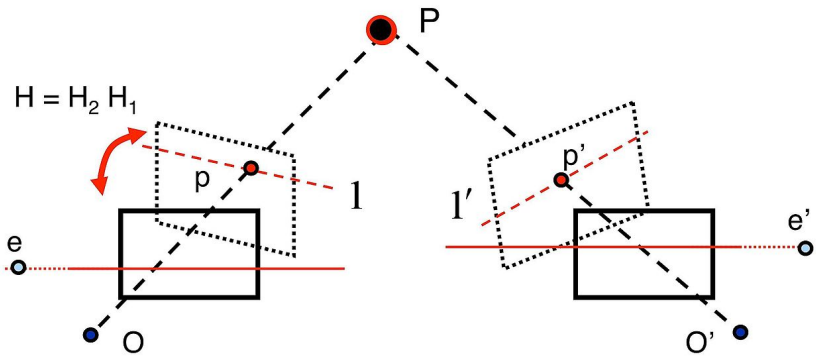


Figure 2.2: Image rectification - Credits to Silvio Savarese

explained.
Main goal of epipolar geometry is the computation of pixels correspondences among the input images. Neighboring pixels information, cameras positions and their calibration data are fundamental to achieve that. Figure 2.1 displays a pixel in one image $\mathbf{p_L}$ projected to its correspondent epipolar line segment in the other image, the dashed line $\mathbf{l_R}$, which is lower bounded by the projection of the first camera center into the second camera plane, i.e. the epipole $\mathbf{E_R}$. Projecting the epipolar line in the second image back to the first, another line is obtained, bounded by the correspondent epipole $\mathbf{E_L}$. The extensions to infinity of these two segment are identified as the epipolar lines, which are defined by the intersection of the two image planes with the epipolar plane. A fundamental property is that the epipolar plane passes through both camera centers $\mathbf{O_L}$ and $\mathbf{O_R}$, as well as point $\mathbf{P}$. Therefore, they all lie in the same plane.

2.1.2 Concepts of rectification

Epipolar geometry for a pair of cameras is relative to pose and calibration of the device and can be computed using the fundamental matrix, which can be obtained applying the eight point algorithm [11]. This geometry allows, then, to find the correspondent pixels between the two images using the constraint of the epipolar lines. This is possible, because, as explained in 2.1.1, considered a pixel in one image, the corresponding one lies on the relative epipolar line, in the other image plane.
Beside that, pixels correlations can be more efficiently performed by rectifying the input images [11]. In Figure 2.2 is clearly visible the outcome of this process and its advantages. In fact, it is not difficult to notice that corresponding horizontal scanlines are epipolar lines. The essential importance of this standard rectified

geometry is clearly explained by the following equation,

$$d = f\frac{B}{Z} \tag{2.1}$$

that leads to a linear relationship between the inverse of the depth Z and disparity d, where f is the focal length (in pixel) and B the baseline. Moreover, the relationship between the corresponding pixels in the left and right images can be defined as follows:

$$x' = x + d(x, y),\ y' = y \tag{2.2}$$

Thus, the main step for recovering a depth image of a scene is the estimation of the disparity map $d(x, y)$.
As introduced at the beginning, the best disparity map is estimated after the rectification process. This is performed by comparing the similarity of corresponding pixels, as defined in equation 2.2, and storing them in a disparity space image (DSI) [12], in general labelled with $C(x, y, d)$, which is then processed with multiple algorithms. Specifically, the concept of DSI can be described as any image or function established over a continuous or discretized representation of a disparity space (x, y, d). Basically, it denotes the cost, as log likelihood or confidence measure, of a specific match related to $d(x, y)$. Therefore, the main task of any stereo matching algorithm is to determine the specific surface included in the DSI that has some optimal property, as can be lowest cost and finest smoothness. This procedure can be more simply described as the finding of the univalued function, contained in $d(x, y)$, which best approximate the structure of the surfaces in the scene.

2.2 Epipolar geometry and Rectification

Fundamental problem of stereo vision is the estimation of 3D locations of points from at least two corresponding input images. This process, which comprises concurrent computation of both 3D geometry and camera pose, is generally known as structure from motion [10].
In the explanation of these topics it is necessary to start discussing about the triangulation. Then, the concept of epipolar geometry is outlined and after that the notions of camera calibration and rectification.

2.2.1 Triangulation

Triangulation is the problem of detecting 3D point positions from a collection of corresponding 2D image locations, assuming that camera poses are known.

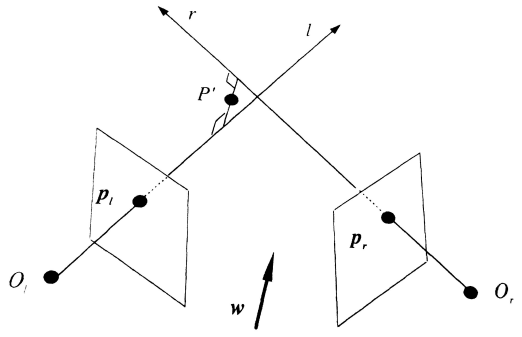


Figure 2.3: 3D triangulation by finding point P' that lies nearest to all of the optical rays

Figure 2.3 shows one of the easiest methods to tackle this problem. Objective is to evaluate the 3D position of P' that have the smallest error to all of the 3D optical rays coming from the camera centers, which identify the 2D point locations in the image plane, i.e. P_r and P_l. As shown in Figure 2.3, the rays starts from the camera centers, O_l and O_r, and go in direction of l and r, which can be defined using the camera matrix, usually identified with $P_j = K_j[R_j|t_j]$. The closest point to P on this ray minimizes the distance,

$$\|O_j + d_j\hat{v}_j - P\|^2 \tag{2.3}$$

Therefore, because of the minimum is $d_j = \hat{v}_j \cdot (p - c_j)$, the nearest points are calculated as:

$$q_j = O_j + (\hat{v}_j\hat{v}_j^\top)(P - O_j) = O_j + (P - O_j)_\parallel \tag{2.4}$$

Hence, the optimal value for P, obtained solving a least square problem, becomes,

$$P = \Big[\sum_j (I - \hat{v}_j\hat{v}_j^\top)\Big]^{-1} = \Big[\sum_j (I - \hat{v}_j\hat{v}_j^\top)O_j\Big] \tag{2.5}$$

where in all of the previous equation $\hat{v}_j$ symbolizes the unit vector.

2.2.2 Epipolar geometry

The intrinsic projective geometry between two views is known as epipolar geometry. It is only dependent on the cameras' internal parameters and pose. The 3×3 rank 2 matrix that defines this geometry is the fundamental matrix F.

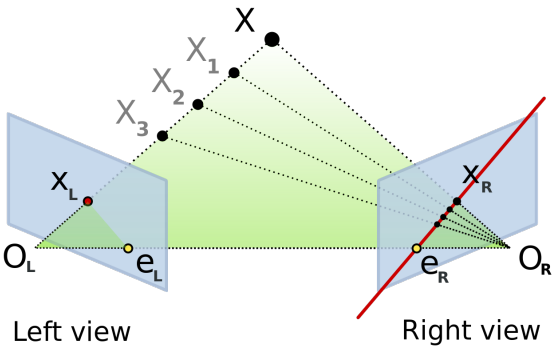


Figure 2.4: Epipolar geometry. Image point x_L back-projects to a ray in a 3D space defined by O_R and x_L. This ray becomes a line, the epipolar line to x_L, in the second view. The image of X, i.e. x_R, must lie on that red line - Credits to Arne Nordmann

The epipolar geometry is the basis for finding corresponding points in stereo matching. It is basically defined by the intersection between image planes and the one on which the cameras baseline lies.
A fundamental property, that makes this geometry extremely useful, is that image points, space point and camera centers are coplanar. Considering Figure 2.4 and assuming that only $\mathbf{x_L}$ is known, that geometry allows to constraint the corresponding point $\mathbf{x_R}$. The epipolar plane is defined by the baseline and the ray that comes from $\mathbf{x_L}$. Hence, knowing that $\mathbf{x_R}$ lies on the same plane, that point belongs to $\mathbf{l_R}$, i.e. the intersection between the epipolar plane and the second image plane, knows as the epipolar line to $\mathbf{x_L}$. Therefore, exploiting this property, the searching of corresponding points is constrained to only one line inside the image.
Mathematical definition of the epipolar geometry is the fundamental matrix F. As already demonstrated through Figure 2.4, for each point $\mathbf{x_L}$ in one image, the corresponding epipolar line $\mathbf{l_R}$ to that point belongs to the other image plane. Moreover, any point $\mathbf{x_R}$ in the second image, which is related to point $\mathbf{x_L}$, lies on $\mathbf{l_R}$. Hence, the epipolar line is described as the projection in the second image of the ray that comes from the point in the first image, passing through its camera center. This defines a map, $\mathbf{x_L} \rightarrow \mathbf{l_R}$, which relates the points in one image with the corresponding epipolar lines in the second image. This correlation, between points and lines, is represented by the fundamental matrix F.
Considering the aforementioned map $\mathbf{x_L} \rightarrow \mathbf{l_R}$ described by F, an important property of the fundamental matrix is defined,

$$x_R^\top F x_L = 0 \tag{2.6}$$

Therefore, assuming two corresponding points $\mathbf{x_L}$ and $\mathbf{x_R}$, it is known that $\mathbf{x_R}$ lies on the epipolar line $\mathbf{l_R} = F\mathbf{x_L}$. Thus, the mathematical correlation is,

$$0 = x_R^\top l_R = x_R^\top F x_L \tag{2.7}$$

Reciprocally, if image points comply the relation 2.6, then the rays identified by these points are coplanar. For corresponding points this is a necessary condition. Equation 2.6 is extremely important because it allows to characterize the fundamental matrix without reference to the camera matrices [11]. Thus, using at least 7 correspondences, it is possible to recover the fundamental matrix F. This estimation is known as *weak calibration*.

2.2.3 Rectification

As introduced in section 2.1.2, image rectification is defined as the process of obtaining a pair of *matched epipolar projections* from a pair of stereo images, which are taken from generally differing viewpoints. In the rectified projections the epipolar lines become parallel with respect to the x-axis. Thus, they match between the stereo pair and so the disparities are in the x-direction only.
In order to obtain a rectified stereo pair, 2D projective transformations are employed to the images, so that the epipolar lines can match. Using this method, the transformations are built up in a way that the corresponding points have almost the same y-coordinate. Actually, this strategy leads to a minimal distortion on the images, being the two transformations arbitrary. However, working on rectified images, the matching problem is highly simplified, being correlated only to epipolar geometry and near-correspondence.
Core problem of this section is to find the appropriate projective transformation H. Indeed, to get epipolar lines parallel with x-axis, the epipole should be mapped to an infinite point. This, has to be done correctly, otherwise intensive projective distortion of the image can happen. For this reason, constraints are set on the definition of H.
First of all, restricting H to be a rigid transformation in the neighborhood of a given point[1], the errors are reduced.
Once the epipole has been mapped to infinity, it is then necessary define a map to match the corresponding epipolar lines. This resampling is build up in such a way that, being l_L and l_R any pair of epipolar lines, then,

$$H^{-\top} l_L = H'^{-\top} l_R \tag{2.8}$$

[1]this means that to first-order, the neighbourhood of the point may be subjected to rotation and translation only

Satisfying the condition above, a matched pair of transformations is recovered. Specifically, at first H' is chosen, so that it can map the epipole e_R to infinity. Then the matching transformation H is defined minimizing the sum-of-square distances,

$$\sum_i d(Hx_{L_i}, H'x_{R_i})^2 \tag{2.9}$$

Therefore, the full algorithm can be summarized as follows.
The outcome of this resampling process is a pair of stereo images whose epipolar lines are horizontal. Hence, the disparities are calculated along the epipolar lines. First of all, at least seven corresponding matches are defined. This allows to compute the fundamental matrix F, applying the so called eight-point algorithm, and after that the two epipoles are found. After that, there is the selection of the projective transformation H', that maps the epipole of the support image to infinity. The corresponding transformation H is found solving the least-square problem. Finally both of the input images are resampled according to H and H'.

2.3 Standard stereo methods and dense correspondence

In this section a brief delineation of the general pipeline implemented in most of the stereo matching method is presented. Moreover, as a theoretical completion of what introduced above, some generic algorithms are further explained.

2.3.1 Standard stereo matching algorithms

Standard stereo matching algorithm follows in general a subspace of the following methods: matching cost computation, cost aggregation, disparity computation and optimization and disparity refinement [2].
A preliminary distinction, based on those phases, separates stereo methods between local or window-base and global methods.
In local methods, the disparity computation depends on the pixel intensities within a limited window, considering a specific region of the image.
On the contrary, global algorithms, are based on an energy function. In the latter smoothness assumptions are defined and then a global optimization problem is solved. These algorithm are mainly distinguished considering the minimization strategy, such that, simulated annealing, graph cuts or belief propagation.
Between these two classes there are iterative and hierarchical algorithms. The latter aim to constraint gradually the disparity estimation from the coarser to the finer level [5].
Considering the first general step of stereo matching algorithms, the matching cost,

there are multiple measures to define it. Among the most prevalent pixel-based algorithm can be included square intensity differences, absolute intensity differences, mean-squared error and mean absolute difference.
Other common matching cost comprehend normalized cross-correlation, which is similar to sum of squared difference (SSD), and binary methods, which, however, tend to not be used any longer.
On the other hand, more robust algorithms are generally used for their insensitivity to non-stationary exposure and illumination changes. Entropy measures and non-parametric functions such as, Rank and Census transform [13], sampling insensitive difference [7] and hierarchical mutual information [5], are some examples. In particular, they allow to obtain accurate performance when considerable exposure or appearance variations are present.
Drawing up some conclusion regarding the local methods, the core steps are the matching cost calculation and the aggregation phase. Disparity estimation, then, becomes trivial. Each pixel takes the disparity level whose cost value is the minimum. This approach is said to be a local *winner-take-all* optimization. A drawback of this approach is that the matches are imposed for the reference image. While points in the support image might have multiple correct matches. For this reason, cross-checking and post-processing become more relevant here.
Summarizing the general pipeline of global stereo matching methods, they often get rid of the aggregation step. They usually perform some sort of optimization steps after disparity estimation, exploiting the smoothness constraints as aggregation part.
Goal of this approach is to find the solution to a global energy function, i.e. the disparity d that minimizes the energy,

$$E(d) = E_d(d) + \lambda E_s(d) \tag{2.10}$$

where $E_d(d)$ is the data term and $E_s(d)$ the smoothness term. Adopting the definition of disparity space image (DSI) matching cost, introduced in Section 2.1.2, the data energy is calculated as:

$$E_d(d) = \sum_{(x,y)} C(x, y, d(x, y)) \tag{2.11}$$

where C is the DSI. Then, the smoothness term is usually defined as:

$$E_s(d) = \sum_{(x,y)} \rho(d(x, y) - d(x + 1, y)) + \rho(d(x, y) - d(x, y + 1)) \tag{2.12}$$

where ρ is some monotonically increasing function of disparity difference. For some implementations, the smoothness energy term can also be based on intensity differences,

$$\rho_d(d(x,y) - d(x+1,y)) \cdot \rho_I(\|I(x,y) - I(x+1,y)\|) \tag{2.13}$$

where ρ_I is a monotonically decreasing function, which depends on the intensity differences and makes the smoothness costs lower at high-intensity gradients. After the energy function has been clearly identified, different categories of algorithms can be exploited to recover a (local) minimum. Graph cut, belief propagation and Markov random field (MRF) based methods have been proved to give the most accurate results.
Mentioning some hybrid methods, there are cooperative algorithms and others based on coarse to fine incremental steps. Cooperative algorithms were some of the earliest proposed for disparity estimation. They are influenced by human stereo vision processing models. They operate by iteratively improve disparity evaluations using non-linear calculations, leading to a result similar to the one of the global methods. Iterative algorithms are among the current best algorithms. The main idea is to successively choose the best disparity among all of the possible ones. A coarse-to-fine framework is usually used to speed up the computations.
Dealing with global optimization methods, it is worth to mention dynamic programming (DP) technique. Unlike solutions based on equation 2.10, dynamic programming allows to reach global minimum exploiting independent scanlines. This approach works over a slice of the DSI, i.e. the matching cost cube, finding the path associated to the minimum cost. The generic implementation of DP along a scanline y and for each input state in a 2D cost matrix $D(m,n)$ leads to combine its DSI value with its previous cost values as follows,

$$C'(m,n) = C(m+n, m-n, y) \tag{2.14}$$

Correct cost selection in presence of occluded pixels and difficulties with scanline consistency are some of the weakness of DP. Multiple algorithms have been proposed to recover from these problems. Scharstein and Szeliski [2] proposed an alternative to standard DP, improving recursively independent scanlines in the global energy function,

$$D(x,y,d) = C(x,y,d) + \min_{d'}\{D(x-1,y,d') + \rho_d(d-d')\} \tag{2.15}$$

An upgrade of this scanline optimization is actually the aggregation cost approach used in SGM method [5]. In this case, a cumulative cost function is evaluated from at least eight directions. Intuitively, this approach accesses accurate results and it is highly efficient.
Considering more recent improvements to stereo matching, segmentation-based techniques hold a prominent position. In this case, an initial segmentation of the reference image is performed. Then, disparities are estimated pixelwise using

local methods. Citing couple of recent approaches, Klaus, Sormann and Karner [8] segment the image with mean shift, to get initial disparity estimations. Then they apply re-fitting with global planes, and perform final MRF with loopy belief propagation.
Wang and Zheng [14] built a similar top ranked algorithm. They segment the image with local plane fits. Then run cooperative optimization of neighboring plane fit parameters. Others developed similar approaches exploiting color correlation and left-right consistency check for occlusion detection [15] or focusing on alpha mattes fractional pixel extraction [16].
As explained in section 2.1 the area of interest for stereo matching and disparity estimation is one of the most extensively researched topic in computer vision. Nowadays approaches based on Convolutional Neural Networks (CNNs) and deep learning are going to be the highest ranked in the standard database. Although, their performance in accuracy tends to decrease a lot when moving from dataset images to real ones.
Therefore, novel strategies based on standard stereo geometry algorithms could reach consistent accuracy even in real time [17]. Thus, as described above, after the structure of the cost volume or DSI has been delineated, the actual pixelwise photoconsistency measures are computed. Multiple methods to achieve this has been proposed during years and already explained in the previous sections. Then, depth computation is obtained with different form of optimizations. These ranges among local, global or hybrid frameworks.
Consequently, starting from classical stereo-based methods and building up a novel pipeline, accurate real time depth map estimations can be achieved.

2.3.2 Deep learning based methods

Considering that disparity estimation from a rectified stereo pair is still one of the most important tasks in computer vision, the latest years have seen an important development of deep learning based methods.
Usually these approaches comprises a main pipeline, based on a standard local or global method, whose parameters are finely tuned exploiting Convolutional Neural Networks (CNNs). Therefore, in most of the cases, Semi-Global Matching (SGM) is used as regularization method, because of its accuracy and relatively low computational time. Then, deep learning based methods are used for tuning the penalty parameters, which are related to smoothness and discontinuity of the disparity map. As described above, those penalties are empirically adjusted in the standard methods. Therefore, the CNNs based approach aims at learn the penalties, in correlation with the 3D structure of the objects in the scene, to achieve an improvement in the disparity map.
As aforementioned, several of these deep learning approaches has been proposed

in the latest years, and some of them has been able to reach state-of-the-art level of accuracy on the KITTI datasets [18].
Even though these method suffer drop in accuracy when shifting from synthetic to real images, it is worth to describe some of them, which are ranked among the state-of-the-art methods in the KITTI benchmark.
Seki and Pollefeys proposed **SGM-Nets** [1], a CNNs based method for penalty estimation, which exploit SGM as regularization technique in the main pipeline. They used small image patches and their locations as inputs to the network to predict penalties for the 3D object structures. Actually, they developed a novel loss function for training the network, whose inputs are small image patches and their related positions. Moreover, they managed to separate positive and negative disparity changes, thus to retrieve object structures more discriminatively.
More in detail, SGM-Net produces P_1 and P_2, which are the specific penalty values generally used in the aggregation cost phase of the standard SGM pipeline, for each pixel. This is achieved though a training and a testing phase. In the former, the network is iteratively trained by minimizing a *Path cost* and a *Neighbor Cost*. In testing, the standard SGM pipeline is run using the penalties estimated by the network. Actually, the *Path Cost* is how the authors of the paper called the loss function that they developed, which they minimized using forward and back propagation. Moreover, they introduced the *Neighbor cost* function for removing the ambiguous disparities traversed along each path, which can lead to wrong penalties.
Another approach similar to the one just described is the method developed by Kuzmin et al. [19]. The core of their method is the prediction of the local parameters of cost volume aggregation process, which they perform exploiting a deep convolutional network. Thus, as in [1], they avoid to apply deep learning of pixel appearance descriptors, using classical matching scores instead. Thus, they refuse learning high dimensional descriptors and matching them, as it used to happen in most of the first deep learning based algorithms. On the contrary, they focus the learning on the cost-aggregation step. Thus, they defined the overall matching cost using a combination of Census transform and Sum of Absolute Difference (SAD). Then, they carry out the cost-aggregation phase, which is peculiar for smoothing the general matching cost and correct the wrong matches, applying the domain transform ([20],[21]). Basically, they develop a deep CNN to estimate on a pixel basis the cost-aggregation parameters and make them spatially varying. In this way, they were able to have smoother disparities on the same object and avoid smoothing across object boundaries. Therefore, combining standard methods for the matching cost and applying end-to-end deep learning process to the cost-aggregation step, they obtain state-of-the-art accuracy in the KITTI 2015 dataset. Differently from [22], [23] and similarly to [24], they build up an end-to-

end learning method that comprises all the part of depth map computation in it. Furthermore, unlike [24], their approach exploits classical stereo matching techniques as modules within a more complex neural network structure. Their process encompasses the definition of a general cost volume, the cost-aggregation step over that volume and the final winner-takes-all label selection. Then, left-right consistency and filling of occluded pixels are performed as post-processing.
A different approach with respect to the previous one stands in the implementation developed by Žbontar and LeCun [22]. They focus their attention on the matching cost computation, which is usually the first step of a stereo matching algorithm. They developed their method using a CNN to learn similarity measures on small image patches. Specifically, they structure the training in a supervised manner building up a binary classification data set with examples of similar and dissimilar pairs of patches. Moreover, they carry out two different architectures, one adjusted for speed, and another one for accuracy. Thus, the output of the network handles the initialization of the stereo matching cost. After that, post processing steps are applied: cross-based cost aggregation, semi-global matching and finally disparity refinement techniques such as, left-right consistency check, subpixel enhancement, median and bilateral filters.
Technically speaking, they present a network that is trained on pairs of small image patches, for which the disparity value is known. Then, they initialize the matching cost using the output of the network. After that, they propose a series of common post-processing steps, which are though crucial to obtain accurate results. Cross-based cost aggregation is exploited for combining matching cost between neighboring pixels with similar intensities. Then, constraints are enforced for smoothness and left-right consistency check applied for detect and eliminate errors in occluded regions. The final disparity map is then obtained applying median and bilateral filters, useful for subpixel enhancement.
Related implementations to [22] are the works by Haeusler et al. [25] and by Spyropoulos et al. [26]. Using a similar approach, they concentrate their attention on predicting the confidence of the calculated matching cost. Specifically, in [25], the aim was using a random forest classifier to connect multiple confidence measures. Similarly, the authors of [26] focused in estimating the confidence of the matching cost by training a random forest classifier. Then, they employed the predictions as mild constraints in a Markov Random Field (MRF) aiming at reducing the errors of the stereo method.
Focusing on confidence prediction intended at the estimation of dense disparity map, it is worth to cite the work of [27]. They adopted two channels disparity patches as inputs of a CNN, predicting the correctness of stereo correspondences, that is the confidence. Using these specific patches, they managed to simultaneously train features and classifiers. Furthermore, they incorporate the predicted

confidence into SGM, adjusting its parameters directly.
Unlike methods based on hand crafted features, which lead to limited accuracy, authors of [27] leverage CNNs to overcome that problem. In fact, CNNs support high performance from low level processing, such as patch based matching, to high level, like scene classification and object detection.
Therefore, as previously stated, they conceive a two channels disparity patch, based on the concept of standard confidence features. Then they apply those patches as input to the network, obtaining a simultaneous training of discriminative features and classifiers. Moreover, they also developed three different types of network structures to manage the trade-off between computational time and accuracy. Finally, the confidence was combined into SGM, so that dense disparity map can be obtained.

2.3.2.1 Confidence measures

Considering some of the deep learning based methods described in Section 2.3.2, becomes necessary to define the main typology of elements proposed during the years to estimate the confidence of stereo correspondences.
Because of many features were introduced by different works in computer vision field, Hu and Mordohai [28] developed a broad evaluation of them, leading to the definition of five groups, used for categorize those characteristics.
The first group is correlated to the matching cost. This means that correct correspondence are unlike to be connected to large matching costs. The second group focuses on local properties of the cost curve. That is, confidence measure becomes the curvature around the minimum matching cost. For example, flat curves, which have smaller values, and describe texture-less areas, express higher ambiguity. Features related to local minima of the cost curve form the third group. As an example, the so called *Peak Ratio (PKR)*, is calculated as the minimum matching cost divided by the second local minima. Then, a probability mass function over disparity defined using the entire cost curve describes the fourth group. The last group includes features that highlight the consistency between left and right disparity map, i.e. correct matches indicate more consistent disparity.
A similar but more recent work is the one carried out in [29] driven by the deep learning breakthrough lately happened in the computer vision area. In fact, changes such as, availability of bigger and more challenging datasets, novel and more accurate stereo algorithms and confidence measures, leverage techniques based on deep learning. Therefore, the authors of [29] focus on implement a complete and updated quantitative analysis of the state-of-the-art confidence measures. As a matter of fact, after the analysis performed in [28], major improvements have happened in the computer vision field. Among those the most relevant can be summarized as follows:

- enhanced confidence prediction algorithms based on deep learning [30] and on random-forests [26], [31];
- larger dataset providing challenging indoor and outdoor scenes [24];
- more accurate stereo algorithms and novel implementation of the SGM pipeline [27], [22]

Therefore, in [29], the authors extend and update the taxonomy previously performed in [28], especially focusing on machine learning techniques. Then they aim at evaluating the algorithms' performance over the novel and larger datasets, concentrating on the correlation between the availability of training data and the correctness of confidence measure prediction. Moreover, they estimate the accuracy of the methods when handling new data and calculate the effectiveness of the predictions when included in state-of-the-art pipelines.
Considering that, among the multiple confidence measures proposed, all of them deal with the cost curve and the relationship between left and right image or disparity map, basing on [28], Poggi et al. [29] grouped confidence measures according to their input cues. Specifically, they considered 76 confidence measures, which were then grouped into 8 categories and evaluate them employing three state-of-the-art stereo algorithms and three ground truth datasets: Middlebury 2014 [32], KITTI 2012 [33] and KITTI 2015 [18]. Performing that exhaustive analysis, the authors carried out the results that learning based approaches seem to be more efficient than conventional ones. Especially, using disparity maps as input cues more accurate results are achieved in terms of correct matches, adaptation to new data and stereo accuracy improvement. Beside that, training remains an issue for those methods, though. As a matter of fact, for deep learning based approaches the general amount of training data is still limited and in most cases they struggle when dealing with new real data.
At this point a broad analysis over the mathematical fundamentals of stereo geometry and rectification has been proposed. Moreover, the main features of both the standard stereo matching algorithms and the newest deep learning based techniques have been widely analysed and explained.
Therefore, in the following sections of this chapter a general introduction about image processing operations is carried out. Then, this extensive analysis on the theoretical basis of this project is concluded with Section 2.5, where edge detection and segmentation techniques are proposed, considering their relevance for the further improvements of the designed algorithm.

2.4 Image processing techniques

In almost all the computer vision methods image processing technique are applied in different phases of the algorithm pipeline in order to model the image in a form convenient for subsequent analysis. Image processing stage is a key component of most of the computer vision applications, such as object recognition, stereo matching, computational photography, scene reconstruction, 3D pose recognition or motion flow, in order to achieve reasonable results. Different types of processing operations are usually applied, depending on the type of task demanded. For example commonly applied procedures includes exposure correction, color balancing, noise reduction, smoothness enhancement, sharpness increasing or feature detection.
In relation to this Master's thesis project, it is worth to focus on some of those operations, which result to be most relevant for the designed algorithm. Especially, point operators, area-based and global image transform techniques will be analysed. Particularly, neighborhood (area based) operators will receive specific attention, being the ones specifically employed in the developed methods.

2.4.1 Point operators

Point operators are defined as the simplest type of image processing transforms [10]. For these operators the output pixel value is entirely related to the corresponding input pixel value. Brightness and contrast adjustment, or color correction and transformation represents some of those techniques.
Basically, an image processing operator is identified, in the continuous domain, by the following function that takes one or more input images and generates a corresponding output:

$$g(x) = w(f(x)) \quad \text{or} \quad g(x) = w(f_0(x), ..., f_n(x)) \tag{2.16}$$

where x is in the D-dimensional domain of the functions and f and g operate over some range. Specifically, for discrete images, the domain comprises a finite number of pixel locations, i.e. $x = (i, j)$, so that:

$$g(i, j) = w(f(i, j)) \tag{2.17}$$

Among these pixel-based operators it is sufficient to cite color transforms, image matting and histogram equalization. Therefore, considering color images as arbitrary vector-valued functions, it is coherent to identify them as highly correlated signals with strong connections to the image formation process.
Taking into account particular image editing application, where for example, the

goal is to take a foreground object from a scene and to put it into a different background. The first step of this processing is called matting, that is cut of a specific object from scene. Whereas, the second part, the positioning over a different background is defined compositing.
Histogram equalization is, rather, a more widely exploited technique. This algorithm is based on the histogram of the individual color channels and luminance values. Thus, using information about minimum, maximum and average intensity values of the image it is possible to equalize the pixel intensity value of the whole image, that is identify the intensity mapping function $f(I)$ such that the resulting histogram is flat.

2.4.2 Neighborhood operator

Differently from the previous type of operators, in this case the output pixel value is evaluated on the basis of its neighboring pixel values. This category of operators is usually applied for local tone adjustment, but more specifically for multiple kinds of image filtering, such as blurring, sharpening, feature detection or noise removal. Considering these local transforms, it is important to distinguish between linear and non-linear filtering operators.
Linear operators are the most generally used in terms of area-based operators. They relate weighted combinations of pixels in a neighborhood in order to estimate the output pixel's value. Basically, they can be described by the following function:

$$g(i,j) = \sum_{k,l} f(i+k, j+l)w(k,l) \tag{2.18}$$

where the entries in the weight kernel $w(k,l)$ are called *filter coefficients*. More simply, the relation 2.18 can be written as:

$$g = f \otimes w \tag{2.19}$$

An important mention has to be done in relation to the separable filters. As a matter of fact, when filters are applied to a 2D image and convolution is taken into account, that requires L^2 operations per pixels, i.e. multiplication and summation, defining L as the kernel size. In most cases, the filtering process can be accelerated carrying out a one-dimensional horizontal convolution and subsequently a vertical convolution. This procedure requires, in fact, only $2L$ operations per pixels. However, this cannot be always done, but it can be applied only to the convolution kernels that are said to be ***separable***.
The proof of that property for a convolution kernel can be achieved by inspecting at its analytic form, as shown in [34]. More precisely that can also be checked handling the 2D kernel as a 2D matrix $\mathbf{K}$ and taking its singular value decomposition

(SVD), as follows:

$$\mathbf{K} = \sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^\top \tag{2.20}$$

Therefore, it is proved that if only the first singular value σ_0 is non-zero, the kernel is separable and $\sqrt{\sigma_0}u_0$ and $\sqrt{\sigma_0}v_0^\top$ give the vertical and horizontal kernels respectively [10]. Among linear filters, there are some that are commonly used for pre and post processing operations over images, which were tested in this project too.
The moving average or box filter is regarded as the simplest one. It basically performs a convolution over the image with a kernel of all ones and the result is then scaled. The bilinear kernel is, instead, a specific version of the *Bartlett* filter. Actually, the bilinear filter is the outer product of two linear splines.
Useful kernel for accurate noise removal are the *Gaussian* kernels. As the operators introduced previously, they are example of low-pass kernels, whose effect is softening higher frequencies, which are correlated with the noise components of the signals[2].
Taking into account the non-linear filters, they are usually exploited when dealing with the need of achieving more accurate results with respect to the outcome given by linear operators. Moreover, on top of that, there are the morphological operators, which are neighborhood kernels working with binary images or thresholded sections of normal images.
Focusing on the non-linear operators, some of them has been employed in the designed of this project, in particular in the image pre-processing step of the main pipeline and in the post-processing phase. Among all the available non-linear area-based kernels, it worth to mention the applied ones.
The median filter is one of these. Basically, it takes the median value from each pixel's neighborhood. This kind of filter becomes quite performing when dealing with shot noise, which is difficult to be removed if applied a standard Gaussian kernel. However, moderate computational cost and the fact that it modify the values pixel-wise are drawbacks of this filter, which has to be taken into account. For this reason, a *weighted* median filter can be implied, instead. In this case, each pixel of the subregion is counted a different number of times depending on its distance from the center. Therefore, this can be equivalently formulated as the minimization of the weighted objective function:

$$\sum_{k,l} w(k,l)|f(i+k, j+l) - g(i,j)|^p \tag{2.21}$$

where $g(i,j)$ is the requested output value and $p = 1$ for the weighted median.

[2]In the Fourier frequency-space the noise appears to be a high frequency signal

Figure 2.5: Original test image for filter comparison. Shot noise is added on top of the image. Credits *Computer Vision: Algorithms and Applications* [10]

Besides its heavier computational cost, compared to the linear filtering, non-linear noise removing operators are usually preferred because of their more accurate *edge preserving* capability. Hence, when cleaning away noisy frequencies, they tend to soft less the edges.

Figure 2.5 shows a test image where shot noise is added. Thus, as visible in Figure 2.6(a), the Gaussian filter, trying to cancel most of the noise, tends to flatten high-frequency details, which are localized close to strong edges. Contrarily, the median filter, shown in Figure 2.6(b), is more edge preserving.
Hence, the median filter was chosen in the initial part of the pipeline of this project for the image pre-processing phase, especially for its capability of preserving the edges, without, then, smoothing away the discontinuities.
Bilateral filtering is another type of non-linear operator, which was tested during the development of the designed algorithm. Basically, during the image pre-processing implementation this specific kernel was one of the analysed options. As a matter of fact, it is an edge preserving filter as the median kernel. For this reason, it is one of the most common option for noise cancelling in computer vision algorithm. Technically speaking, in the bilateral filter, the value of each transformed pixel is a weighted combination of its neighboring pixel values, as defined by the following equality:

$$g(i,j) = \frac{\sum_{k,l} f(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)} \tag{2.22}$$

where the coefficient $w(i,j,k,l)$ is correlated to the multiplication between a *domain kernel*, defined in equation 2.23, and a data-based *range kernel*, illustrate

(a) Gaussian filtered test image (b) Median filtered test image

Figure 2.6: Test images example taken from *Computer Vision: Algorithms and Applications* [10]

by equation 2.24.

$$d(i,j,k,l) = \exp\Big(-\frac{(i-k)^2+(j-l)^2}{2\sigma_d^2}\Big) \tag{2.23}$$

$$r(i,j,k,l) = \exp\Big(-\frac{\|f(i,j)-f(k,l)\|^2}{2\sigma_r^2}\Big) \tag{2.24}$$

Therefore, the *bilateral weight function* can be finally expressed as:

$$w(i,j,k,l) = \exp\Big(-\frac{(i-k)^2+(j-l)^2}{2\sigma_d^2} - \frac{\|f(i,j)-f(k,l)\|^2}{2\sigma_r^2}\Big) \tag{2.25}$$

However, an unavoidable drawback of the bilateral filter is closely correlated to its computational time, if compared to regular separable filter. For this reason, as outlined in [10], multiple acceleration methods have been developed during the last decade. Nonetheless, those algorithms are prone to a higher memory used w.r.t. the regular filtering, thus they should not be applied to full-color image filtering.
A last family of non-linear operators that should be introduce in this section consists of morphological filters. Regarding the designed algorithm, morphological operators have been actually implemented in the post-processing phase in order to remove small estimation error in the 3D point cloud, enhancing the final dense depth 3D reconstruction of the scene.
From a theoretical point of view, *morphological operations* are typically implemented over binary or thresholded images. This kind of filtering techniques are executed by first convolving the input image with a binary structuring element,

which could have different shapes, and after that the result value is defined basing on the outcome of the convolution.
If a binary image f is considered and m is the morphological kernel, the convolution operation is defined as:

$$s = f \otimes m \tag{2.26}$$

where s is the number of 1s inside each structuring element when shifting through the image. Therefore, designating L, the kernel size, the principal morphological operations, which has been tested in the post-processing phase of the algorithm pipeline, comprise:

- **dilation:** $dilate(f, m) = \theta(s, 0)$
- **erosion:** $erode(f, m) = \theta(s, L)$
- **opening:** $open(f, m) = dilate(erode(f, m), m)$
- **closing:** $close(f, sm) = erode(dilate(f, m), m)$

2.5 Edge detection and segmentation algorithms

Strictly correlated to the latter family of non-linear operation analysed and to the overall performance of the designed algorithm there is the concept of edge detection.
Techniques for finding object boundaries in an image are extremely relevant in computer vision and especially when dealing with stereo matching and disparity estimation. As a matter of fact, object borders coincide with occlusion points in 3D, where the majority of wrong matches takes place (in depth estimation). Moreover, segmentation methods are strongly related to this topic and remarkably useful in the preparatory phase of an accurate stereo matching algorithm.
Specifically to this work, segmentation techniques have not been completely developed. This was due to the fact that the initial data provided by the laser points grid gives already a good amount of information for obtaining an accurate 3D dense disparity map. Beside that, after the evaluation of the first reasonable results, further improvements of the algorithm has been considered for enhancing smoothness and continuity among the estimations. Thus, edge detection and segmentation methods appear to be effective initial operations for multiple reasons, such as, reducing the computational time of the entire pipeline and increasing accuracy and density of the final estimations.
Therefore, it is worth to present a brief but exhaustive theoretical outline of the most common edge detection algorithms designed in the computer vision area,

considering that they will be part of the future improvements, which will be applied to the work developed so far.
From a qualitative point of view edges are usually located in areas where there are changing in color or texture. However, segmenting an image basing on this information is usually complicated and specific methods has been developed for that.
Therefore, for pure edge detection, regions of fast intensity variation are taken into account. Specifically, edges are usually located in an image where there are steep slopes, in terms of intensity. Mathematically, these surface's characteristics can be defined through its gradient, i.e.:

$$\mathbf{J}(\mathbf{x}) = \nabla I(\mathbf{x}) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)(\mathbf{x}) \tag{2.27}$$

where the local gradient vector $\mathbf{J}$ symbolizes the direction of the steepest ascent in terms of intensity. Because the percentage of noise is higher at high frequencies, and considering that derivatives enhance those frequencies, a low-pass filter, e.g. Gaussian, is usually applied before the gradient.
Nevertheless, to make edge detection really efficient, it would be desired to refine that continuous gradient to the single pixel locations on the edge contours. This can be achieved by finding the *local maxima* in the gradient magnitude along its direction. Therefore, the maxima are calculated by taking the derivative of the gradient, that means apply a dot between a gradient operator and the previous result. This will lead to the so called *Laplacian*:

$$L_\sigma(\mathbf{x}) = \nabla \cdot \mathbf{J}_\sigma(\mathbf{x}) = [\nabla^2 G_\sigma](\mathbf{x}) * I(\mathbf{x}) \tag{2.28}$$

where σ symbolizes the Gaussian filter used for the initial image smoothing. This final result is usually called *Laplacian of Gaussian*(LoG) kernel, which is a separable filter. However, commonly a slightly different kernel is employed, the *Difference of Gaussian* (DoG), which gives a similar result.
Hence, once the function $L(\mathbf{x})$ has been defined, its *zero crossing* is computed and so the edge elements can be estimated.
Obviously, if high accuracy is needed, higher-order steerable filters can be exploited.
Furthermore, taking into account the standard DoG kernel, the fundamental parameter is σ, i.e. the filter spatial scale parameters. In fact, it influences the amount of noise on the image and thus the edges scale.
It has already been presented that defining confined edges is convenient for multiple applications, such as in the pre-processing image phase of a stereo matching algorithm. Considering that, combining together these edges, designing a sort of continuous contour, would be even more useful.

For example, if the procedure explained above for defining the edges is taken into account, meaning that the edges were defined through zero crossing, link them together becomes quite simple. As a matter of fact, adjacent edges have the same endpoints. Then, multiple methods exist to encode the edges together forming the contours.
Strongly related to edge detections, there is the topic of segmentation, i.e. group together pixels that share the same characteristic. In computer vision segmentation algorithms are among the first studied and designed topics ([35], [36], [37]) and they are still nowadays a widely analysed problem ([38], [39]). Some basic segmentation techniques related to the morphological filters has already been presented in the Section 2.4.2. In this section, the analysis of some commonly used algorithm is carried out. Since in this field the researches have been numerous simple explanations of some of them, such as *active contours*, *region splitting and merging*, *mean shift*, *normalized cuts* and *binary Markov random fields* are presented.
Since the merely theoretical purpose of this analysis and considering that these specific methods have not been widely tested in this project, it is not necessary to describe in detail all of these algorithms. Therefore, if there would be interest in the mathematical background of the aforementioned techniques, we remind to the manual by Szeliski [10].

Active Contours Active contours [40] method groups together different techniques, such as *snakes* [41], *intelligent scissors* [42] and *level set* algorithms. These methods reach the final solution iteratively thanks to the combination of image and arbitrary user-guidance.

Region splitting and merging Differently from algorithms entirely based on threshold selection and connected components computation, a useful segmentation technique is related to a recursive splitting of the entire image into subregions, which are then merged together hierarchically. Therefore, region splitting and merging is fundamentally focused on multiple stages of split and merge of image areas, at different level of density, mainly basing on region statistics.
An example of this approach is the *watershed* algorithm [43]. The core of this algorithm stands on dividing the whole image into different *catchment basins*, which are identified taking into account all the local minima present and labelling them. In order to make the algorithm suitable also for color images, it is usually applied to the gradient magnitude of the figure.
Moreover, an interesting algorithm, which appeared to be useful as a pre-processing phase of more complex algorithms such as stereo matching, optic flow and object recognition, is centred in region merging. Actually, areas of the image close among

each other are combined together considering their average color difference. Therefore, if it is below a certain threshold the adjacent portions of the image are merged into a so-called *superpixel* [44].

Mean shift Mean-shift algorithms, as well as mode finding, k-means and mixture of Gaussian, consider position, color or other features of image points and cluster them as they would be taken from a probability density function, finding the peaks of that distribution.

Normalized Cuts The idea of the Shi and Maki algorithm [45] is to divide the image into groups distinguished by weak affinities, i.e. similarities. Therefore, in this methods pixels that show strong similarities will belong to the same group. The different groups are, then, separated taking into account the sum of the weights of all the cuts among pixels of different regions. Because, this could lead to unreasonable clusters, the normalized cut measure is used.

Markov Random Field Markov Random Field (MRF) optimization is an energy-based algorithm. These energy minimization problems based on MRF are usually solved employing graph cuts techniques.

Chapter 3

Environment

The developed thesis project is based on a major environment that comprises multiple structures. First of all, the Middlebury 2014 [32] dataset, which contains detailed stereo pairs, was exploited to perform multiple tests. Moreover, real indoor rectified stereo images taken with the device designed by the company were employed.
In this chapter a brief explanation of the LaDiMo stereo device follows. Furthermore, it is worth to describe the main characteristics of the dataset adopted. As a matter of fact, it results extremely useful during the overall designing of the implementation, obtain visual outcomes of the improvements provided to the code. The ground truth images available in the dataset were especially effective for simulating the behaviour of the company device during the first part of the code implementation.

3.1 Middlebury Dataset

The so called Middlebury 2014 dataset is a high-resolution stereo dataset comprising static indoor scenes and including highly detailed ground-truth disparities. The images were acquired exploiting a novel structured light system. This also consists of updated methods for accurate 2D subpixel correspondence search and self-calibration of cameras and projectors with modelling of lens distortion.
Generally speaking, Scharstein et al. [32] provide 33 new indoor 6-megapixel datasets using their system. Thus, achieving an accuracy in the disparity estimation of 0.2 pixels on most analysed surfaces, they produce demanding challenges for the stereo algorithms developed since that.
That was, in fact, one of their main objective, being the datasets available until their work insufficient in terms of ground truth accuracy, resolution, complexity and realism. Therefore, aiming at updating and improving the work of Scharstein

(a) Left Stereo Image Example (b) Ground Truth Left Image Exaple

Figure 3.1: Middlebury 2014 dataset example - Credits to Middlebury 2014 dataset [32]

and Szeliski [46], they obtained Middlebury 2014 dataset, which brought major improvements in the level of calibration accuracy for stereo images.
Specifically, the system described in [32] comprises the following novel features:

- a stereo equipment made of two digital single-lens reflex (DSLR) cameras and two point-and-shoot cameras;
- a method for robust interpolation of lighting codes and 2D subpixel correspondence search for obtaining accurate floating-point disparities;
- bundle adjustment for calibration and rectification procedures;
- a robust model selection for self-calibration of structured light projectors and lens distortion;
- a so called *imperfect* version of the dataset showing realistic rectification errors.

Figure 3.1 shows an examples of the datasets provided, which comprise the input images, with different levels of exposure, and *perfect* and *imperfect* rectified images with accurate 1D and 2D floating-point disparities, respectively.
In order to obtain accurate high-resolution stereo datasets, the authors of the described work based their idea on establishing ground-truth disparities from the input views. In this manner, calibration problem are prevented and the process can be performed via structured light. However, the drawback of this is that correct disparities can be achieved only for scene points that are non-occluded in both images. Therefore, extending the idea in [46], that is based on a self-calibration of structured light sources from initial non-occluded disparities, they managed to register illumination disparities in half-occluded regions and model projector lens

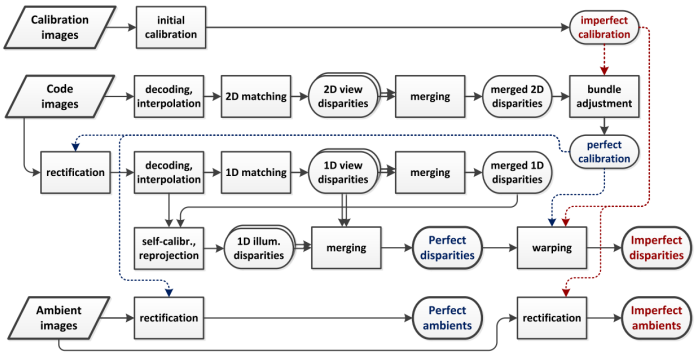


Figure 3.2: Pipeline of the overall system for creating the Middlebury 2014 dataset [32]

distortion as well. Moreover, imposing initial correspondences they enhance significantly the rectification accuracy. Then, subpixel precision was obtained exploiting a high number of binary patterns under multiple exposures and applying a robust interpolation.

A brief overview of the pipeline of the process, deeply illustrated in [32], is now presented.
Figure 3.2 depicts the overall pipeline of the system defined by Scharstein et al. [32] for reconstruct the stereo datasets.
As simply described, the system's inputs are: calibration images of a standard checkerboard calibration target; code images obtained with structured lighting projectors from different positions; *ambient* images collected with multiple lighting conditions. First of all, re-factoring operations are applied to the unrectified code images, i.e. thresholding, decoding and interpolation. This leads to floating-point coordinates of the projector pixel illuminating the scene. Thus, the achieved values are employed as unique identifiers to impose the correspondences between the two input images. So that, the resulting *2D view disparities* are used to refine the *imperfect* calibration in the bundle-adjustment phase. The next step of the process is using the rectified input images to generate the *1D view disparity*. The self-calibration of the projector is, therefore, accessed using the merged disparities and thereby produce the *1D illumination disparities*. View and illumination disparities are, then, combined together into the *perfect* disparities. Last row in Figure 3.2 shows that rectifying with both calibration allows to achieve the corresponding sets of ambient images.
Further details of the single steps of the process are reminded to the original

Figure 3.3: Primary stereo image with LaDiMo grid point overlapped. Image obtained with the *micro* device

paper [32] so as not to make the discussion much convoluted.
Nevertheless, it is worth to disclose that considering the evaluation of the datasets showed in [32], the choice of the Middlebury 2014 images for the initial tests can be assumed as a correct decision. As a matter of fact, experimental results reported in the original paper demonstrate how that datasets can be considered highly challenging in terms of accuracy and scene complexity. Therefore, they can be assumed as a useful starting point for recovering information from the ground truth images available and for the stereo algorithm evaluation.
As a matter of fact, in the area of dense stereo matching methods, in addition to the Mddlebury 2014, other datasets are available, such as the KITTI 2012 [33] and the KITTI 2015 [18], which are among the most commonly employed. However, as for the authors of [47], who developed a similar project, the Middlebury dataset was the preferred choice for the tests accomplished during the algorithm designing. This choice was mainly driven by the high resolution of the dataset, which provides also a desirable subpixel accuracy for the disparity values in the ground truth images. In fact, the KITTI dataset, being generated through a LiDAR assisted device, shows less accurate disparity annotation compared to the Middlebury data, especially in occluded areas.

3.2 Stereo Photogrammetry LaDiMo device

One of the fundamental mainstay of this Master's Thesis project is the device developed by LaDiMo, the company at which this project has been realized. Therefore, it is rather useful for a solid understanding of the work done for this thesis to describe, even briefly, the hardware designed by the company and its implemen-

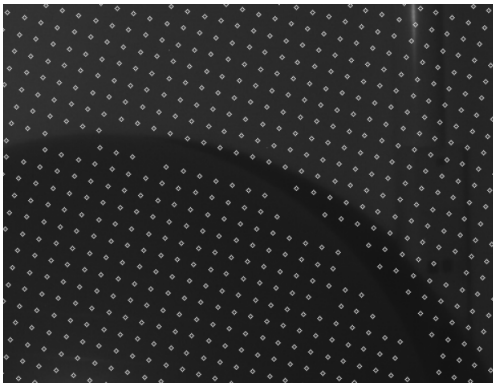


Figure 3.4: Detail of the right hand side of the image, close to the ball, where there is a high percentage of missing data

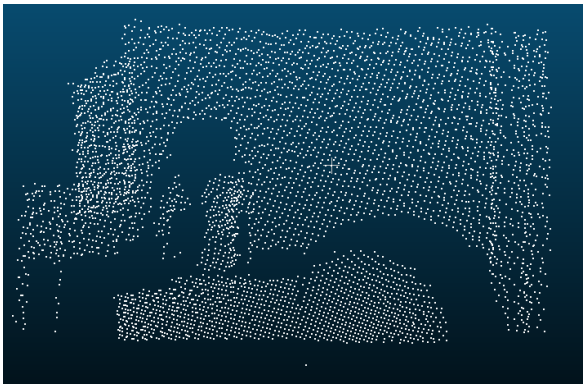


Figure 3.5: Point cloud generated from a test scene

tation in the developed work. Moreover, it is worth to mention that two have actually been the company devices exploited to obtain the real environment images. Specifically, their general structure is the same. The differences between the two employed systems stand in the internal components of the cameras. Thus, the first of these stereo device, known as the *micro* device, is more compact, however it is characterized by a lower accuracy in 3D point detection. On the contrary, the second used systems, defined as the *4-eye* device, reaches a higher accuracy compared to the *micro*, though its dimensions limit the possibility to embed it in a relatively small system, as could be a robotic arm or in a car cockpit.
Thus, generally speaking, LaDiMo device comprises multiple structured light assisted cameras. Its core functionality is to output a 3D point cloud data of the analysed scene for the listening client. From a technical point of view, the beams

are generated by a laser projector, a collimator lens and a diffractive optical element (DOE), which splits the beams in multiple directions over the scene. That set of laser points is then accurately measured by a pair of infra-red cameras sensitive to laser's wavelength, which is specifically of $808nm$. Moreover, two RGB cameras are additionally mounted on the device, so that sets of stereo images of the scene can be retrieved. Therefore, this system allows the user to generate a sparse grid of regularly separated points that are only tilted of almost $16.7°$, which contain information on their 3D location, as visualized in Figure 3.5.
So that, the described device can be exploited in order to improve in both accuracy and efficiency a stereo matching problem. This is, indeed, the base of the study presented in this Master's thesis. In fact, as already deeply described in Chapter 2, one of the main drawbacks of creating a 3D dense point cloud using stereo matching techniques stands in the aggregation cost part. That is, in fact, closely related to the overall range of disparities associated to each stereo pair. Moreover, other key factors are the definition of the image size, the amount of texture-less regions contained in the analysed scene and the number of edges, which make the algorithms used in those processes even more complicated and computationally expensive. However, as visible in Figure 3.5 and 3.3, the laser grid generated with the LaDiMo system cannot be considered enough dense to constitute an accurate 3D representation of the space. As a matter of fact, the points hit only some areas of the object and their sparsification does not allow to determine a continuous and smooth change between the surfaces of different objects, or even between different regions in the same object. Furthermore, as extremely visible in the lower right-hand side of Figure 3.3 and as highlighted in the detail of Figure 3.4, there are sections of the objects that almost completely lacks of points, usually in close proximity of occluded or shaded regions.
Therefore, an efficient strategy that can be applied to tackle the problem of creating a dense depth image from a pair of rectified stereo images is to exploit the initial piece of information contained in the grid points data, in order to make the whole process extremely faster, while keeping a high accuracy of the final data. As a matter of fact, those data can be employed to make the overall pipeline of the algorithm more efficient.
First of all, the error rate regarding wrong matches can be highly reduced, then, the disparity range over which the corresponding points are matched can be recursively selected basing on the disparity values of the neighborhood of the analysed point. So that, these are only some of the improvement that can be applied to the process utilizing the information that comes from the LaDiMo device. Thus, as can be therefore understood, the company's system was fundamental for the efficiency of the overall pipeline of the algorithm. The different steps of the algorithm and the achieved results will be deeply and precisely presented in the following

chapters. As regards the current analysis of the work, it would be sufficient to introduce the general strategy applied and the reasons that lead to the thought strategy.
Taking into account the output of the stereo device, it comprises the two stereo images of the scene, which have to be rectified in order to have the corresponding points along the same scanline for both images, and the grid of points generated from the laser projector and captured by the infra-red cameras, on whose lenses a filter sensible to the laser wavelength is mounted. Regarding the laser grid, as can be understood from the brief system description, it can be correctly recovered for one of the images only. As a matter of fact, the grid points are thus overlapped with the primary image, usually the left one. This configuration is, thus, exploited to run the entire pipeline. Moreover, it would be also possible to define the corresponding grid for the right image, employing the camera matrix parameters. So that, a second reciprocal pipeline could be carried out in parallel. In this manner, two different dense depth map are obtained and then handled to apply specific post-processing procedures such as left-right consistency check, which allow to determine possible wrong matches and especially depth values for occluded pixels. As aforementioned, the corresponding coordinates of the points can be recovered exploiting the camera matrix parameters, which allow to build up a second grid over the support image. However, this phase is not completely necessary for the main purpose of the designed stereo matching algorithm. It will allow to perform specific post-processing procedures. Nevertheless, this has to be evaluated in terms of desired accuracy and computational time, considering that, even simpler post-processing techniques, such as morphological, bilateral and median filters has shown to give an accurate 3D dense point cloud.

3.3 Software Environment

Considering the programming related part of the built environment, it comprises multiple software and platforms. Although, some computing environments and frameworks were not entirely employed during the overall designing of the project. This was due to the way in which the work was conceived and the choices regarding the applicable strategy.
As a matter of fact, in the initial phase different methods and approaches have been tested using MATLAB[1]. Despite the known fact that further working implementation of the algorithm would not be run using an academic version of the aforementioned software, only a first researching phase was initially planned. Multiple reasons support that decision, such as a personal considerable knowledge of

[1]The software was used for academic purposes being available through the personal university account

the software and its built-in libraries and functions, the possibility to check the results of the performed tests in an user-friendly manner and the side own need to get confidence with the company's hardware and software environment.
Taking those aspects into account, the initial part of the work was dedicated to research and tests necessary to evaluate the proper strategy for the algorithm development and to conduct comparisons between some initial ideas over different available methods.

3.3.1 MATLAB computing environment

As introduced above, primary tests have been handled using MATLAB® and especially some built-in functions correlated to stereo matching and image manipulation.
MATLAB can be defined as a platform composed of a multi-paradigm[2] numerical computing environment and a proprietary programming language created by MathWorks. It is significantly used for matrix manipulation, algorithm implementation and plotting of data and functions. Moreover, one of its most utilized packages is Simulink, which allows the user to develop even extremely complex algorithms, especially for dynamic and embedded systems, with a block-based interface.
Although its high versatility and possibility to design programs that can be run of different systems, it was decided not to make it the main environment of the project, in order to not be limited from both software and patent perspectives.
Consequently, after the initial preparatory phase of the project, the primary development of the algorithm has been realized through a C++ code, which has been built up using Microsoft Visual Studio and the OpenCV library.

3.3.2 Microsoft Visual Studio integrated development environment

Microsoft Visual Studio is an integrated development environment (IDE) crated by Microsoft. It is used for several purposes, such as software development, websites, web services, mobile as well as web applications. Obviously, it is based on Microsoft software development platforms, e.g. Windows API, Windows Form and Windows Presentation Foundation. It is also advisable to mention some of its assets, which make it the most preferred IDE, in terms of popularity[3].
First of all, it allows to create both native, or machine language, code and man-

[2] functional, imperative, procedural, object-oriented, array

[3] According to data from Google Trends and available at `https://pypl.github.io/IDE.html`

aged code. Moreover, among its most powerful features there are IntelliSense, the intelligent code completion included in the editor, and the integrated debugger, which operates both as source-level debugger and a machine-level debugger. Furthermore, it contains lot of other built-in tools, designed for multiple applications and objectives, plug-ins that enhance coding capabilities and it supports almost any programming language.
Considering specifically its architecture, it actually does not support inherently any programming language, solution or tool. Rather, it permits the integration of functionalities coded as VSPackage, i.e. Visual Studio Package, which are software modules that broaden the Visual Studio IDE by supplying UI elements, services, projects, editors and designers. Therefore, at its native behaviour, it works as a *Service* solution. As an example, the support for programming language is added by using a specific VSPackage knows as *Language Service*. Exploiting that service various functionalities can be included in the IDE.

3.3.3 C++ programming language

Subsequently to the initial preparatory phase, in which the main employed environment was MATLAB, the actual design of the final algorithm has been developed basing on the C++ syntax.
The decision of exploiting MATLAB only for the initial planning of the work and for the first tests correlated to the feasibility of the various ideas was mainly due to multiple reasons, which cover different aspects of the entire design of the project. First of all, since the beginning there was the intention of designing an algorithm that would be adaptable to different types of hardware and which would be further adjusted to other programming languages, e.g. CUDA, in order to obtain better final performances. Moreover, considering plausible forthcoming commercial issues, there was the need to ensure a developing as free from copyright as possible. Flexibility and efficiency were other fundamental drivers of that choice. As widely know in the technological field, C++ programming language can ensure velocity and efficiency of the designed software as well as flexibility with different types of hardware. In fact, it can be optimal for coding both at high levels of abstraction, and at low levels, e.g. at machine language level. Then, in relation with this very project implementation, using C++ was possible to widely exploit the functions of the OpenCV library.
Therefore, adopting C++ as the fundamental programming language of this Master's Thesis project, it was possible to reach a sufficient level of computational speed, trying to minimize the memory requirements and simultaneously keep a high amount of abstraction, necessary for working effectively in the computer vision field.

3.3.4 OpenCV cross-platform library

OpenCV, Open Source Computer Vision Library, is a library of programming algorithms primarily intended at real-time computer vision.
It was employed as a fundamental support structure of this project mainly because of the functions accessible in that library, the fact that it is open source and the remarkable documentation available. Moreover, thanks to those algorithms, the implementation of methods and functions inside the project pipeline became faster, more reliable and multiple pre-processing and post-processing techniques has been tested, following structural changing in the main algorithm structure.
Therefore, OpenCV library can be considered as an important resource of this work. For that, and for more theoretical purposes, it is appropriate to introduce the main features of that computer vision based library.
First of all, OpenCV was initially developed by Intel, then later maintained by Willow Garage, a robotic research lab and technology incubator and subsequently by Itseez, currently acquired by Intel. Precisely, the library is cross-platform and open-source, under BSD, Berkeley Software Distribution, license. The library was initially created to advance CPU-intensive applications, related to real-time tracing and 3D display walls project. Considering that, it is therefore clear why it is widely exploited when dealing with computer vision correlated works.
As defined in [48], the main targets of the initial project were:

- Accelerate vision research providing open and optimized code for essential vision framework.
- Aim for a more readable and interchangeable code by providing a universal infrastructure to use.
- Enhance vision-based commercial projects by producing a portable and optimized code available with a free license.

Currently, OpenCV library is one of the most used platform by computer software companies. As a matter of fact, it has been calculated that it counts more than 47 thousand people of user community and an evaluated number of downloads surpassing 18 million [4].
Thus, it has been chosen as one of the main assets of the developed work, because of the aforementioned reasons, and in relation to the available applications, which can be used in multiple computer vision related areas, such as, 2D and 3D feature toolkits, facial recognition systems, mobile robotics, object detection, segmentation and recognition and diverse machine learning fields.

[4]Data from `https://opencv.org/about/`

3.3.5 CloudCompare 3D editing and processing software

CloudCompare is a software that was exploited during this project almost exclusively for the accurate 3D point cloud visualization, which it can easily provide. Therefore, it can be considered as a side software used in this work, which helps to get coherent visualizations of the result, things that results difficult or over-complicated employing OpenCV functions and C++. On the contrary, with this software was easy to visualize the computed results, and deeply understand if the modifications made in the code work correctly, improving the final result.
Therefore, to provide a brief presentation of this software, it is basically a 3D point cloud processing software. Originally, it was developed to operate comparisons between two dense 3D points clouds, such as the ones collected with laser scanner, or between a point cloud and a triangular mesh. On top of that, it has been further expanded to a more multi-purposes point cloud processing software, providing multiple advanced algorithms for data handling and processing.

Chapter 4

Methods

A properly planned project requires the analysis of multiple methods and strategies that can be applied during its design. Therefore, this chapter displays, focusing on the multiple aspects of the entire project, the different techniques, encountered revising the broad literature accessible in this field, and the decisions that brought to the actual development of the algorithm, which will be thoroughly described in the Chapter 5.
In this chapter, an extensive outline of the work made is not produced, though. Instead, the following sections focus on a concise, yet exhaustive, tracing of the methods available for a correct dense disparity estimation and for image data processing. Concurrently, the choices made for the specific aspect of the algorithm are defined and explained.

4.1 Standard disparity estimation methods

An extensive and broad analysis of the standard and the deep learning methods for dense disparity estimation has already been presented in Chapter 2, where the review of the literature related to this Master's thesis project has been carried out. Nevertheless, these sections focus on a different aspect of that analysis, which is correlated to the algorithms that turned out to be the most relevant for this work and to the features of some of the current machine learning based techniques, which appear to be applicable to this project.
Considering the standard algorithms for recovering a dense depth image from a stereo pair, the initial approach on which it was decided to focus on for the designing of the project refers to the Semi-Global Matching (SGM) stereo method developed by Hirschmüller [5]. The decision of starting to concentrate on that evaluation of the stereo matching problem was mainly driven by the recent approaches designed for dense disparity estimation. As a matter of fact, most of

the algorithms created in the last decade are based on the Hirschmüller idea or, at least, they make reference to several aspects of the functions outlined in [5]. This fact is clearly comprehensible considering that the SGM method was classified as the best algorithm at the time of its publication and it is currently one of the top-ranked algorithm in terms of sub-pixel accuracy and computational time. Hence, taking into account the benchmark data based on two of the most employed dataset, i.e. Middlebury [32] and KITTI [33] datasets, and after an initial general review of the literature publish in this field, it results that the top-performing algorithms use key features of the SGM in their main pipeline. Moreover, it is worth to point out that the majority of the most accurate real time deep learning based algorithm tends to adopt the main functions of the SGM model, exploiting neural networks to ensure real time evaluations of the disparities. Therefore, it was initially though that a reasonable option for the development of an algorithm, which would work in real time providing accurate outcomes, would focus the attention of the Hirschmüller method.
As already introduced in Section 2.3, the SGM method combines the idea of the standard local based algorithms, where the main cost calculations are made over limited size windows, and global algorithm, for which the best pixel disparity estimation is based on a global energy function. Thus, mainly because of its hierarchical cost calculation, SGM method outperformed the top-ranked algorithm over the dataset [46], proving real-time performance over those images. However, with the current dataset, such as Middlebury 2014 [32], KITTI 2012 [33] and KITTI 2015 [18], the standard SGM algorithm provide optimal results in terms of accuracy but it tends to slightly suffer if computational time is considered. For this reason, the most recent real time methods for dense disparity estimation establish the cost computation part of their pipeline over the SGM idea, although, they exploit hardware efficiency or even neural network structures to reach extremely fast computations.
Thus, considering this project, it was decided to initially build up the main pipeline on the SGM method structure and employ the stereo device designed by the company to reach real time performance, without loosing in accuracy. Furthermore, as previously anticipated, an upstream choice was made. As a matter of fact, the current top-performing algorithms strongly depend on convolutional and deep neural network to achieve fast computations. Contrarily to that, we decided to based the skeleton of our method on the data available through the stereo system to reach a real time implementation, similarly to what done in [47]. In this manner, all the problem related to the shifting between synthetic and real environment, faced by deep learning algorithms, would be avoided.
The first outline of the algorithm was, hence, partially based on the Semi-Global Matching method [5]. In order to analyse the suitability of the implementation,

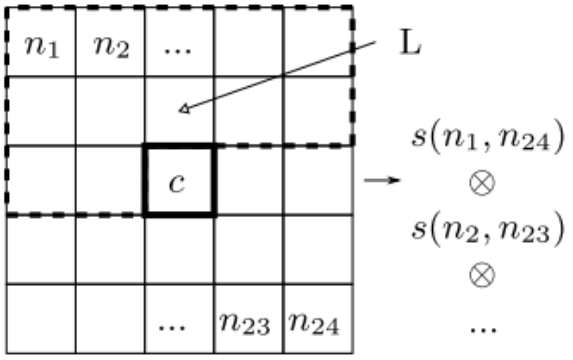


Figure 4.1: Center-Symmetric Census transform schema for a 5×5 image region. Credits to [52]

absolute computational time performances were not initially taken into account. At first it was chosen to focus on the relative efficiency that would be estimated among the different methods used in the principal pipeline.
As aforementioned in Chapter 3, these preliminary evaluations has been developed in MATLAB. In that phase an implementation sufficiently similar to the standard SGM method was designed. Thus, the focus was centred on the different types of algorithms used for matching the corresponding pixels and, then, on the part of the pipeline related to the matching and the aggregation costs, as precisely defined by Scharstein and Szeliski in [2]. The employment of this working strategy was due to the following motivations. First of all, the predominant research aim of this project, whose purpose would only later shifts in more a market based perspective, then the need of testing the effective performance in accuracy of pure SGM based method and the requirement of analysing the relative efficiency between that algorithm and our method, whose pipeline would be based on the information acquired from the company stereo device.
A fundamental part of this initial testing phase was the focus on the matching cost evaluation, therefore, multiple matching cost algorithm has been tested in order to find the most efficient one mainly in terms of accuracy. Therefore, basing on the work done in [49], [50] and [6], and making reference to the algorithm used for computing visual correspondence described in [13] and [51], the performance of different matching cost algorithms has been analysed.

4.1.1 Census transform

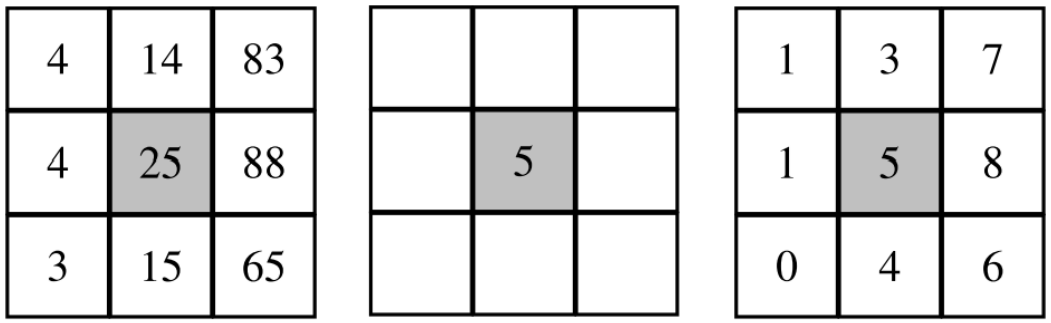
A crucial phase of the SGM algorithm stands in the matching cost computation. As briefly explained in Chapter 2, this cost is evaluated by computing the differ-

ence in the value of intensity among corresponding pixels. Therefore, especially when dealing with unexpected luminance variation inside the analysed scene, it comes out that non-parametric local transforms are optimal as the support for that correlation [13]. As a matter of fact, these kind of measurements do not rely on the actual intensity values of the pixels, but on the relative local intensities instead. Therefore, using this kind of models, a considerable number of outliers can be tolerated and improved results are visible especially along object boundaries.
Among the commonly used and best performing non-parametric local transforms, Census and Rank transform are described. Moreover, couple of different implementations of those two classes of transform are introduced.
It is widely known in computer vision field that correspondence problem is crucial in depth estimation, when starting from stereo pairs. Correspondence can be, thus, defined by transforming the images with a specific method and then establishing correlations. In that, the key factor is the transformation, which must tolerate unexpected variation in intensity, for example due to unwanted light source, and in general changes in image bias and gain.
Therefore, non-parametric area-based transform rely on local ordering of intensities and not on the actual intensity values. If an image is considered, and defining as p a random pixel, its intensity becomes $I(p)$. Then, let $N_d(p)$ be the set of pixel in a certain square neighborhood of size d, where p is the central pixel. Hence, the binary function $\beta(p, p')$ takes value 1 if $I(p') < I(p)$, 0 otherwise. A non-parametric local transform only rely on the set of pixels in the specific neighborhood. Therefore, the Census transform, $C_\tau(p)$ maps the image subregion surrounding p to an array of bits representing the set of local pixels whose intensity is less or equal than that of p. Mathematically, the Census transform can be formulated as:

$$C_\tau(p) = \bigotimes_{[i,j]\in D} \beta(p, p + [i, j]) \tag{4.1}$$

where, $N(p) = p \oplus D$, with $\oplus$ is the Minkowski sum and D the set of displacements, and denoting with $\otimes$ the concatenation. After the Census transform has been applied, the actual comparison among corresponding pixels in done by computing their *Hamming* distance.
Higher accuracy in corresponding pixel matching can be achieved using an extension of the Census transform, defined as Center-Symmetric Census transform [52]. Therefore, if a square image subregion is taken into account, which has dimensions $n \times m$, we already know from equation 4.1 that $s(p, p') = 0, if p \leq p', 1$ otherwise, where now $s()$ is the sign function. Precisely, in this notation the Census transform can be thus defined as:

$$CT_{m,n}(x, y) = \bigotimes_{i=-n'}^{n'} \bigotimes_{j=-m'}^{m'} s(I(x, y), I(x + i), y + j) \tag{4.2}$$



(a) Grayscale pixel intensity values (b) Rank transform (c) Complete Rank transform

Figure 4.2: Example patch with grayscale pixel intensity values. Credits to [51]

where $n' = [n/2]$, and $m' = [m/2]$. Hence, as shown in Figure 4.1, the Center-Symmetric Census transform can be mathematically defined as:

$$CS - CT_{m,n}(x,y) = \bigotimes_{(i,j)\in L} s(I(x-i, y-j), I(x+i, y+j)) \tag{4.3}$$

where $L = L_1 \cup L_2$, $L_1 = R_{-n',0} \times R_{-m',0} \setminus (0,0)$, $L_2 = R_{1,n'} \times R_{-m',1}$ and $R_{a,b} = x \in \mathbb{Z} | a \leq x \leq b$. Because, only Center-Symmetric pairs of pixels are compared, the formulation in equation 4.3 needs less bits than the one in 4.2 to describe the same patch. As aforementioned, the actual matching cost is then computed using the Hamming distance between corresponding pixels.

4.1.2 Rank Transform

Another patch-based pixel descriptor is the Rank transform. As the Census, it is morphologically invariant. Therefore, it is not sensitive to any monotonically increasing of the grayscale intensity values.
The definition of this measure is, actually, quite simple. The pixels in the specific patch considered are ordered basing on the ranking of all grayscale values in that area. Then, the reference pixel *Rank encoding* is determined by counting the number of neighboring pixels with a gray intensity value smaller than the considered pixel, that is, basically, its position in the intensity ranking of the patch. Figure 4.2 shows a 3×3 example patch where the central pixel, i.e. the reference one, is map to its scalar rank encode. This patch-based descriptor, as the Census transform, was introduced by Zabih and Woodfill [13] and, defining an image pixel with p and its neighborhood with $N(p)$, it can be mathematically expressed by the following equality:

$$R_\tau(p) = \|p' \in N(p)|I(p') < I(p)\| \tag{4.4}$$

Looking at the mathematics of the Census and the Rank transforms, it is evident that the former encodes more information.
In addition to the two presented non-parametric local transforms, there is another area based descriptor that need to be mentioned.
The Complete Rank transform can be thought as an improvement of the Rank transform. It was more recently presented by Demetz et al. [51], with the goal of keep a higher amount of information with respect to its standard version. Actually, it works in the following manner. At first the normal Rank transform is applied to all the elements of the patch. Then, these ranks are concatenated, as it is done for the Census, so that the Complete Rank descriptor is generated. As visible in Figure 4.2, pixels with the same intensity take the same rank value. Therefore, the encoding for the patch displayed in the image is:

$$\mathbf{enc}_{CRT} = (1, 3, 7, 1, 5, 8, 0, 4, 6)^\top \tag{4.5}$$

4.1.3 Conventional intensity based methods

Then simplifier but generally equally effective methods are the conventional intensity based operations, such as, sum of absolute difference (SAD), sum of squared difference (SSD), normalized cross correlation (NCC).
Specifically to this project, the first two algorithms have been implemented and tested in the derivative based approach. As a matter of fact, they resulted as fundamental when evaluating the best disparity value among the four possible estimations generated by exploiting the derivatives.
Therefore, in relation to the analysis performed during the development of the project, the best performing strategy, considering accuracy and computational time, was actually the simple yet effective absolute difference. As a matter of fact, tests performed over different images proved this basic operation as enough accurate for selecting the most accurate disparity value for the new grid point, among the four possible guesses produced exploiting the local derivatives related to the four subregion corners.
However, it appears to be beneficial to describe the aforementioned algorithms for comparing the pixel intensity values, thus to obtain the best match. Therefore, the explanation of SAD, SSD and NCC follows.
Sum of Absolute Difference is determined by defining a window of specific dimension within the two considered images. Then, the differences among the intensity values of the corresponding pixels inside the defined region are evaluated, as equation 4.6 shows.

$$SAD = \sum_D (I_l(x, y) - I_r(x, y - d)) \tag{4.6}$$

where D is the defined image subregion, x and y the pixel coordinates in the left image, d the disparity and $I(\cdot, \cdot)$ points to the intensity value. Interesting features of this algorithm are that it is computationally light, however it is generally prone to outliers.
Similarly, the Sum of Squared Difference, the calculated differences inside the window are, then, squared and aggregated. This allows to achieve a relatively higher accuracy, however the complexity rises, due to the multiple multiplications involved, as proved by equation 4.7.

$$SSD = \sum_{D} (I_l(x, y) - I_r(x, y - d))^2 \tag{4.7}$$

Then, considering the Normalized Cross Correlation method. A window of defined size slides over the image. The correspondence is evaluated by computing the normalized sum of the product of the intensities in the subregion and dividing it by the standard deviation of the image intensities. Equation 4.8 explains mathematically this concept.

$$NCC = \frac{\sum_D (I_l(x, y) \cdot I_r(x, y - d))}{\sqrt{\sum_D I_l^2(x, y) \cdot \sum_D I_r^2(x, y - d)}} \tag{4.8}$$

4.2 Deep-learning based methods

As introduced in section 2.3.2 the deep learning method approach for stereo matching was not chosen as the base strategy for the development of this project for different reasons, which have already been explained in Chapter 2.
However, the structure of a deep learning based project was extremely useful in the definition of the approach used for the designing of our work. Fundamental is the novel stereo matching method developed by Poggi et al. [3], who exploited a small quantity of sparse initial depth data to enhance the overall stereo matching problem. In their paper, they presented ***Guided Stereo Matching***, an innovative stereo algorithm, which uses a sparse set of accurate depth measurement obtained from reliable hardware system, e.g. a LiDAR, with the aim of improving the general performance of a standard depth estimation algorithm.
Therefore, considering the structure of our system, the strategy used in [3] appears to have valuable insights, which would be exploited in the definition of our own work. Actually, they proposed to use sparsify yet reliable depth measurements for the following tasks: improve the general accuracy of the neural network designed for achieving the depth estimations, reduce the domain shift, i.e. the drops in accuracy that deep learning based methods face when dealing with data from a new environment, and the potential improvement of the result when training the

network starting from that sparse set of inputs.
Similarly to that approach, in this project, we decide to base our overall estimations on the knowledge of the sparse group of 3D measurements coming from the LaDiMo stereo device. However, differently from them, the algorithm described in this thesis uses standard methods in the stereo matching pipeline, which appear to work better for multiple types of environments.
Thus, taking into account the results provided in [3], where the authors demonstrate how their strategy outperforms the current state-of-the-art stereo matching methods, we assumed that a reasonable approach to apply would exploit the initial cues coming from the laser grid of points in order to improve the performance of the main algorithm. As a matter of fact, in their paper, Poggi et al. [3] assert that the initial set of estimations can be obtained with whatever device that can give sparse data.
Consequently, during the initial stages of the development some tests have been performed designing a simulated point grid, in order to have a clear outcome of the reliability of this strategy. Hence, employing the Middlebury 2014 dataset, the initial set of clues was designed taking the sparse depth information from the ground truth images available. Obviously, the depth measurements coming from the dataset can be considered highly accurate, differently from the data from the LaDiMo device, which would most likely be affected by some sort of noise, mainly Gaussian. Nevertheless, it is actually extremely useful to perform these tests in order to figure out since the first phases of the project, the qualitative performance of the adopted strategy. The details of the conducted tests are widely described in Chapter 5, where some of the temporary results obtained are also proposed.

4.3 Pure derivative based data estimation algorithm

As previously introduced, two different algorithm structures have been implemented and compared in this project, in addition to the standard Semi-Global Block Matching (SGBM) algorithm available in the OpenCV library, whose result was used as a baseline for carrying out an appropriate comparison with the novel implementation designed.
Therefore, the first one of these methods is closely related to the standard SGM implementation, it exploits the initial knowledge of the sparse depth estimations, however, the core of the algorithm follows the Hirshmüller algorithm [5]. Differently, the other implementation designed is more strictly related to the initial sparse depth clues, and it is focused on those for the overall pipeline of the algorithm. As it will be reported in Chapter 6, this latter approach will provide

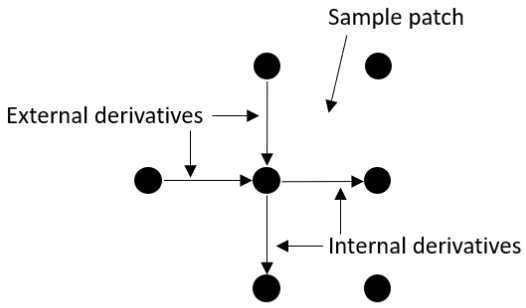


Figure 4.3: Example of a 4-neighbors image patch where the internal and external derivatives related to one of the patch corners are highlighted

a faster yet, in general, equally accurate implementation. Therefore, these two designed strategies will be deeply explained in Chapter 5. Beside that, it is worth to propose here a concise outline of both of them, in order to introduce their main features.
Regarding the last introduced method, it highly exploits the information coming from the input sparse clues. Therefore, starting from that initial 3D point cloud and using a pair of stereo images of the considered scene, a dense point cloud is built by estimating the space positions of an high amount of scene points. The key of this algorithm actually stands on the way in which those estimations are carried out. Thus, considering the point grid given by the stereo device, the local derivatives among neighboring points are evaluated. These will be then used for two main scopes: identify if inside a specific subpatch of the point cloud there could be an edge, meaning that points belonging to the same small cluster have indeed highly different values of depth, and for computing the denser point cloud by interpolating the 3D value of neighboring points and exploiting the values of the pre-calculated derivatives for making the interpolation faster.
It can be, hence, easily understood how core of this strategy stands on the local derivatives calculation. As a matter of fact, let us consider a square cluster of 4 neighboring points from the initial point cloud, it is possible to evaluate the *internal* and the *external* derivatives among these points. Specifically, the former are referred to be identified by the vectors that points *inside* the square patch. Instead, the external derivatives are said to be the ones evaluated using the other points, which are neighbors to the patch corners. Figure 4.3 shows the aforementioned point cloud patch and the calculated derivatives relative to a corner. Summarizing, the denser 3D point cloud is estimated by filling up each one of the subpatches that can be identified from the initial grid. In order to perform this, the estimations are

carried out exploiting the derivative vectors, which are weighted by the distance from the new estimated point and each one of the window corners. This detail, in fact, shows that four temporary estimations are done for each new point. In this way, the best estimation is actually chosen by projecting the evaluated 3D point to the image plane and calculating the matching cost between the corresponding pixels for each temporary estimation. For the matching cost, as it will be more widely explained in Chapter 5, different methods have been investigated. However, the plain absolute difference appeared to be the best in terms of performance for that algorithm. In fact, at the end of the whole computation, similar results have been found with different matching cost algorithms. Hence, in that case, it is worth nothing to use most expensive methods, which will make the computations heavier giving only slightly better results. Besides that, specific penalties have been added on top of the pure matching cost to enhance the distance from the respective window corner in terms of x and y coordinates.
Summing up the general outcome gained from this approach, it results to be a lightweight yet effective method. As it will be presented in the Chapter 6, this algorithm provides 3D dense point clouds that can be considered sufficiently accurate for a general use employing a relatively low amount of resources. Despite that, internal parameters of that method can be adjusted, depending on the user needs, in order to obtain outcomes with a different grade of accuracy. Evidently, this will affect the performance in terms of computational time and used resources, however, this is always considered for every specific environment.

4.4 SGM based data estimation algorithm

As aforementioned, the second method that has been designed is based on a more standard approach, closely related to the SGBM algorithm. Basically, it exploits the information coming from the input point grid to make the standard pipeline of the SGBM method faster, which could be extremely expensive when dealing with high density images. Actually, the initial points information is employed to select the disparity levels on which the main algorithm is run. Simply speaking, the final implementation can be considered as a *disparity-wise* Semi-Global Block Matching, where, usually, there are not continuous changing in the disparity levels, especially if edges are detected inside the specific image patch.
Taking into account the general performance of this approach, it runs slower than the previous strategy. Moreover, the final 3D point cloud shows an higher density, however, looking at the achieved results, the increase in accuracy obtained in this case seems not to be worth the observed rising in computations.
Therefore, at the reached stage of the project, it appears that a slightly decrease in accuracy can be considered reasonable for a quite high increase in computation

speed.
However, every use case has to be widely analysed in order to choose the most profitable solution. Additionally, it is necessary to point out that this project should be regarded as still in a researching phase. Multiple tests have been performed with both dataset, i.e. almost perfect, and real images. However, for a good analysis of the specific use case, it is necessary to run various tests on the specific working environment, and eventually, adjusting parameters or side sections of the main pipeline.
Beside that, a further discussion over those topics will be presented in Chapter 7. Anyhow, it was considered helpful for a deeper understanding of the results, which will be presented in Chapter 6, to introduce also here some brief comments over the outlined algorithms.
Hereafter, a short overview on the pre and post-processing techniques employed in the whole project is presented. Further details will be described in the following chapter. Anyway, it is useful to introduce here the main techniques used and especially the differences in performance, which have been analysed, and also their scope in relation to the final accuracy of the algorithms.

4.5 Pre-processing techniques

In the first section of the algorithm pre-processing techniques have turned out to be, not indispensable, but useful for the accuracy of the final results. The focus was especially centred on the noise removal. As a matter of fact, if a stereo pair is given, it is generally affected by Gaussian, i.e. random, noise, which can be due to multiple reasons, such as instability of environmental conditions, lightning, local temperature gradients that can affect the stereo system or errors in the setting of the stereo device.
Therefore, for those reasons and for the purpose of launching the algorithm having the input image as clearer as possible, especially in terms of pixels' intensity, blurring filters have been tested and analysed during the development of this first phase of the pipeline. Specifically, median and bilateral filters occur to be ideal for smoothing away noisy components in the frequency spectrum, keeping the performance of the whole algorithm reasonable.
The details of that analysis are left to Chapter 5. However, it is worth to indicate that, even if more computationally expensive, the median filter comes out as the preferred choice for image blurring when dealing with real stereo pairs. As a matter of fact, when the images from the datasets have been tested, there was not any needs to apply some sort of pre-processing, being them already cleaned from noisy components.
Contrarily, talking about the real dataset and especially about the initial grid of

points obtained from the LaDiMo stereo device, some pre-processing operations have been performed over those data too. The goal of this process was the requirement of starting the main computations basing on accurate and possibly complete data. Focusing on that, it actually happened that the input point cloud was, in general, not fully complete. Instead, some of the points had missing data. This was probably due to occlusions, in fact most of the *bad* points areas were located on edges or on regions affected by shadows.
Therefore, this problem was tackled in the following way. First of all, all those input data have been analysed and, specifically, the percentage of completely missing values with respect to the total amount of points was calculated. This was performed to obtain an initial estimation of the percentage of errors present in the input data, thus to evaluate the degree of effort that it should be required to have a final accurate result. Precisely, let us think about the case of having an extremely low amount of errors, e.g. the 1.5% of the entire point cloud. Moreover, consider the condition in which these errors are sparsified. In this case would be rather complicated to fill in those missing data. Thus, it would be worth nothing to spend resources to improve the completeness of the input data, reaching an outcome that would be, most likely, only slightly more accurate and for which the whole algorithm would be way computationally heavier and more complicated. On the contrary, a feasible option can be the one of running multiple time a very lightweight algorithm, exploiting the most accurate new estimated points to improve the density of the successive 3D clouds.
For the actual purposes of this chapter it is unnecessary to present further details of this data processing, which would be more widely analysed in the following sections.
On the other hand, it is convenient to introduce here some features about the post-processing operations, which have been carried out in the last section of the algorithm.

4.6 Post-processing techniques

Considering the results achieved with both the Middlebury 2014 dataset and the real environment images an additional blurring process has been run over the final data. Differently from the the smoothing operations employed in the first part of the depth estimation method, in this phase the morphological filters emerge to be the most appropriate trade-off between final result and computational resources.
Also during the designing of this area of the project several tests have been carried out in order to obtain an estimation of the most suitable post-processing technique to run over the final result.
Therefore, considering the disparity images achieved using the stereo pair from the

perfect Middlebury dataset, the application of an additional median filter seemed more appropriate. However, for the outcome accessed using images taken with the LadiMo device, the use of morphological filters appear more convenient. Additionally, several of those have been tested. Opening, closing, erosion and dilation have been run over both the raw disparity result and over the raw denser 3D point cloud. Moreover, in order to have a clear idea of the performance of the different kernels, even mixture of those morphology operations have been generated.
At the end, the application of an opening filter gave the most accurate result on the 3D point cloud.

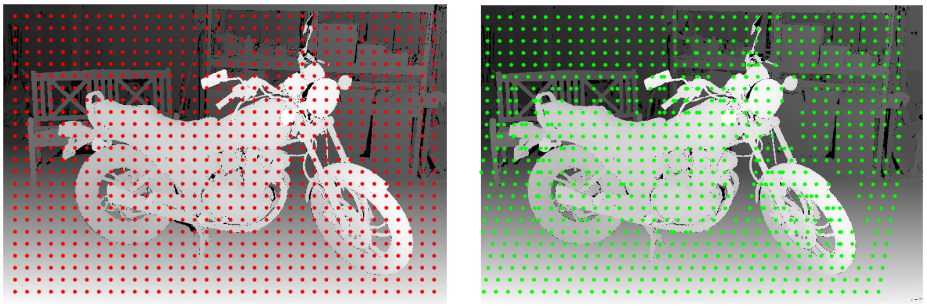
Chapter 5

Algorithms implementation

The actual implementation of the designed algorithms introduced in Chapter 4 is now extensively described. As anticipated in the previous chapter, here all the details of the developed phases of the project are deeply analysed and explained. A path similar to the one followed during the actual designing of the algorithms is used for outlining the whole project.
First of all, the preparatory tests performed for evaluating the feasibility of the conceived idea are reported and explained. Subsequently, the focus moves on the two main methods designed, the derivative based one and the strategy based on the standard SGM pipeline. After that, the side phases of the work are presented. The pre and post-processing operations are accurately defined and the comparisons among the different filtering operations are carried out.
Regarding the description of the designed algorithms outlined in this chapter, some specific details related to the actual code are omitted, thus to not make the explanation too much convoluted. Therefore, pseudocode concerning the main section of the algorithm pipeline is proposed in ChapterA, thus to be checked if further details about the methods are needed.

5.1 Preparatory and testing phase on MATLAB

As anticipated in Chapter 4, the first phase of the project can be mainly considered as a researching stage. As a matter of fact, exploiting the stereo images from the Middlebury 2014 dataset, some tests have been performed to obtain an estimation of the feasibility of the conceived idea.
Therefore, as Figure 5.1 displays, ground truth images have been initially used. Thus, a point grid, similar to the one generated with the LadiMo device, has been simulated and overlapped to the input images. In particular, as the difference between Figure 5.1(a) and Figure 5.1(b) illustrates, to have a reliable simulation of



(a) Left ground truth stereo test image overlapped by a simulated point grid

(b) Right ground truth stereo test image overlapped by a simulated point grid

Figure 5.1: Ground truth images from the Middlebury dataset [32] used for the initial testing phase

the 3D point cloud generated by the device, the only grid that was actually defined was the one over the principal image, i.e. the left one. As a matter of fact, as defined in section 3.2, the hardware exploited in the project is provided with only one laser projector. Therefore, this allows to utilize the produced point cloud only over one image[1]. After that, those input data, together with the grayscale stereo pair, have been employed in a stereo matching pipeline to recover a disparity map of the analysed scene.
In order to acquire useful baseline data regarding the practicability of the thought strategy, tests and analysis over the different areas of the main loop of the algorithm have been investigated. Actually, this research has been performed with a precise method.
First of all the stages of a standard stereo matching method, as defined by Scharstein and Szeliski in [2], have been separately examined. Thus, in order to do this in a reasonable way, the literature regarding both standard and novel stereo matching algorithm has been studied and the main features of the designed algorithm checked out. As an example, if the matching cost phase is taken into account, different approaches are presented in the literature for evaluating the similarity among the image pixel values for the respective disparity levels. As a matter of fact, as introduced in Chapter 4, the Hamming distance between corresponding pixels can be exploited to determine that measure. In relation to that, different algorithms exist to apply a transformation to the input images, making them available for an appropriate implementation of the aforementioned matching cost calculation. For the actual tests carried out, local methods have been mainly anal-

[1]Technically speaking the initial cloud of point created through the device can be independently related to both of the image, depending on which is used as the reference one, just applying the correct transformation between the laser and the principal image coordinates systems

ysed, being more appropriate for the type of data available and for the purposes of the designed strategy.
Therefore, as matching cost measures, simple operations have been initially taken into account, such as the Sum of Absolute Difference (SAD) and the Sum of Squared Difference (SSD). Moreover, more accurate methods have been later evaluated, i.e. the Rank transform, and its improved version, the Complete Rank transform, broadly discussed in [51]. Then, the Census transform and the Center-Symmetric Census transform proposed by Spangenberg et al. in [52] have been investigated. Considering all of those algorithm, it resulted that the most suitable methods are equivalently the Rank and the Census transform. In fact, being these non-parametric transformations, they are not sensitive to changing in lightning inside the environment, thing that usually happen when managing real industrial areas. Considering that, at the end the Center-Symmetric Census transform has been chosen because of its reliability, which is comparable to the aforementioned methods, and for the fact that it uses a lower amount of bits to describe the single pixel value compared to the standard Census transform.
During this phase of testing, the decision was to run a single pipeline of an *SGM-based* algorithm, whose aggregation cost phase would be based on the four orthogonal directions. Moreover, another operative choice was to use the whole image for the complete execution of the designed method. Therefore, it was thought that, at least for this researching phase, it would be meaningless to define a hierarchical implementation of the code, where the image is initially downscaled and subsequently scaled up again in order to make the execution faster, as made by Hermann and Klette in their application of the SGM algorithm [53].
At the end of this *research* section of the project, the results did not show an exact path to follow, meaning that there was not a clear improvement over the normal SGM-based approach by using the initial set of grid point, compared to the MATLAB built-in method used as baseline. Specifically, the disparity image obtained was actually lower in accuracy compared to the method implemented in MATLAB by Eric Psota and available at the website of the Middlebury stereo dataset [2] and to the MATLAB built-in method `disparitySGM()`. Moreover, the developed implementation was quite slow, if correlated to the benchmark of most of the standard SGM-based methods.
However, some considerations should be proposed regarding that outcome. First of all, the algorithm designed runs iteratively, that is no parallel loop is coded to make the execution faster. This is, in fact, explained by the decision of being interested in the relative performance among the own designed methods. Furthermore, any sort of GPU enforcing was not used. Hence, it was quite expected to not obtain good performance. Therefore, the thought strategy was evaluated as

[2] links Middlebury College Stereo

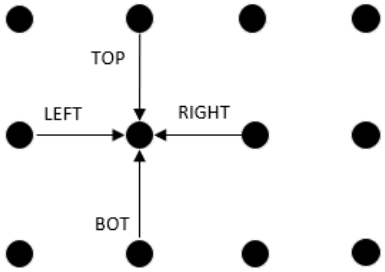


Figure 5.2: Detail of a point grid patch where a set of neighbouring points is highlighted

feasible and, additionally, the exploiting of proper initial data provided by the stereo camera would likely be an enhancement to the overall performance of the algorithm. Moreover, the derivative based algorithm has not been initially tested, however, considering its main idea, it should theoretically give accurate results in a reasonable computational time.

5.2 Derivative based method implementation

Verified the feasibility of the considered strategy, the project design shifted to the actual algorithm development. Therefore, exploiting the OpenCV libraries and using the images from the Middlebury 2014 dataset, the derivative based method started to be implemented in a C++ code.
In the initial phase of its designing, the simulated grid based on the ground truth images was still used to obtain the input point cloud data. As a matter of fact, the only practical advantage of employing such simulated data stands on the amount of noise that they contain. As a matter of fact, an appropriate comment to that strategy would be that, once the real data from the device would be used, the result would probably be worse in terms of overall accuracy. Anyway, at that initial stage of the algorithm development, the interest was focused on comparing the performance of the pure SGM-based and the derivative-based implementations.

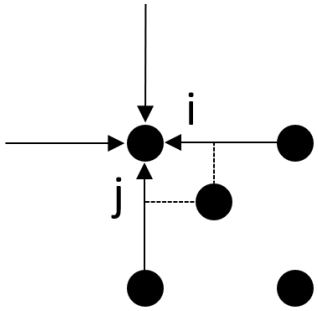


Figure 5.3: Detail of a point grid patch where the estimation of a new point at relative coordinates i and j is performed

5.2.1 General features of the method and simulated grid specifications

The so called derivative-based method, which arises to be both efficient and generally accurate, based its strength on the sparse point cloud gained through the structured light assisted stereo camera.
The main features of that algorithm are now precisely described. Moreover, attention is focused on the conventions utilized to define the main parts of the method and on the functions created for obtaining the 3D point estimations over the initial laser grid. Subsequently, some fundamental sections of the designed implementation are more broadly described, thus to provide to the reader a complete understanding of the work done.
For this reason, the analysis starts by considering a simulated grid of points, which was defined in the C++ code in order to initially mimic the real laser grid generated by the LaDiMo device. In fact, the actual grid of points generated by the company device is not perfectly parallel to a vertical plane, as the simulated one, contrarily it is tilted at 16.7° w.r.t. the vertical axis. However the approach followed, as aforementioned, does not invalidate the outcome obtained through all the designing process of the method. Instead, it is a reasonable strategy for achieving some initial temporary results, which can then provide a guidance to the following phases of the whole process.
Therefore, let us consider the entire point cloud as a regular grid where each point is defined by its coordinate in space. Additionally, it can be also thought that the corresponding 2D coordinates in the reference image plane can be easily retrieved using the camera projection matrices. Hence, if a neighboring set of four points is taken into account, as highlighted by Figure 5.2, the local derivative vectors for all

the considered coordinates can be easily calculated. This estimation of the derivative is extremely useful for the following reasons. First of all, they are, actually, exploited to generate the estimations inside each grid sub-region. Moreover, they can be employed to evaluate if, inside the specific patch of points, there could be an edge of some shape. In this latter case, the magnitude of the Z component of the derivative vector will, in fact, be higher than a set threshold, which can be defined looking at the average values of the Z coordinate of the grid points.
Therefore, explaining the project design procedure implemented, it can be summarized as follows. Initially, for each one of the points in the grid the internal and external derivatives have been calculated with respect to its four neighbors, as the arrows in Figure 5.2 underline. This means that each point will have, then, four distinct set of local derivatives, for both the internal and the external ones, conventionally defined as follows:

- **from top:** derivative between the specific point and its top neighbor;
- **from boottom:** derivative between the specific point and its bottom neighbor;
- **from left:** derivative between the specific point and its left neighbor;
- **from right:** derivative between the specific point and its right neighbor.

In these definition a neighbor to a point is defined with the subsequent rule. Let us consider a point, which is classified through the grid coordinates r, along the row grid direction, and c, along the column direction. Its neighboring points are, hence, identified by:

- **top** neighbor: with coordinates $r - 1$ and c;
- **bottom** neighbor: with coordinates $r + 1$ and c;
- **left** neighbor: with coordinates r and $c - 1$;
- **right** neighbor: with coordinates r and $c + 1$;

Using that scheme the four derivatives estimated for each point will contain the difference between the actual point ***full location components*** and the ones of the relative neighbor. In the implemented code, the introduced ***full location components*** are in the order: the X, Y and Z space coordinates of the point, in meters, the x and y components of the point projected in the principal image plane, therefore they are measured in pixels, and finally, d is the disparity value, in pixel, related to the depth of the point, which can be evaluated through equation 2.1 and adding the relative *disparity offset* if required. This gain was, in fact, necessary

when using the data from the Middlebury dataset, where the relation between the ground truth disparity values and the corresponding depths is defined as in the equality 5.1

$$Z = \frac{B \cdot f}{d + d_{offs}} \tag{5.1}$$

where, as in 2.1, B is the camera baseline, f the respective focal length and d_{offs} the disparity offset, which is the different between the x coordinate of the principal points, as following clarified:

$$d_{offs} = c1_x - c0_x \tag{5.2}$$

where $c1_x$ is the x coordinate of the principal point for the support camera, while $c0_x$ is the x coordinate of the principal point for the main camera.
Therefore, let us now consider to estimate an unknown point inside a subregion of the whole grid, that was decided to be identified by 4 points, as Figure 5.3 displays. The final 3D coordinates of the new point will be, at first, defined by four different estimations, each one related to the four corners of the grid subwindow, as follows.

$$est_{TL} = TL + i \cdot der_{from-left} + j \cdot der_{from-top} \tag{5.3a}$$
$$est_{TR} = TR + (i-1) \cdot der_{from-right} + j \cdot der_{from-top} \tag{5.3b}$$
$$est_{BL} = BL + i \cdot der_{from-left} + (j-1) \cdot der_{from-bottom} \tag{5.3c}$$
$$est_{BR} = BR + (i-1) \cdot der_{from-right} + (j-1) \cdot der_{from-bottom} \tag{5.3d}$$

where TL, TR, BL and BR respectively point to the estimation done with respect to the top-left corner of the grid sub-window, the top-right corner, bottom-left and bottom-right corner. Then, i and j are the relative x and y coordinates of the estimated point, whose values range between 0 and 1. Finally, in the notation employed, der_{dir} stands for the different derivative vectors related to each corner point and visualized by the arrows in Figure 5.3. Here, in order to keep the generality of the notation, no distinction has been made between internal and external derivatives. As a matter of fact, in both cases the calculations performed are actually the same. The difference only interests the specific values of the vectors and the analysed case, considering that the internal derivatives are in general used. The external ones are exploited when some sort of edge is detected inside the analysed patch, after having verified that other edges are not located on the close neighborhood of the targeted subwindow.
Once, these estimations are achieved exploiting the derivatives, as equations in 5.3 define, the best evaluation is obtained by calculating each matching cost between the corresponding pixels between the stereo images. The matching cost is determined following the subsequent routine.

In relation do this, it should be reported that, due to the researching aspect of this project, lots of changing have been made to the code. Therefore, to provide a clear explanation of the decisions made and the procedure followed, which, otherwise, could appear somehow winding, all the relevant changes to the functions used are disclosed. Beside this brief comment about the designed code, the best, among four, estimation of the new grid point is such evaluated.
The estimation provides the values of the 3D coordinates of the calculated point and its disparity. The point location is, then, projected through the camera matrix to the reference image (usually the left image) pixel coordinates and the corresponding position in the support image is determined using the estimated disparity value. At this point the matching cost is computed as the absolute difference between the intensity values of the analysed pixels in the left and in the right image planes. However, this kind of measure appears to give weak results. As a matter of fact, the main problem was that, in some situations, the estimation labelled as the best one actually contained a value of disparity doubtful, if compared to the one of the point's closest neighbors. Moreover, this generally happened in texture-less regions, where it is obviously more difficult to make accurate estimations.
To overcome this problem, a penalty value was added to each partial estimation. Specifically, that value was weighted linearly basing on the distance between the estimated point and the corresponding corner. In this manner, the estimation, related to the disparity and to the coordinate values of the closest corner, is most likely to be addressed as the best one. Once the best estimation for each new point is calculated, the evaluated values of the 3D point coordinate are processed, applying a morphological filter, in order to remove undesired noise and smooth out the overall data.
The estimation procedure just described is not actually able to handle all kind of situations, concerning the provided input objects. In fact, incorrect values used to be estimated in subregions affected by edge features, and thus occlusions.
Therefore, to correctly manage these situations, multiple cases have been defined, regarding the edge cases.

5.2.2 Edge cases and penalty values discussion

At first, four main conditions have been distinguished. These comprise the possibility to do not have any type of edge inside a grid subregions, the existence of a vertical edge in the patch, an horizontal edge case and finally a so called *undefined* case has been defined.
Therefore, in each one of these situations, the point estimation procedure differs slightly, so that the aforementioned linear penalty can be more wisely set. Regarding this two distinct values of the penalty have been employed, a lower and a higher value. In fact, because it is not efficiently possible to correctly estimate

the position of the potential edge, the choice of the correct estimation has to be somehow driven by exploiting the linear penalties.
Moreover, to identify the correct type of edge, the derivatives have been exploited and, specifically, the magnitude of the Z component of the vector. Basically, a general edge case, which is equivalent to the *undefined* condition, is initially verified by computing the absolute difference between the Z components of all the possible couples of corners, i.e. the Z value of the local derivatives, and comparing that with a pre-determined threshold, which is related to the average depth value of the initial points of the grid.
Therefore, during the algorithm pipeline, the possibility of having an edge feature inside the specific patch is initially checked. Then, if the depth values of the four corner are almost equivalent, this means that the area is actually planar. Thus, in this situation, a fast interpolation method is run, so that the overall computation can be even faster.
On the contrary, if a vertical or an horizontal edge is found, the small and big penalties are used accordingly. For example, if the edge is vertical, the small penalty will be multiplied to the relative y coordinate of the estimated point, instead the big penalty will weight the x coordinate.
Finally, for the undefined case, the penalty weight is equivalent in both directions, x and y, and its magnitude ranges between the two previously defined values.
Those three different edge cases are the one initially tested. However, later appeared that a more meaningful strategy should be based on the definition of two grater categories, which would comprise the cases described above. These two bigger classes describe, thus, ***soft*** and ***strong*** edges. In fact, results of tests done with dataset images show that employing a single threshold for the depth difference and focusing only over the edge direction was not entirely adequate. As a matter of fact, in a generic scene different types of objects exists. Therefore, adapting the edge analysis to the object distances, for example between the whole foreground and the background and between multiple objects of the foreground, that would give a smoother and more accurate final dense 3D point cloud.

5.2.3 Derivative computation and sanity check

The core of the estimations are, actually, the local derivatives computed for each point of the grid. Basically, each grid point contains four vectors, which are the local derivatives between the reference point and its four orthogonal neighbors.
Initially the mere derivatives have been calculated as pointing towards the reference point, like Figure 5.2 shows.
However, it came out that this procedure was not enough accurate, to generate a proper dense 3D point cloud. Therefore, a *sanity check* has been later defined inside the derivative calculation pipeline to achieve preciser computation of the

local derivatives, especially for the grid sub-patches where edges are most likely to be found.
Moreover, a higher accuracy of the final result has been achieved when evaluating, for each grid point, two slightly different types of derivative vectors. Specifically, ***internal*** and ***external*** derivatives. This distinction allowed to achieve faster and more accurate results, especially when dealing with edges in the grid subregion. Precisely, if the presence of an edge is verified for the considered patch, the external (with respect to the sub-window) derivatives were used. Otherwise, exploiting the internal derivatives the computation could be made faster, being them equal in couple among the corners of the patch.
Referring to what introduced at the end of Section 5.2.2, in the latest implementation designed, the estimation process is mainly driven by the edges analysis. As aforementioned, the test results demonstrate that a distinction based only upon the edge shape is not enough to reach a good degree of accuracy. Therefore, the analysis cases have later been defined basing on the strength of the edges between the objects in the scene.
Hence two different thresholds have been so determined. The first is used to identify occlusions that basically occur between foreground objects and the background. These have been designated as ***strong*** edges. Differently, the other case is specified as the ***soft*** edges case. This is related to occlusions happen between parts of the same object at different values of depth, as could be between the chin and the neck of a person.
Moreover, on top of this first level of edge selection, the following cases have been then defined, which are correlated to the specific value of the penalty. Precisely, they are *pure vertical* and *pure horizontal* edges, for which it should happen that the depth values of the corners are equals in couple. Then, there are some particular cases, which have the scope of leading the disparity estimation to a higher level of accuracy. They are: *diagonal top left*, *diagonal top right*, *diagonal bottom left*, *diagonal bottom right*. Furthermore, there is the undefined case, which is triggered when the potential edge found in the subregion has a strange shape.
Taking now into account the actual estimation functions, these are slightly different for the aforementioned edge cases identified. Summing up the overall procedure, it can be said that, the main distinction, which guides the whole process is the first one, that is the one that distinguish among, strong, soft and no edges. Then, for all the subcases, the main difference stands on the values and order of the penalties.
Hence, in the *no edges* case, four temporary estimations are performed for each new grid point by carrying out a linear interpolation, which exploit the *internal* derivative vectors. Then, for the other two main cases, the procedure is generally the same. In fact, the fundamental scope of distinguish between those types of

occlusions is to make the final outcome smoother.
Thus, when dealing with edges, at first the algorithm checks for each temporary estimation, that is for each subregion corner, if there could also be edges in the neighborhood outside of the region. If this is false, a linear interpolation is computed employing the *external* derivatives. Otherwise, the interpolation is done with the *internal* derivatives and only for the x and y pixel positions. Instead, the depth value is matched with the one of the corresponding corner, thus to enhance the smoothness of the final result. Regarding the X, Y and Z values, these can be easily recovered from the previously estimated pixel coordinates and disparity, by exploiting the camera projection matrices and the relative equations.

5.3 Semi-Global Matching based method implementation

The other implemented method based on the initial grid of points follows more closely the standard stereo-matching structure. Moreover, it does not distinguish between ***strong*** and ***soft*** edges, but the different cases are identified only considering the shape of the potential edge present inside the subwindow.
Therefore, this algorithm initially computes the *matching cube cost* for the specific grid patch. The difference between the standard SGM method stands on that cube, whose third direction does not generally contains continuous values of the disparity. On the contrary, those values are defined by the sub-window corners, whose disparity define the levels of the cost cube.
Therefore, if no edge is found, only the matching cost part of the stereo algorithm is carried out. Otherwise, the aggregation cost phase is implemented. In this latter case, the cost is actually aggregated only on the direction orthogonal to the shape of identified edge, thus to make the implementation faster. As a matter of fact, let us consider a specific sub-patch in the principal image and the pixels that belong to it. Let us, then, assume to identify a vertical edge inside that subregion. It would be worth nothing to aggregate the matching costs related to each pixel along the vertical direction. The differences among the intensity values of the pixel along that direction would, in fact, be extremely lower than the variations along the orthogonal direction. Therefore, because in general the matching costs are aggregated along all of the four orthogonal directions, the values along the path orthogonal to the edge will weight more. While the costs along the path parallel to the edge will have less relevance, technically speaking. Therefore, it would be worth nothing to aggregate the cost along all the four directions, obtaining as the only outcome an increasing in the algorithm complexity. On the contrary, if a particular edge is determined inside an image patch, it would be more efficient to

run the aggregation cost step along the two orthogonal directions only.
At the end of the pipeline simple post-processing operations are performed over the obtained data, as done for the other algorithm. Considering the overall method, it result to be give a denser point cloud compared to the previous method. However, its computational time is extremely higher. This is clearly due to the fact that this algorithm estimates each single pixel inside the specific subregions. Contrarily, in the *lightweight* strategy the number of samples that the user wants to estimate can be adjusted. In this way the trade-off between accuracy and resources utilization can be externally defined.

5.4 OpenCV built-in Semi-Global Block Matching algorithm

In comparison to the designed algorithms, it was decided to generate some results exploiting the Semi-Global Block Matching method available in the OpenCV library. This was mainly done to set a baseline outcome, extremely useful for comparing the disparity images obtained. As a matter of fact, the performance and the accuracy of the OpenCV built-in algorithm were essentially related to the outcomes carried out applying the simulated grid procedure.
Therefore, a clear measure in terms of both computational time and absolute accuracy has been available during project development, thus to have a transparent benchmark over the potential of the designed strategy.

5.5 Pre-processing testing and analysis

The pre-processing phase can be ideally divided in multiple parts. First of all, general operations have been applied for both estimation methods in order to remove noisy components from the stereo pair and from the ground truth images, when exploiting the simulated grid for the estimations. Then, a section of the pre-processing phase can be considered the importing of all the necessary calibration data and the creation of a sort of look-up table, which was used together with the grid of points.
On the contrary, when working with the real data coming from the LaDiMo stereo device, some initial operations have been necessary to clean up *bad* point values and for estimating some of the missing data, which would be determined without losing in accuracy.
Regarding the image pre-processing, different noise removal filters have been tested over database images, in order to understand which would provide the best trade-off in most of the cases. Median and bilateral filters appear to be, almost equiv-

alently, the best choice for smoothing away the noise while loosing the smallest amount of information from the input images, because of the peculiarity of being edge preserving filters.
Considering the others pre-processing operations implemented in the algorithm, they depend on the condition of working or not with the simulated grid.

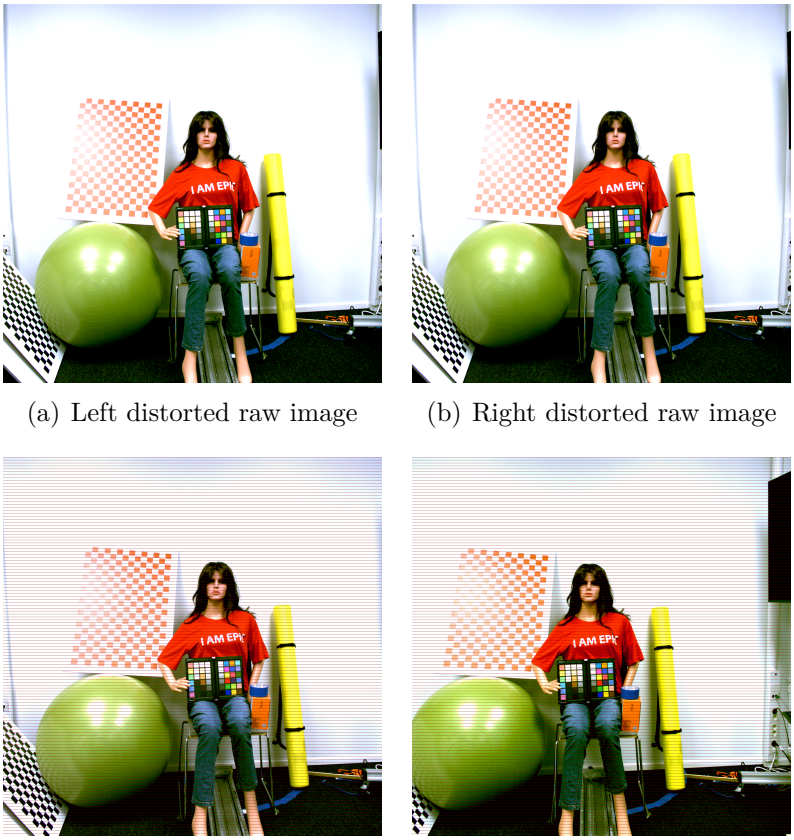
5.5.1 Look-up table and point grid pre-processing

Therefore, for the initial testing phase of the algorithm designing, when multiple tests have been carried out using the images from the Middlebury dataset, a look-up table has been defined in order to ensure a fast data retrieving. This structure has been easily built by progressively numbering the simulated gridpoints, so that their values correspond to the row index of the *complete data matrix*. That is the matrix that actually contains the space and relative pixel information for all the point belonging to the initial grid, i.e. X, Y and Z in meters or millimetres and x, y and *disparity* in pixels.
Regarding the implementation run over the real data, instead, some pre-processing operations have been necessary to adjust the initial device data, which were not always perfect. As a matter of fact, a low percentage of points of the laser grid, which ranges between 3.5 and 5.0 %, for the *micro* device and between 7.5 and 9.0 %, for the *4-eye* device, depending on the specific scene, was missing. It used to happen that the points were, in fact, not visible in the grid, as overlapped to the raw image, however, the raw data coming from the device present their grid coordinates alone, without any further information about their location. Therefore, those points were labelled and ordered by the algorithm running on the device, however no data were available regarding their space or image positions. This behaviour usually happened for the points located close to edges, and thus close or even above occlusions. In those cases the accuracy of the device drops locally and the provided results are, in fact, missing. So that, pre-processing procedures were necessary in order to reduce the percentage of initial errors, which usually diminishes between 0.8 and 1.2 %, for the *micro* device, and between 1.5 and 2.0 %, for the *4-eye* device, depending on the density of missing data in the occluded regions. However, even if in general the error dropping was not high, it marginally impacts on the overall performance of the algorithm, providing more accurate results in regions interested by edges and, hence, occlusions.
Specifically, two main operations have been performed over the initial raw data to reduce the errors. First of all, the grid points have been entirely analysed in order to identify the area where the percentage of missing data were low enough to allow a highly accurate reconstruction of the values of those point. As a matter of fact, a non perfectly exact estimation would carry errors over all the algorithm pipeline, which is a not desirable outcome.

Hence, where enough raw data were present in the neighborhood of occlusions, the estimations have been performed by evaluating a fast yet accurate linear interpolation exploiting the four closest neighbors to the specific point. Related to this, it was proved that performing a bilinear interpolation or using a weighted average to estimate missing values had the same grade of efficiency as the simple average, employing at least two orthogonal neighbours.
After the data filling a minor percentage of missing values was generally still present. Therefore, before running the algorithm those grid points was removed, being useless in the whole pipeline. In regard to this, it is worth to make notice that the *bad* points were in most of the cases almost the 2.5 - 6 % of the entire grid data. Therefore, it can be concluded that the value would not affect sensibly the overall accuracy of the final result and that a high density of point could be certainly reached with the available amount of raw data.
On the other hand, the application of a high expensive method, which would probably only provide an additional error reduction of the 0.5 % would be worth noting and it would most likely affect the speed of the algorithm, without providing a reasonable improvement of the density of the final point cloud. Therefore, it was considered beneficial to apply the aforementioned data recovering, over the condition of filling the missing data with a high accuracy. Considering, in fact, that the initial amount of errors was generally acceptable for achieving final accurate results, a mild data retrieving, which would not affect the performance of the algorithm while slightly increasing the amount of initial information, has been evaluated as valuable for the whole method.

5.5.2 Image undistortion and rectification

Concerning the tests carried out with the real environment images taken from the LaDiMo device, some pre-processing was necessary on the stereo pair too. In fact, the hardware employed gives as an output, on top of the point cloud data, a pair of distorted raw RGB images, which has to be first undistorted and then rectified, in order to be correctly used for the stereo matching.
Specifically these operations have been completed using the OpenCV built-in functions `stereoRectify()`, `initUndistortRectifyMap()` and `remap()`, which employ the camera calibration data, already available for the used camera, and give as output the matrices that map an image from the distorted reference systems to the stereo rectified space. In this section a concise explanation of the chain of operations that has been implemented with those functions is proposed. If further details regarding the specific parameters correlated to those functions are needed, the broad OpenCV documentation available is reminded.
Firstly the `stereoRectify()` function is fed with the original camera matrices and the distortion parameters of the two camera, which are available from the



(a) Left distorted raw image (b) Right distorted raw image

(c) Left rectified stereo image (d) Right rectified stereo image

Figure 5.4: Distorted raw and stereo rectified images from the Ladimo device

pre-computed calibration procedure. Moreover, as input to the function there is the transformation matrix from the coordinate system of the principal camera to the secondary camera. Actually, that is provided as the rotation matrix and the translation vector separately. Hence, the OpenCV function provides as its main outputs the two rectification transform matrices, that are the matrices that bring the 3D coordinates of a point from the unrectified coordinate system to the rectified coordinate system. Moreover, the camera projection matrices, which are fundamental in the stereo-matching process, are supplied.
After that, the function `initUndistortRectifyMap()` is exploited for both cameras in order to finalize the rectification procedure. As a matter of fact, it employs, on top of the initial camera matrices and the distortion coefficients, the rotation and projection matrices for each camera generated from the previous step. Therefore, with this operation the maps between the initial coordinate system, i.e. distorted, and the destination space, i.e. corrected and rectified, are computed, in

order to be, then, fed into the `remap()` function, which will perform the last step of the rectification process, providing the rectified stereo pair.
Figures from 5.4(a) to 5.4(d) display the input and the output of the whole rectification procedure. In the rectified stereo pair in Figures 5.4(c) and 5.4(d) the image scanlines, which in that coordinate system correspond to the epipolar line, are highlighted in order to provide a visualization of the correctness of the process. However, if that output is accurately analysed, it is possible to notice some mismatches between the two rectified images. Those are certainly due to errors in the calibration procedure, which needs to be adjusted for further experiments.

5.6 Post-processing testing and analysis

Two main typologies of filters have been applied to the 3D dense point cloud and the disparity image achieved as the output of the estimation algorithm.
Specifically, the disparity images obtained during the initial phases of the methods designing have been processed with bilateral or median filter. The latter result tends to perform better in terms of overall accuracy, even being slightly more expensive from a computational point of view. The fundamental feature of that filter is the edge preserving, which make it as the preferable choice for a non-invasive noise removal procedure.
Differently, the X, Y and Z data, estimated for the creation of the dense point cloud, have been handled at the end with a cascade of morphological operations, such as erosion, dilation, opening and closing. These have been merged together in order to find the best combination, which would be able to simultaneously adjust the result and preserve the achieved output. Therefore, these operations help in cleaning the whole data from estimations that were clearly wrong and enhance the smoothness and the homogeneity of the final cloud.

5.7 Preparation to future improvements

Considering the two designed methods for the estimation of a 3D dense point cloud through a stereo photogrammetry procedure, and comparing the results from both perfect dataset images and the ones gathered from the LaDiMo device some general conclusions about the work done and the further improvements that can be implemented in the algorithm can be drawn here, leaving a more interesting and complete discussion to the following chapters.
First of all, an important notice that should be done is that, compared with the OpenCV built-in algorithm `stereoSBGM()`, the lightweight developed algorithm is in general less precise. However, it is faster and the number of estimations that

can be performed inside the specific subregion can be easily adjusted in order to modify its accuracy, obviously loosing performance in computation.
Moreover, it has been proved that, if calibration mismatches are present in the stereo images, the OpenCV method tends to generate really noisy disparity images, especially in texture-less regions and near occlusions. Furthermore, the multiple parameters of that function must be wisely adjusted for every specific image to obtain a reasonable result.
On the contrary, the lightweight method developed is able to recover from small errors present in the initial raw images. This is due to the fact that this method does not completely rely on the stereo images but it exploits the data from the input cloud of sparse points. Therefore, thanks to this, small calibration errors do not affect drastically the final result of the algorithm, even if they are not desirable.
A final comment has to be introduced regarding the possible improvements that could be added to the algorithm. For sure, a higher accuracy of the final point cloud would be desired, meaning that smoothness and continuity should be enhanced among the edges, thus to provide a better looking result, making easier the understanding of the object in the specific scene. Therefore, a first idea that has already been thought to achieve this, keeping potentially real time performance, is to design a hierarchical procedure, where the most accurate points are iteratively used for further estimations, considering that two or three would be sufficient. Concerning this, if the computational time has to remain low, the estimations in the subsequent runs of the procedure can be focused on the regions close to edges and small objects.

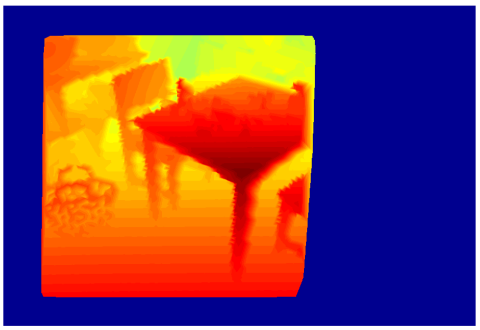
Chapter 6

Evaluation and Results comparison

In the previous chapters the description of the methods and of the key sections of the designed algorithms has been carried out. In Chapter 4 a general outline of the implemented idea has been presented. Instead, Chapter 5 illustrates the key functions and the computations that build up the entire algorithm.
Therefore, the functions exploited in the pre and post-processing phases have been previously outlined. Moreover, the two novel strategies designed have been reported, underlining the procedure implemented for obtaining the estimations of the grid points, the matching cost methods tested and their efficiency throughout the algorithm and the correlation with previously created stereo matching algorithms. Chapter 4 contains also a brief presentation of similar ideas developed in standard and in deep learning based methods, which were exploited to obtain an initial proof of the potential performance of the algorithm.
Thus, this chapter delineates the results obtained supporting the discussion with images of the point cloud and the disparity maps obtained. Moreover, some general comments related to the different result are going to be presented. In fact, a discussion focused on the considerations that can be carried out from the outcomes obtained will cover Chapter 7.
Here the interest is centred on the results achieved and in the comparison between the OpenCV method, defined as the baseline, and the algorithms designed. Moreover, in the first part of the following analysis the outcomes collected during the initial phase of the whole work are proposed. As previously introduced, their main function was to provide, at the beginning of the developed project, a proof of the feasibility of the thought strategy. That was, in fact, extremely useful for having, from the very beginning, a qualitative guidance on the project development.

(a) Left RGB test image

(b) Right RGB test image

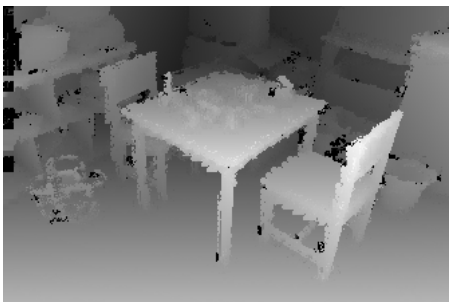


(c) Disparity test image

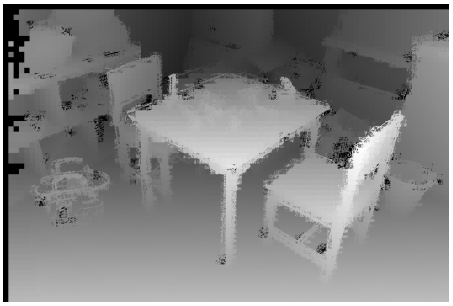
Figure 6.1: Test images from the Middlebury 2014 [32] dataset with disparity image estimated with linear simple Semi-Global Matching based method to test feasibility of thought strategy

6.1 Researching phase results

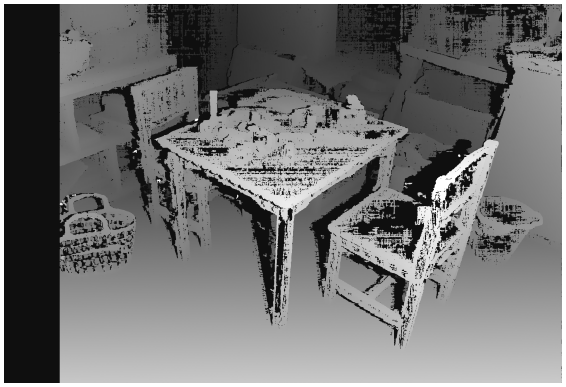
Figure 6.1 displays the result of the estimation of a disparity image exploiting a stereo pair from the Middlebury 2014 dataset. The disparity map shown in Figure 6.1(c) has been achieved applying a rather simple and linear implementation of a semi-global based method, where information from an initial simulated sparse grid of point is employed. As a matter of fact, in that initial stage of the project designing, the overall performance was not taken into account. The main focus was the feasibility of the procedure compared to standard SGM-based algorithms. In fact, the core of the algorithm computations stand in the matching cost phase, which is based on the Center-Symmetric Census transform and on the Hamming distance computation. Then, the aggregation cost step is carried out only over the four main orthogonal directions. Finally, a basic left-right consistency check procedure is applied, which generate the lost of information of the two image sides.



(a) Derivative-based method



(b) SGM-based method



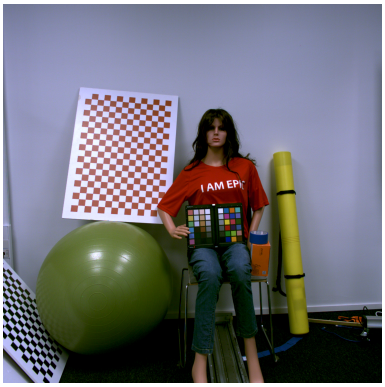
(c) OpenCV stereoSGBM method

Figure 6.2: Results of disparity images obtained with the disparity-based lightweight method, the SGM-based method and the baseline result from the OpenCV built-in method

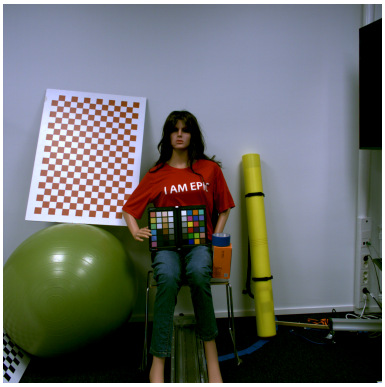
Clearly, this behaviour is related to the disparity range employed in the algorithm.

6.2 Derivative method results

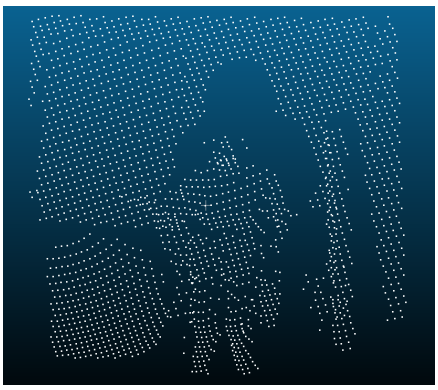
As Figure 6.1(c) proved, it is possible to achieve reasonable results with the designed method. However, the interesting thing is that, the initial tests performed show that, exploiting a sparse enough grid of points, which carry information on their 3D positions, the overall implementation can be made faster.
Therefore, after the initial researching phase, the algorithm started to be implemented in C++, in order to have a faster execution with respect to MATLAB, employing functions from the OpenCV library. In this regard, Figure 6.2 presents the grayscale disparity image achieved by applying the two designed methods, the derivative-based one and the SGM-based one, and then the outcome of the OpenCV built-in method is proposed as baseline for a visual comparison.



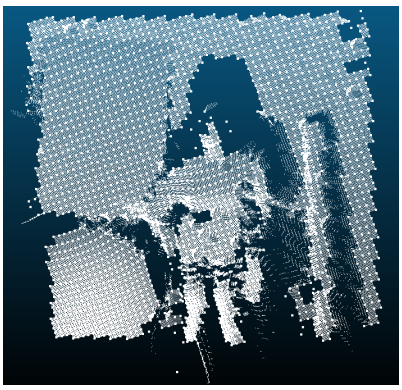
(a) Input left rectified image



(b) Input right rectified image



(c) Input point cloud



(d) Output dense point cloud

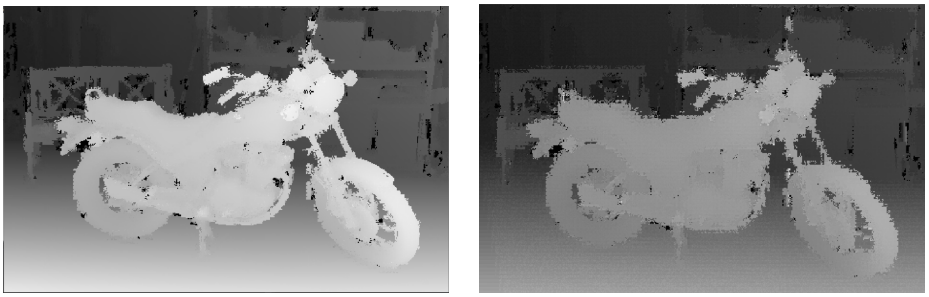
Figure 6.3: Pair of raw images from the LaDiMo device, initial grid of point and final point cloud estimated with derivative-based method

Therefore, considering the results of the disparity image of the *Playtable* dataset, some thoughts can be formulated. First of all, as expected, in the baseline result in Figure 6.2(c) the object are more detailed. This is, thus, highly visible where the object present small or thin features, as for the bag in the middle-left area of the image. Moreover, with the OpenCV `stereoSGBM()` algorithm there is an higher accuracy for the regions close to edges.
However, it has to be pointed out that, even if the OpenCV method has a lower amount of *local* noise, e.g. along the object edges, which are visibly clear, it is affected by higher amount of *general* noise, especially in occluded or texture-less areas. Regarding this detail, both of the results obtained with the designed algorithms present a lower accuracy along edges, highlighted by the *hairy* contours, however the surfaces are in general smoother. Let us consider the top of the table or of the chairs, or even the wall in the background, as an example. In the results

of Figures 6.2(a) and 6.2(b) the black spots are not generated. They are, most likely, due to the occlusions presents in the two stereo images, considering the table and the chairs, or they are generated in texture-less regions, as for the wall, for which it is known that the SGBM algorithm tends to fail.
More interesting results can be then visualized when taking into account the outcomes of the different algorithms obtained using images acquired by the company device. As a matter of fact, in this case, contrarily to the images from widely-used datasets, such as the Middlebury 2014, the KITTI 2012 and KITTI 2015, there could likely be calibration errors in the stereo pair. This leads to a mismatch between the epipolar lines of the two images, which should be overlapped to the image scanlines, when using a rectified stereo pair. Therefore, this problem causes a complete failure of the normal SGBM procedure.
On the other hand, the designed method, thanks to the information coming from the points grid, is able to recover from small mismatching errors. This condition proves the suitability of such a method in various situations, making it appropriate for those situations in which errors, even if minimal, cannot be avoided. It is, in fact, well known that in most of the real environment conditions it is generally difficult to have stable and controlled conditions, especially for lighting and for the physical stability of the hardware. The latter requirement is then extremely important for the accuracy of the final result. For this reason, in every industrial and commercial device employed in 3D estimation, as could be LiDAR or standard camera based hardware, a strong attention is provided to the system setup. However, errors related to unpredictable events or to little software bugs can affect the further computations, which would exploit the device outcome.

6.2.1 Consistency check over derivative computation

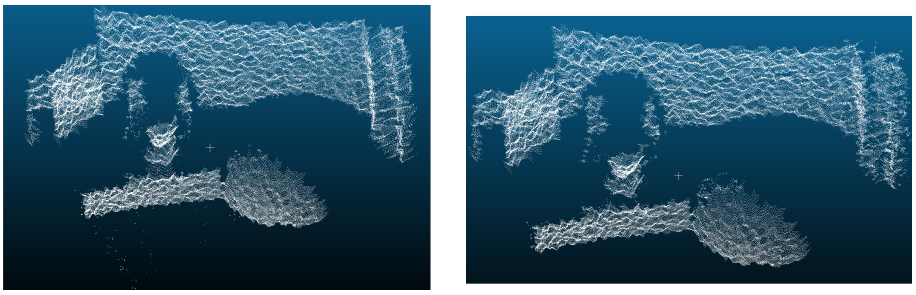
During the algorithm designing a specific phase of the pipeline demonstrates to be extremely important. Specifically, it consists on the *consistency check* performed during the derivative computation when running the algorithm exploiting the simulated grid of points. Actually, this consistency check becomes relevant in proximity of the edges, where the magnitude of the Z component of a derivative calculated between two points that belong to far objects, and thus cut by an occlusion, tends to explode. Therefore, this will affect the estimation procedure, which will most likely carry out wrong values for the computations performed in those areas.
Figure 6.4 clearly demonstrate the difference on the final result when the consistency check over the derivatives calculations is applied or not. When the magnitudes of the derivatives evaluated for a specific points are not correctly adjusted, if a set threshold on the Z components of the derivative vector is exceeded, the final disparity image will show a huge amount of noise, which is obviously related to in-



(a) Consistency check for derivative calculation

(b) No consistency check for derivative calculation

Figure 6.4: Comparison between applying and not a consistency check procedure over the same image



(a) Utilization of unique threshold over the whole input grid

(b) Utilization of multiple threshold for distinguish edge strength

Figure 6.5: Comparison between utilization of different thresholds distinguish edge strength

correct estimations. Figure 6.4(b) is the proof of that. Contrarily, in Figure 6.4(a) a better disparity map is visualized, which comes out when the consistency check operations are carried out.
Therefore, since the tests performed over the dataset images and using a simulated point grid, the necessity of the derivative correction was proven to be a valuable, and yet fundamental, step to reach an accurate result.

6.2.2 Different thresholds for edge intensity result comparison

Taking into account the results obtained employing the company device, some considerations can be performed in relation to useful improvements that have been implemented to the algorithm during its development.

A modification to the initial pipeline of the algorithm, which shows to give interesting improvement in the accuracy of the final 3D point cloud estimated, stands in the utilization of multiple threshold for the edge analysis. Specifically, as visible in Figure 6.5(a), utilizing a unique threshold for edge checking over the whole input cloud wrong estimations are generated. These are depicted in Figure 6.5(a) by the outlier points and are particularly visible where there is not a strong difference in depth among neighbouring point, as it happens between the face of the statue and calibration board on the foreground.
On the contrary, when different cases have been implemented in relation to different types of edges, strong and soft, those outliers have gone away, as Figure 6.5(b) depicts.

6.3 Stereo-matching based method results

Taking into account the results obtained from this method it has to be said that they does not reflect the optimal trade-off compared to the derivative-based algorithm. As a matter of fact, the computational time of this algorithm is actually relatively long. Despite that, the point cloud obtained in output is really dense, as expected, being the algorithm based on pixel-wise computations. However, if the results among all the analysed approaches are compared, this increase in density is not worth the rise in computational time, thing that would probably make this method intractable for real time applications. A real time execution of this approach could be, after all, reached with a proper GPU implementation and setting up a wise construction of the pipeline. Besides that, its best application could be most likely implied for use cases that need a high level of accuracy by sacrificing proper real time performance.
Therefore, the principal comparison could be done between the derivative-based approach, which can be described as a lightweight method, and the standard SGBM method built in OpenCV.

6.4 OpenCV method baseline results

Comparing the designed lightweight method and the built-in OpenCV SGM-based `stereoSGBM()` algorithm the following considerations can be provided.
First of all, with database images that are taken under optimal conditions and properly adjusted the OpenCV function results to be more accurate and it outputs a more correct disparity map. The downside is that, it is relatively computationally expensive, at least when executed on a normal laptop exploiting only the CPU.
On the other hand, the developed algorithm comes out to be sensibly faster, if

run on the same machine and in the same conditions. At the same time, it is less accurate, though. Regarding that, it has to be underlined that its computation is not optimized. Moreover, only basic pre and post-processing operations have been applied to its pipeline. Therefore, it could provide most likely better and more accurate results and probably even in a faster time.
Furthermore, an important achievement stands on the use of the initial grid of sparse points. As a matter of fact, it allows to recover from calibration errors and noise, which are almost unavoidable when images are taken in a real environment under normal conditions. Additionally, the input grid helps to deal with texture-less regions of the specific scene, which tend to make the normal SGM method fail. Therefore, at this stage of the work the result obtained with the derivative based method cannot be considered as optimal and the algorithm itself need some further work in order to be considered as competitive in the market, guaranteeing the level of accuracy necessary for providing safety in a real world application. Nevertheless, the implementation done until this point and the tests accomplished showed that improvements can be provided to reach a high level of accuracy in the estimation, while keeping the computational time in a range suitable for real-time applications.
Now, a final overall comparison among the results obtained with all the methods will follow in the next section. Some last comments about the achieved outcomes and their suitability are going to be outlined.
In conclusion, a final discussion over the overall work is going to be proposed in Chapter 7, where considerations about further improvements will be delineated.

6.5 Overall comparison of the outcomes

Summing up the outcomes obtained and the theoretical capabilities of all the different methods studied, the results of the derivative-based method can be considered as accurate result for the purposes of a researching project. Obviously, when compared over perfect dataset images with properly optimized methods, as the `stereoSGBM()` available in the OpenCV libraries, a certain lack in accuracy is visible.
However, some important features of the designed method, which highlight its potential suitability for real-time computer vision based applications, can be pointed out by analysing the outcomes achieved. Firstly, a key component of that method is the input grid of sparsified points, which allows to make accurate estimations over all the scene and recover from small calibration errors. Moreover, that feature supports the possibility of getting accurate results even in texture-less areas of the scene, which are extremely challenging for pure stereo matching based methods. As a matter of fact, in the literature studied, that was a problem that many researchers tried to overcome in multiple ways, such as designing deep learning based

algorithm in the latest years. In those publications, impressive results have been obtained by improving a standard SGM-based routine with convolutional neural networks. However, even if they outperforms almost all the previous algorithms, that performance is reached over standard dataset images, such as the Middlebury 2014, KITTI 2012, KITTI 2015 and over synthetic datasets. Those datasets obviously cover a massive number of real world situations, yet, it would be always possible to deal with environments for which the network has not been properly trained.
Therefore, another core feature of the designed algorithm stands exactly on that. It can be employed in every kind of scene and it does not need any type of pre-training. Thus, once it has been embedded in a stereo device, and adjusting the parameters relatively to the hardware specifications, it can provide a dense and accurate 3D point cloud of the scanned scene with a potential real-time performance.

Chapter 7

Discussion

The predetermined main objective of this project was to estimate a 3D dense point cloud starting from an initial sparse set of points and a pair of stereo undistorted images. Moreover, this overall algorithm should have been executed in real time, in order to be following embedded in a stereo device, thus to be suitable for autonomous driving, automotive and object detection related purposes. Therefore, tests for guaranteeing a potentiality for this type of usability have to be carried out.
Overall, this can be evaluated as a challenging work, also considering the different algorithms developed since the early nineties in this field, which is considered as one of the most researched areas in computer vision, especially nowadays.
Moreover, since the initial stages of the project, there was the need of studying and understanding the company device and, thus, find the optimal way to embed it in the algorithm pipeline. Additionally a novel strategy has to be thought, making it suitable for the requirements of the uses cases identified and competitive with the current methods. Furthermore, since the beginning there was the decision to focus on a standard approach, differently from most of the latest algorithms, which are based on deep learning techniques. This decision was made in order to do not encounter the problem of the domain shifting, that is the drop in accuracy that most of the CNN-based algorithms face when dealing with new type of environment, especially real ones, on which the network has not been properly trained [3]. Moreover, another factor that pushed for basing the algorithm on a more standard stereo matching strategy stands on the requirement of being able to embed the algorithm on a relatively small device. As a matter of fact, one of the company goals is to join the market of single-user devices, designing a product that should be extremely compact. Therefore, there would be quite infeasible to execute in the stereo system algorithms that requires a high computational cost, such as ones based on neural network structures. As a matter of fact, in this latter case, pre-training should be developed over an expensive machine, facing again the

problem of the environment shifting.
Thereby, in order to guarantee a generally accurate result in every possible situation, and for the task of embedding the algorithm on compact devices, the decision was to design a standard based and lightweight method, which would be able to ensure, at least theoretically, a real-time execution. Considering the whole work done, those tasks were overall achieved. Surely, further improvements are needed for guaranteeing an extremely fast execution, while keeping a high level of accuracy. Besides that, if the initial requirements and the different challenges faced during the project development are taken into account, the final result achieved can be considered as a reliable proof of the accomplishment of the initial objective defined.

7.1 Overall discussion over the project developed

Considering the overall project designed and especially comparing the outcomes achieved with the results of the OpenCV based Semi-Global Block Matching (SGBM) algorithm, which has been used as the baseline, the work done certainly showed that the strategy adopted is actually feasible for the predetermined goals. However, a certain lack in accuracy is visible. Moreover, the initial pre-processing section of the algorithm should be slightly improved, first of all the calibration of the stereo device, which is still not perfect.

7.2 Drawbacks of the designed algorithm

The main advantage of the algorithm developed is that it is sensibly fast. However, it also hold some side drawbacks, which do not affect its main execution, although they limit the possibility of getting perfect results.
As a matter of fact, the algorithm relies a lot on the initial set of points provided through the device, which is surely highly accurate, however it has a certain lack of performance in the detection of small objects. As specified since the initial chapters, the input points cloud is sparse, thus to ensure the maximum coverage of the scene. However, this cannot simultaneously guarantee high performance in terms of accuracy for all the objects in the scene. Therefore, a more related stereo matching strategy could be employed for those areas, thus to reduce the amount of errors and enhance the overall smoothness and continuity of the final dense cloud. Taking that into account, some approaches have been thought in order to improve the algorithm on that side. An example would stand on an initial segmentation of the scene. This will identify the different areas of the analysed environment, and for those regions on which the number of points is below a specific threshold, a more

pure SGM based strategy would be applied. In fact, the fundamental downside of the classical SGM method is that it is highly expensive, computationally speaking, especially when the disparity range and the image density increase. Therefore, if the number of disparity levels cannot be reduced drastically, the other feature that can be tackled is the dimension of the analysed area. Moreover, exploiting the depth information coming from the few points located in the area of interest, the disparity range can be reduced as well. Hence, if it is applied to small regions, i.e. the segmented areas of small objects, a final potentially real-time execution could be ensured.
Another imperfection present in the current version developed is related to the overall smoothness of the estimated 3D cloud. As a matter of fact, at this stage the estimated point cloud does not present smooth edges and some contours of the objects appears to be a bit rough. This problem can be effortlessly overcome by running multiple times (most likely two or three times should be enough) the algorithm and using the most accurate points from the previous stage, thus to enhance the density of the final cloud. Clearly this hierarchical method will slightly reduce the whole performance, considering that the multiple runs cannot be launched in parallel because the first estimation is needed for the following improvements. However, in the stages after the first one, not all the computations are produced again, therefore they can be considered as post-processing steps, from a computational point of view, meaning that they will not affect strongly the chance of produce an accurate results in real-time.

7.3 Further improvements and future work

Therefore, some further work will be needed in this project in order to make it reach a level of accuracy that can compete, in terms of accuracy and computational time, with most of the current algorithm designed for depth estimation. The developed work and the analysis done show a strong potential for the though technique. Moreover, they prove that the implemented strategy is reasonable for the predetermined scope and the employment of the initial cloud taken from the company device helps in keeping the computation fast and it ensures, at the same time, accurate results.
Summing up the development produced, the analysis carried out and the results achieved the following considerations can be outlined. First of all, the outcomes produced highlight that the strategy thought is reasonable and it allows to obtain generally accurate results in a reasonable time, which could be brought to a real-time performance by implementing the code in the GPU.
Secondly, considering the possible further improvements that could be implemented in the code, they mainly concern the smoothness of the final outcome

and the point cloud density, which could be highly enhanced by carrying out the algorithm pipeline in a hierarchical manner, exploiting at each stage the increasing number of certainly accurate points.
Moreover, the idea of including some segmentation on the input rectified images, in the pre-processing stage, in order to identify the regions that contain a low amount of initial point, would be analysed. The main scope of that would be improving the densification, i.e. the object reconstruction, in those areas where the laser points are limited, so that the estimation errors could easily decrease. Additionally, in the real environment images taken with the company device, areas with clusters of missing date have been observed close to edges, i.e. occluded regions. In the current implementation a initial pre-estimation of the 3D position of those points is realized by exploiting the data from the neighboring points.
However, in some region, because of the relatively high amount of missing data, an accurate generation of data was not possible, therefore, in those cases, it has been preferred to discard those points, thus to not affect the following estimations with wrong initial data. Anyway, the amount of those missing data do not influence strongly the whole procedure, considering that those points are only among 2.5 and the 5% with respect to the entire initial cloud. Therefore, that amount of initial can be accepted, being a reasonable percentage when dealing with real world data.

Chapter 8

Conclusions

In the previous chapter some considerations regarding the tasks initially assigned to this project and the final results achieved has been proposed. In relation to that, it is appropriate to summarize here the main acknowledgements and observations concerning the developed Master's thesis.

8.1 General and conclusive analysis

As outlined in Chapter 7, the predominant goal set at the initial stages of the project designing was the development of a novel method aimed at generate a dense 3D point cloud starting from a relatively small cluster of sparse points, thus to accurately describe the analysed environment.
This would be then be suitable for several applications, at different level of integration, in the computer vision area, such as object detection, robot manipulation, autonomous driving, monitoring car driver macro physical conditions[1].
This can be reasonably evaluated as not a simple objective to accomplish, also taking into account the multiple algorithms developed so far in this researching area. Therefore, different strategies to tackle the problem have been conceived since the very beginning of the project. Moreover, lots of tests over the algorithms produced have been carried out, in order to have clear and visible responses about the followed path.
The final results achieved can be, then, measured as a appropriate and suitable realization of the predetermined goals. Certainly, as mentioned in Chapter 7, further improvements will be added to the current work, thus to make it useful for real time applications, while guaranteeing a high level of accuracy of the outcome. However, especially regarding the *lightweight* algorithm designed, it clearly prove

[1] as an example an interesting application is tracking the car driver behaviour, thus to enhance the response of the safety system of the car for unexpected situations.

that, first of all, a high level of accuracy can be accomplished by exploiting its main idea, as displayed by the results shown in Chapter 6 Secondly, through proper improvements of the code, aimed at increasing its execution, real time applications can be considered as feasible.
Therefore, the accomplishment of the initial task identified should be regarded as produced, taking into account the main researching aspect of the work carried out. In relation to that, future improvements of the project have already been planned, thus to make it certainly competitive with the current state-of-the-art algorithm and suitable for the market requirements.

Bibliography

[1] A. Seki, M. Pollefeys, T. Corporation, E. T. Zürich, and Microsoft, "SGM-Nets: Semi-global matching with neural networks," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, no. 1, pp. 6640–6649, 2017.

[2] D. Scharstein, R. Szeliski, and R. Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Proceedings - IEEE Workshop on Stereo and Multi-Baseline Vision, SMBV 2001*, no. December, pp. 131–140, 2001.

[3] M. Poggi, D. Pallotti, F. Tosi, and S. Mattoccia, "Guided Stereo Matching," 2019.

[4] A. Tonioni, F. Tosi, M. Poggi, S. Mattoccia, and L. D. Stefano, "Real-Time Self-Adaptive Deep Stereo," pp. 195–204, 2020.

[5] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, 2008.

[6] J. Ko and Y. S. Ho, "Stereo matching using census transform of adaptive window sizes with gradient images," *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2016*, pp. 2–5, 2017.

[7] S. Birchfield and C. Tomasi, "Depth discontinuities by pixel-to-pixel stereo," *International Journal of Computer Vision*, vol. 35, no. 3, pp. 269–293, 1999.

[8] A. Klaus, M. Sormann, and K. Karner, "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure," *Proceedings - International Conference on Pattern Recognition*, vol. 3, pp. 15–18, 2006.

[9] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph cuts," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2, pp. 508–515, 2001.

[10] R. Szeliski, "Computer vision: algorithms and applications," *Choice Reviews Online*, vol. 48, no. 09, pp. 48–5140–48–5140, 2011.

[11] R. Hartley and A. Zisserman, "Multiple view geometry in computer vision. cambridge university press, isbn: 0521540518," 2004.

[12] Y. Yang, A. Yuille, and J. Lu, "Local, global, and multilevel stereo matching," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 274–279, IEEE, 1993.

[13] R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondence," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 801 LNCS, pp. 151–158, 1994.

[14] Z. F. Wang and Z. G. Zheng, "A region based stereo matching algorithm using cooperative optimization," *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2008.

[15] Q. Yang, L. Wang, R. Yang, H. Stewénius, and D. Nistér, "Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 492–504, 2008.

[16] M. Bleyer, M. Gelautz, C. Rother, and C. Rhemann, "A stereo approach that handles the matting problem via image warping," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 501–508, IEEE, 2009.

[17] D. Hernandez-Juarez, A. Chacon, A. Espinosa, D. Vazquez, J. C. Moure, and A. M. Lopez, "Embedded real-time stereo estimation via Semi-Global Matching on the GPU," *Procedia Computer Science*, vol. 80, pp. 143–153, 2016.

[18] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3061–3070, 2015.

[19] A. Kuzmin, D. Mikushin, and V. Lempitsky, "End-to-End learning of cost-volume aggregation for real-time dense stereo," *IEEE International Workshop on Machine Learning for Signal Processing, MLSP*, vol. 2017-Septe, pp. 1–6, 2017.

[20] E. S. Gastal and M. M. Oliveira, "Domain transform for edge-aware image and video processing," *ACM Transactions on Graphics*, vol. 30, no. 4, 2011.

[21] C. C. Pham and J. W. Jeon, "Domain transformation-based efficient cost aggregation for local stereo matching," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 7, pp. 1119–1130, 2013.

[22] J. Žbontar and Y. Lecun, "Stereo matching by training a convolutional neural network to compare image patches," *Journal of Machine Learning Research*, vol. 17, pp. 1–32, 2016.

[23] J. Žbontar and Y. Le Cun, "Computing the stereo matching cost with a convolutional neural network," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June, no. 1, pp. 1592–1599, 2015.

[24] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 4040–4048, 2016.

[25] R. Haeusler, R. Nair, D. Kondermann, C. Mostegel, M. Rumpler, F. Fraundorfer, H. Bischof, M. G. Park, K. J. Yoon, I. H. Md Yusof, M. An, M. H. Barghi, W. Luo, A. G. Schwing, Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang, "A Deep Visual Correspondence Embedding Model," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June, no. i, pp. 885–894, 2015.

[26] A. Spyropoulos, N. Komodakis, and P. Mordohai, "Learning to detect ground control points for improving the accuracy of stereo matching," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. i, pp. 1621–1628, 2014.

[27] A. Seki and M. Pollefeys, "Patch based confidence prediction for dense disparity map," *British Machine Vision Conference 2016, BMVC 2016*, vol. 2016-Septe, no. c, pp. 23.1–23.13, 2016.

[28] X. Hu and P. Mordohai, "A quantitative evaluation of confidence measures for stereo vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2121–2133, 2012.

[29] M. Poggi, F. Tosi, and S. Mattoccia, "Quantitative Evaluation of Confidence Measures in a Machine Learning World," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 5238–5247, 2017.

[30] M. Poggi and S. Mattoccia, "Learning from scratch a confidence measure.," in *BMVC*, 2016.

[31] M. G. Park and K. J. Yoon, "Leveraging stereo matching with learning-based confidence measures," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June, pp. 101–109, 2015.

[32] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8753, no. 1, pp. 31–42, 2014.

[33] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[34] W. T. Freeman and E. H. Adelson, "The Design and Use of Steerable Filters," 1991.

[35] R. Ohlander, K. Price, and D. R. Reddy, "Picture segmentation using a recursive region splitting method," *Computer Graphics and Image Processing*, vol. 8, no. 3, pp. 313–333, 1978.

[36] C. R. Brice, C. L. Fennema, and A. I. Journal, "by Paper to be submitted for publication in Technical Note 17 The research reported here was supported by the Advanced under Contract F30602-69-C-0056 , and is continuing under Administration and the Advanced Research Projects Agency .," no. April, 1970.

[37] R. M. Haralick and L. G. Shapiro, "Image segmentation techniques," *Computer vision, graphics, and image processing*, vol. 29, no. 1, pp. 100–132, 1985.

[38] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[39] D. Cremers, M. Rousson, and R. Deriche, "A review of statistical approaches to level set segmentation: integrating color, texture, motion and shape," *International journal of computer vision*, vol. 72, no. 2, pp. 195–215, 2007.

[40] A. Balke and M. Isard, "Active contours, the application of techniques from graphics, vision, control theory and statistics to visual tracking of shapes in motion," 1998.

[41] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International journal of computer vision*, vol. 1, no. 4, pp. 321–331, 1988.

[42] E. N. Mortensen and W. A. Barrett, "Intelligent scissors for image composition," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 191–198, 1995.

[43] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 6, pp. 583–598, 1991.

[44] G. Mori, X. Ren, A. A. Efros, and J. Malik, "Recovering human body configurations: Combining segmentation and recognition," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2, pp. II–II, IEEE, 2004.

[45] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[46] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, vol. 1, pp. I–I, IEEE, 2003.

[47] L. Keselman and A. Grunnet-jepsen, "Stereoscopic Depth Cameras," no. 1, pp. 1267–1276, 2017.

[48] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.

[49] H. Hirschmüller and D. Scharstein, "Evaluation of cost functions for stereo matching," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.

[50] S. Patil, J. S. Nadar, J. Gada, S. Motghare, and S. S. Nair, "Comparison of Various Stereo Vision Cost Aggregation Methods," *International Journal of Engineering and Innovative Technology*, vol. 2, no. 8, pp. 222–226, 2013.

[51] O. Demetz, D. Hafner, and J. Weickert, "The complete rank transform: A tool for accurate and morphologically invariant matching of structures," *BMVC 2013 - Electronic Proceedings of the British Machine Vision Conference 2013*, 2013.

[52] R. Spangenberg, T. Langner, and R. Rojas, "Weighted semi-global matching and center-symmetric census transform for robust driver assistance," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8048 LNCS, no. PART 2, pp. 34–41, 2013.

[53] S. Hermann and R. Klette, "Iterative semi-global matching for robust driver assistance systems," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7726 LNCS, no. PART 3, pp. 465–478, 2013.

Appendix A

Pseudocode of relevant algorithms

Some relevant algorithms, which can be useful for a complete understanding of the project, are proposed here in form of pseudocode. Specifically, it is worth to provide the indicative pipeline of the instructions implemented for the *lightweight* methods, which result to be the best performing in terms of accuracy and computational time.
Moreover, significant in relation to the developed work is the derivative calculation procedure used in the *simulated grid* method. Specifically, the **consistency check** function added to the evaluation of the local derivatives is following presented, hence it allows to achieve important improvement during the designing of the algorithm.

Regarding the pipeline presented in Algorithm 1, the other cases are omitted in order to not make the comprehension of the algorithm convoluted, being the procedure for the other conditions similar to the scheme outlined.
As a matter of fact, the steps applied are the same for all the grid points. Basically, each point of the initial grid is represented by a *lookup table*, whose value points to the corresponding row of the matrix, which stores the data related to the initial points.
Precisely, in the procedure some particular cases have to be mentioned. These are related to the points that belong to the borders of the grid, and thus of the table too. In those situations, there are no all the four neighbors for the considered point. Therefore, the actual derivatives are evaluated with the available data, the others are defined with the values equal to the opposite derivative but changed in sign. Let us consider, as an example, the point on the top-left corner of the grid. It has only two neighbors, the right and the bottom ones. Hence, the right and bottom local derivatives are exactly estimated computing the differences among the complete data. The left and top derivatives are, then, set equal to the ones previously evaluated, where the values are changed in sign.
Regarding the *consistency check*, it is worth to present the pseudocode related to

Algorithm 1 Local derivatives calculation (only one case highlighted)

```
1: for i = 0, 1, 2, ..., lookup_table.rows do
2:     for j = 0, 1, 2, ..., lookup_table.cols do
3:         Run the basic case, i.e. for the internal grid points
4:         if i > 0 ∧ j > 0 ∧ i < lookup_table.rows − 1 ∧ j < lookup_table.cols − 1
   then
5:             Get lookup_table value at position (i, j)
6:             It reminds to the row of the data matrix corresponding to the
   searched point
7:             Define point data
8:             Define top neighbor data
9:             Define bottom neighbor data
10:            Define left neighbor data
11:            Define right neighbor data
12:            Calculate the four local derivatives
13:            Apply consistencycheck between left and right derivatives
14:            Apply consistencycheck between top and bottom derivatives
15:        end if
16:    end for
17: end for
```

its implementation, shown in AlgorithmA. As a matter of fact, it represents an important step of the designed algorithm, allowing to reach interesting result when dealing with the simulated grid, as already explained in Section 6.2.1.

Moreover, pseudocode related to the general pipeline of the *lightweight* algorithm is worth to be reported, considering that this method resulted to be the best performing among the one tested.

Algorithm 2 Consistency check procedure between left and right derivatives to a points

if $abs(der_{left}.Z) > depth\_threshold$ **then**
    if $abs(der_{right}.Z) < depth\_threshold$ **then**
        $der_{left} = -der_{right}$
    else
        Estimate new correct value for the X_{mt} datum of the left derivative
        Estimate new correct value for the x_{px} datum of the left derivative
        Substitute the values to the initial ones
    end if
end if
if $abs(der_{right}.Z) > depth\_threshold$ **then**
    if $abs(der_{left}.Z) < depth\_threshold$ **then**
        $der_{right} = -der_{left}$
    else
        Estimate new correct value for the X_{mt} datum of the right derivative
        Estimate new correct value for the x_{px} datum of the right derivative
        Substitute the values to the initial ones
    end if
end if

Algorithm 3 Main pipeline of the *lightweight* algorithm

Import calibration data and images
Import initial raw point cloud data
Set the initial observed values obtained from the raw data
Define the LaDiMo grid
Define the grid squares
Calculate the internal and external derivatives
Image preprocessing
Set the estimations
procedure GRID POINT ESTIMATION
    Evaluate specific edge case
    Calculate matching cost
    Get best estimated values accordingly
end procedure
Post processing over estimations