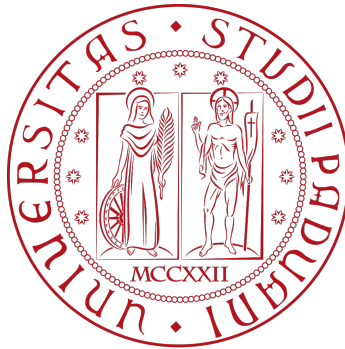


University of Padova
Department of Information Engineering

Ph.D. School of Information Engineering
Information Science and Technology
XXX Class



**Deep Learning for
Scene Understanding
with Color and Depth Data**

Ludovico Minto

Supervisor:
Pietro Zanuttigh, Ph.D.

Ph.D. School Director:
Prof. Andrea Neviani

January 15, 2018

Abstract

Significant advancements have been made in the recent years concerning both data acquisition and processing hardware, as well as optimization and machine learning techniques. On one hand, the introduction of depth sensors in the consumer market has made possible the acquisition of 3D data at a very low cost, allowing to overcome many of the limitations and ambiguities that typically affect computer vision applications based on color information. At the same time, computationally faster GPUs have allowed researchers to perform time-consuming experimentations even on big data. On the other hand, the development of effective machine learning algorithms, including deep learning techniques, has given a highly performing tool to exploit the enormous amount of data nowadays at hand.

Under the light of such encouraging premises, three classical computer vision problems have been selected and novel approaches for their solution have been proposed in this work that both leverage the output of a deep Convolutional Neural Network (ConvNet) as well jointly exploit color and depth data to achieve competing results. In particular, a novel semantic segmentation scheme for color and depth data is presented that uses the features extracted from a ConvNet together with geometric cues. A method for 3D shape classification is also proposed that uses a deep ConvNet fed with specific 3D data representations. Finally, a ConvNet for ToF and stereo confidence estimation has been employed underneath a ToF-stereo fusion algorithm thus avoiding to rely on complex yet inaccurate noise models for the confidence estimation task.

Contents

1	Introduction	4
1.1	Thesis Overview	5
2	Depth Cameras	8
2.1	Introduction	8
2.2	From 3D World to Image Plane	10
2.2.1	The Pin-Hole Camera Model	12
2.2.2	Heikkila’s Distortion Model	14
2.3	Time-of-Flight Cameras	15
2.4	Stereo Vision Systems	15
2.4.1	The Correspondence Problem	18
2.4.2	A Locally Global Disparity Refinement	20
3	ToF and Stereo Data Fusion	23
3.1	Introduction	23
3.2	Related Works	24
3.3	Proposed Method	26
3.4	Locally Consistent ToF-Stereo Data Fusion	28
3.5	SYNTH3: A ToF and Stereo Synthetic Dataset	28
3.6	ConvNet for Joint ToF and Stereo Confidence Estimation	30
3.7	Experimental Results	32
4	Segmentation and Semantic Labeling	38
4.1	Introduction	38
4.2	Related Works	39
4.3	Proposed Method	40
4.4	Over-segmentation	42
4.5	NURBS Fitting	43
4.6	A ConvNet for Semantic Labeling	44

<i>CONTENTS</i>	3
4.7 Region Merging	46
4.8 Experimental Results	49
4.8.1 Evaluation of the Segmentation Accuracy	51
4.8.2 Evaluation of the Classification Accuracy	53
5 3D Shape Recognition	58
5.1 Introduction	58
5.2 Related Works	59
5.3 Surface and Volume Representations	60
5.3.1 Multi-View Representation	60
5.3.2 Volume Representation	61
5.3.3 Surface Representation	62
5.4 3D Shape Recognition with ConvNet	63
5.5 Experimental Results	67
6 Conclusions	73

Chapter 1

Introduction

A series of rapid advancements concerning both data acquisition and processing hardware, as well as algorithmic solutions and software implementations, has recently opened the path to the development of novel and effective strategies to tackle several classical computer vision problems like image recognition, segmentation or semantic labeling. Furthermore, new problems have appeared in the meanwhile that were not even conceivable a few years ago, such as image caption generation [109], visual questioning answering [63, 47] photo-realistic image generation [82, 123] or amodal instance segmentation [58].

On one hand, the introduction of depth sensors in the consumer market has made possible the acquisition of 3D data at a very low cost, allowing to overcome many of the limitations and ambiguities that typically affect algorithms based on color information alone. At the same time, significant improvements in GPU technology have increased the computational power available at the hands of researchers by a considerable factor, enabling the test of time-consuming approaches even on large scales. On the other hand, the development of more powerful machine learning algorithms including recent deep learning techniques has allowed to take advantage of the enormous amount of data nowadays available.

Deep learning techniques and in particular deep Convolutional Neural Networks (ConvNets or CNNs) have represented a real breakthrough in computer vision, becoming a widely used tool for many research groups working in this field. They have basically allowed to move from pattern recognition approaches based on hand-crafted feature extraction to fully trainable processing pipelines capable to infer compact hierarchical representations even for objects belonging to complex high-dimensional spaces, as is the case of multi-channel images. Starting from Rosenblatt's perceptron [85], a binary classifier capable to model at most linear functions [68], significant

advances have been made first with shallow Multi-Layer Perceptron (MLP) networks and back-propagation [112], then with ConvNets [124, 56] allowing for multiple convolutional and pooling layers to be stacked in a deeper structure, until recently with Residual Networks [39] encompassing thousands of layers. A comprehensive overview on deep learning containing a detailed explanation of various state-of-the-art approaches can be found in [29].

This thesis aims at showing how exploiting deep learning techniques and depth data from consumer range cameras can positively impact on the solution of a number of classical yet challenging problems in the computer vision field, namely data fusion, joint segmentation and semantic labeling and object recognition. In particular, three novel methods are provided in Chapters 3, 4 and 5 to perform each of the aforementioned tasks respectively, together with experimental validation results and comparisons with state-of-the-art approaches.

On one hand, this research demonstrates how effective deep learning approaches can be when applied to even long-term scene understanding problems like semantic segmentation or recognition, for which several competing solutions have already been proposed in literature along the years. While it is clear how approaches using deep learning and in general machine learning can alleviate the burden of manually design ad-hoc low-level processing tools, it is not obvious whether they can actually achieve better or even comparable results on any given task.

On the other hand, it shows the importance of depth data as a valuable source of information complementary to color data, experimentally demonstrating how jointly exploiting both types of data from RGB-D images can lead to improved performances. In general, such a result cannot be taken for granted either, given the significant amount of noise and artifacts that usually affect depth data acquired with consumer range cameras as the ones employed in the experiments. Moreover, equipping a system with depth cameras usually results in an increased price and energy consumption (e.g. ToF cameras) or higher requirements in terms of computational power (e.g. stereo systems). Experimental results provide an estimate of the improvement one can get when depth data are coupled to color data, allowing for choosing a trade off between an improved accuracy and an increase in the overall system cost.

1.1 Thesis Overview

This research focuses on the application of deep learning techniques to the solution of several computer vision problems, as well as on the joint exploitation of color and depth data for improved performances.

Chapter 2 contains a brief overview on depth data acquisition and related concepts, providing some deeper insight on Time-of-Flight (ToF) cameras and stereo vision systems. In particular, the pin-hole projection model and Heikkila’s distortion model are first presented, giving the reader a theoretical representation for the depth acquisition process that is independent from the specific device implementation. Then, ToF cameras and their key working principles are presented, including some considerations about the most common sources of noise affecting data from such devices. Finally, binocular stereo vision systems are introduced, devoting some attention to the stereo correspondence problem (also known as the matching problem), which is at the core of any stereo disparity estimation algorithm. Many concepts and notations used in this chapter will be used in the subsequent chapters as well.

In Chapter 3 a novel framework is proposed for the fusion of depth data produced by a Time-of-Flight (ToF) camera and a stereo vision system. The key problem of balancing between the two sources of information is solved by extracting confidence maps for both sources using deep learning. A new synthetic dataset accurately representing the data acquired by the proposed setup is also introduced and used to train a ConvNet to regress the reliability of both data sources at each pixel location. The two depth fields are finally fused enforcing the local consistency of depth data taking into account the confidence information.

In the subsequent chapters two approaches are proposed for two scene understanding tasks, namely RGB-D image segmentation and semantic labeling and 3D shape recognition.

Chapter 4 presents a joint color and depth segmentation scheme exploiting together geometrical clues and a learning stage. The approach starts from an initial over-segmentation based on spectral clustering. The input data are also fed to a ConvNet thus producing a per-pixel descriptor vector for each scene sample. An iterative merging procedure is then used to recombine the segments into the regions corresponding to the various objects and surfaces. The proposed algorithm starts by considering all the adjacent segments and computing a similarity metric according to the ConvNet features. The couples of segments with higher similarity are considered for merging. Finally, the algorithm uses a NURBS surface fitting scheme on the segments in order to understand if the selected couples correspond to a single surface.

In the last chapter a novel approach is proposed for the classification of 3D shapes exploiting surface and volumetric clues inside a deep learning framework. Specifically, the algorithm uses three different data representations. The first is a set of depth maps obtained by rendering the 3D object. The second is a novel volumetric representation obtained by counting the number of filled voxels along

each direction. Finally NURBS surfaces are fitted over the 3D object and surface curvature parameters are selected as the third representation. All the three data representations are fed to a multi-branch ConvNet. Each branch processes a different data source and produces a classification vector by using convolutional layers of progressively reduced resolution. The extracted classification vectors are eventually fed to a linear classifier that combines the outputs in order to get the final results.

Chapter 2

Depth Cameras

2.1 Introduction

Depth cameras and range sensing systems in general [119] allow to easily extract valuable 3D information describing the geometry of the scene by providing range measurements relative to a set of observed surface points in the scene. The use of depth cameras has recently experienced a great boost following their introduction into the mass market segment resulting in a substantial decrease of their price and a consequent increase in the number of possible applications. These two factors, namely the ability of collecting considerable amounts of depth data using inexpensive yet accurate devices and the intrinsic appeal generated by their much wider application spectrum, have encouraged more and more academic and industry researchers to explore novel approaches explicitly tailored to advantage of the 3D information content carried by such data in order to tackle long-term challenging problems in the computer vision field, as well as new ones.

Several depth cameras typologies are currently available, corresponding to as many different range sensing methods and working principles. Despite this variety, depth cameras can be conceived as functionally equivalent in that they are all capable to provide a depth map of the scene as output, i.e. a dense or sparse set of values spatially arranged as a lattice, each value roughly accounting for the distance between some observed surface point in the scene and the camera.

A more precise explanation can be given about the content of a depth map by defining a 3D coordinate system called the Camera Coordinate System (CCS) associated to the camera frame ¹ with axes x , y and z oriented as depicted in Figure

¹Whenever a multiple camera system such as a stereo vision system is considered, a 3D coordinate system associated to the reference camera frame is defined.

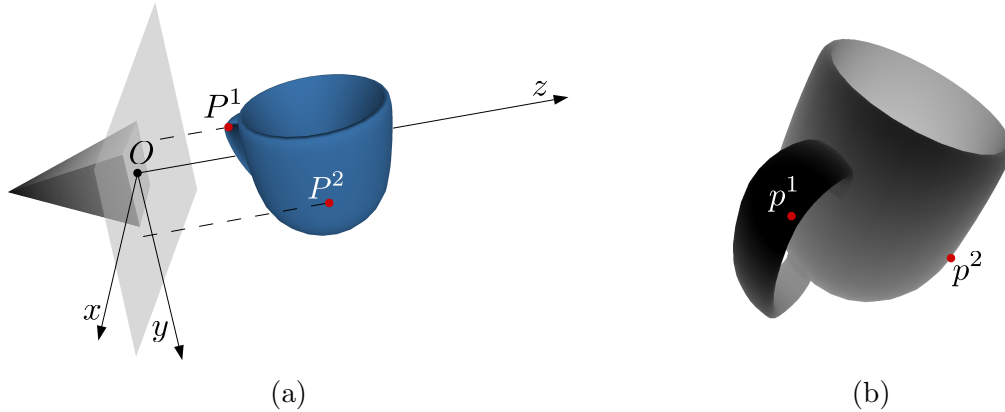


Figure 2.1: An intuitive example showing the content of a depth map for a generic acquisition scenario. Range data acquired by a camera (a) are mapped on a depth map (b). The value of a pixel in the depth map encodes for the z -coordinate of its corresponding point in the scene given with respect to the Camera Coordinate System (CCS). Darker regions correspond to lower depth values (e.g., pixel p^1 corresponding to the point P^1 with CCS coordinates $\mathbf{P}^1 = (x^1, y^1, z^1)^T$), lighter regions to higher ones (e.g., pixel p^2 corresponding to point P^2 with CCS coordinates $\mathbf{P}^2 = (x^2, y^2, z^2)^T$, note that $z^1 < z^2$).

2.1 i.e. with origin O located at the camera aperture and the z -axis aligned with the camera optical axis. Notice that, in doing so, a series of assumptions have been subsumed regarding the geometry of the camera, partially anticipating the pin-hole camera model that will be introduced in the following section. Modeling each pixel or cell in the camera sensor array as a point in a 2D normalized lattice, a dense depth map extracted from the camera is a function mapping each 2D point in the lattice to the z coordinate of a corresponding 3D point in the scene given with respect to the camera coordinate system CCS. The way 3D points in the scene can be associated to sensor pixels will be the subject of the following section. In a sparse depth map not all pixels are assigned a value, or equivalently a are assigned a conventional invalid value such as a negative number.

Many algorithms exploiting depth data are completely unaware of the working principles beneath the range sensing process implemented by the particular depth camera used to extract the data. As well, their design does not require any detailed knowledge about the way depth data are acquired. From this perspective, depth cameras can be treated as black boxes from which depth maps can be pulled at a certain frame rate and then processed.

On the contrary, other algorithms exist that need to explicitly account for the particular depth data acquisition process in order to be properly designed. An example is the Stereo-ToF data fusion algorithm described in Chapter 3 for which limitations and error sources peculiar to stereo vision systems and ToF cameras respectively have been considered. Also, aside from depth maps, other data might be exposed to the user depending on the particular device implementation. As an example, many Time-of-Flight cameras allow the user to retrieve from the camera buffer both amplitude and intensity data as well.

For this reason it is worth to briefly introduce here some of the differences between the various range sensing methods that are commonly implemented by most consumer depth cameras.

To start with, all depth cameras considered in this and following chapters implement some kind of reflective optical distance measurement method. This large family can be further divided into two main groups encompassing methods relying on passive and active range measurements respectively.

The first group includes methods that exploit passive sensing techniques in order to estimate the range of the various scene points. On-board sensors are used to capture the radiation already present in the scene, e.g. visible light from a natural or artificial light source, and reflected by the various scene surfaces towards the camera. Stereo vision systems such as the StereoLabs Zed camera [122] (see Figure 2.2(d)) belong to this first group.

Methods belonging to the second group work by first projecting some form of electro-magnetic radiation in the scene. The transmitted radiation is then back-reflected by the scene surfaces and captured via suitable sensors similarly to passive range measurement methods. Commonly used radiations are infrared (IR) or near-infrared (NIR). Structured light and matricial Time-of-Flight (ToF) cameras are both part of this group, yet they rely on completely different working principles. Creative Sens3D (see Figure 2.2(c)) and the first version Microsoft Kinect (see Figure 2.2(b)) are two popular ToF cameras, while the first version Microsoft Kinect is a prominent example of structured-light depth camera (see Figure 2.2(a)). A deeper insight on the working principles of ToF cameras and stereo vision systems is given in Section 2.3 and Section 2.4 respectively.

2.2 From 3D World to Image Plane

This section aims in providing a mathematical formulation modeling the way electro-magnetic radiation from surface points in the observed scene is projected onto the camera sensor. Although strong assumptions are made, the proposed model and de-



Figure 2.2: Examples of consumer depth cameras: Microsoft Kinect v1 (a), Microsoft Kinect v2 (b), Creative Sens3D (c), StereoLabs ZED (d).

rived projection function are expected to accurately fit most of the reflective projection cameras now available, including standard cameras used in stereo vision systems or ToF cameras, both described in more detail in the next sections.

To express this goal in more precise terms, let a 3D coordinate system called the World Coordinate System (WCS) be associated to the scene frame and denote with $\mathbf{P}_W = (x_W, y_W, z_W)^T$ the coordinates of a scene point P given with respect to WCS. Moreover, consider S the camera sensor plane (also known as the image plane), define on it a 2D coordinate system called the Sensor Coordinate System (SCS) and denote with $\mathbf{p} = (u, v)^T$ the coordinates of a sensor point p given with respect to SCS. The idea is to provide an accurate projection function π taking in input the coordinates \mathbf{P}_W of a 3D point in the scene and returning as output the coordinates \mathbf{p} of a corresponding 2D point on the sensor plane, i.e.

$$\pi(\mathbf{P}_W) = \mathbf{p} \quad (2.1)$$

In order to simplify the whole analysis, an ideal pin-hole camera model is introduced to describe the geometry of the camera. Starting from this premise, a straightforward projection function can be easily derived as detailed in Section 2.2.1

using simple geometry tools. The Heikkila’s model is then used in Section 2.2.2 to account for the distortion introduced by the several optical elements that the incoming radiation usually encounters along its path to the camera sensor.

Both the pin-hole model and the Heikkila’s distortion model are defined through a fixed set of parameters whose values must be properly chosen in order to achieve a good accuracy. A set of factory default values are usually available to this purpose. Alternatively, an optimal set of parameter values can be explicitly derived following a process called camera calibration. An extensive overview on several calibration procedures together with a detailed explanation can be found in [119].

2.2.1 The Pin-Hole Camera Model

The pin-hole camera model assumes the camera aperture is an ideal point, small enough to let one light ray only per observed point in the scene to pass through it. The model uses a few intrinsic parameters to describe the geometry of the camera, while a set of extrinsic parameters encode for the camera pose with respect to the world coordinate system WCS.

Leveraging the simplicity of the pin-hole model, a straightforward yet accurate approximation for the projection function (2.3) can be derived, parametrized with respect both to the intrinsic and extrinsic parameters

In order to further simplify the derivation process leading to the desired projection function, a 3D coordinate system called the Camera Coordinate System (CCS) is associated to the camera frame, denoting with $\mathbf{P} = (x, y, z)^T$ the coordinates of a scene point P given with respect to CCS. The derivation of a projection function in the form of Equation (2.3) can be split into two separate steps. In the first step, points with given CCS coordinates are projected to points on the sensor plane expressed in terms of SCS coordinates. In the second step, such projection is modified in order to directly project scene points with given WCS coordinates rather than CCS ones.

To start with, let the coordinate system origin or center of projection O be located at the camera aperture and assume that the x , y and z are oriented as depicted in Figure 2.3(a). Also, assume the sensor plane S is parallel to the xy -plane of the CCS coordinate system, intersecting the z -axis at the negative z -coordinate f , where $|f|$ is the camera focal length. Finally, let the z -axis intersect S at the point c with coordinates $\mathbf{c} = (c_x, c_y)^T$ given with respect to SCS. Within this framework, the following relationship holds between SCS and CCS coordinates:

$$\begin{aligned} u &= x + c_x \\ v &= y + c_y \end{aligned} \tag{2.2}$$

Using triangle similarities, it is easy to write down the equations projecting a scene point P with coordinates $\mathbf{P} = (x, y, z)^T$ to a corresponding sensor point p with coordinates \mathbf{p} , namely

$$\begin{cases} u &= f \frac{x}{z} + c_x \\ v &= f \frac{y}{z} + c_y \end{cases} \quad (2.3)$$

The same projection formulated by Equation (2.3) can be put into a convenient matrix form by using homogeneous coordinates, leading to

$$z \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (2.4)$$

Typically, two scale factors k_u and k_v are introduced to deal with the actual horizontal and vertical size respectively of sensor cells in the camera sensor matrix, obtaining

$$z \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathbf{K} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (2.5)$$

where \mathbf{K} is the intrinsic parameter matrix associated to the camera, defined as

$$\mathbf{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \quad (2.6)$$

with $f_x = fk_u$ and $f_y = fk_v$ x -axis and y -axis focal lengths respectively.

Finally, let \mathbf{R} and \mathbf{t} be the rotation matrix and translation vector of the rigid transformation mapping WCS to CCS, a concise mathematical formulation for the function mapping a scene point P with coordinates \mathbf{P}_W to its projection p onto the camera sensor plane can be written using homogeneous coordinates as

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \frac{1}{z} \mathbf{K}(\mathbf{R}|\mathbf{t}) \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix} = \frac{1}{z} \mathbf{KM} \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix} \quad (2.7)$$

where $\mathbf{M} = \mathbf{K}(\mathbf{R}|\mathbf{t})$, called the projection matrix, is built from both the intrinsic parameter matrix \mathbf{K} and the extrinsic parameters \mathbf{R} and \mathbf{t} .

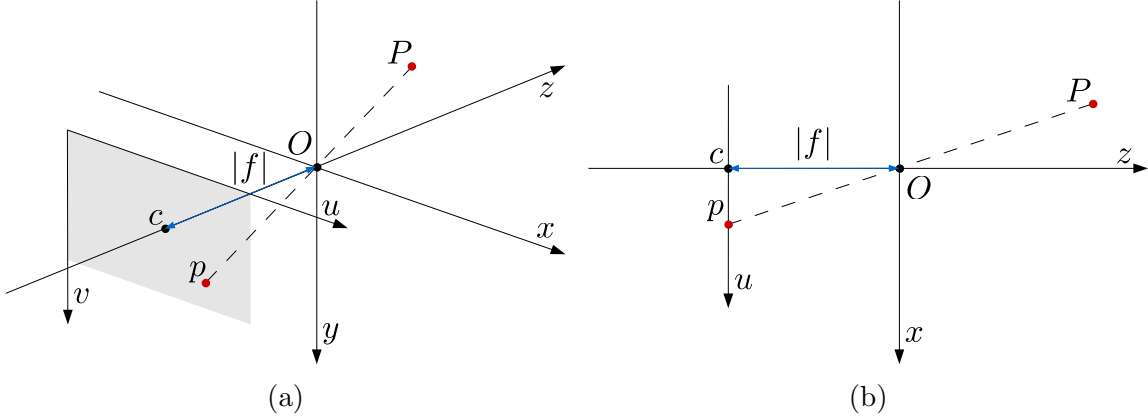


Figure 2.3: A simple sketch representation for an ideal pin-hole camera model: perspective view (a), side view (b).

2.2.2 Heikkila's Distortion Model

The presence of lenses and other optical elements introduces distortions that are not captured by the pin-hole camera model of Section 2.2.1, making the projection defined by Equation () inaccurate. In general, such distortions can be modeled by a function ψ so that a light ray coming from a point P in the scene and entering the pin-hole camera aperture is projected onto the image plane at coordinates $\psi(\mathbf{p})$ instead than coordinates \mathbf{p} as computed by Equation (). As a result, the value measured by pixel p with coordinates \mathbf{p} can no longer be associated to scene points P with coordinates \mathbf{P}_W such that $\pi(\mathbf{P}_W) = \mathbf{p}$.

To cope with this mis-matching issue an anti-distortion transformation ψ^{-1} can be applied to adjust pixel coordinates \mathbf{p} to new coordinates $\psi^{-1}(\mathbf{p})$. Alternatively, one can update the projection function π as defined in Equation () so that it includes the distortion transformation ψ .

In many scenarios, the model proposed by Heikkila in [41] is enough to accurately describe the distortion introduced by camera optics. Following the model, the anti-distortion function is defined as

$$\psi^{-1}(\hat{\mathbf{p}}) = \mathbf{p} = \begin{pmatrix} \hat{u}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2d_1 \hat{u} \hat{v} + d_2 (r^2 + 2\hat{u}^2) \\ \hat{v}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + d_1 (r^2 + 2\hat{v}^2) + 2d_2 \hat{u} \hat{v} \end{pmatrix} \quad (2.8)$$

where $r = \sqrt{(\hat{u} - c_x)^2 + (\hat{v} - c_y)^2}$, $\hat{\mathbf{p}}$ are distorted image plane coordinates and \mathbf{p} are the corresponding undistorted ones. As can be noticed, just five constant parameters are required to completely determine the model, namely k_i with $i =$

1, 2, 3 accounting for radial distortion and d_i with $i = 1, 2$ accounting for tangential distortion. The model can be easily included in the projection function of Equation () and the distortion parameter vector $\mathbf{d} = (k_1, k_2, k_3, d_1, d_2)^T$ added to the set of intrinsic camera parameters.

2.3 Time-of-Flight Cameras

ToF devices operate following the RADAR (Radio Detection and Ranging) principle thus relying on the fact that the electro-magnetic radiation travels in air at light speed $c \approx 3 \times 10^8$ [m/s]. Hence the distance ρ [m] covered in time τ [s] by an optical radiation is $\rho = c\tau$. The radiation $s_E(t)$ emitted at time 0 by the ToF transmitter TX travels straight towards the scene for a distance ρ . It is then echoed or back-reflected by a scene surface point P , covering again a distance ρ to travel back. At time τ it reaches the ToF receiver (or sensor) RX, ideally co-positioned with the transmitter, as signal $s_R(t)$ (see Figure 2.4). Assuming this is the only path covered by the signal (no other reflections in-between), the round-trip distance is equal to 2ρ and the following relationship can be established between ρ , the distance of P from the sensor, and τ , the time spent for the signal to come back, i.e.

$$\rho = \frac{c\tau}{2} \quad (2.9)$$

Equation 2.9 is the basic expression of a ToF camera's distance measurement.

The accurate measurement of the round-trip time τ is the fundamental challenge in ToF systems. Methods that solve this task can be divided into two broad families. On one side direct approaches, that address the measurement using pulsed light and trying to directly estimate the time τ , or alternatively the phase shift ϕ using a continuous wave. On the other side, indirect approaches try to derive τ or ϕ from time-gated measurements of signal $s_R(t)$ at the receiver. More details can be found in [119].

2.4 Stereo Vision Systems

Stereo vision systems or simply stereo systems are able to provide range estimates using a set of multiple standard cameras with partially overlapping fields of view. Binocular stereo systems represent the most common configuration for such typology of devices. They are made by two cameras, one of them referred to as the reference camera, the other one as the target camera. In this section, some basic concepts

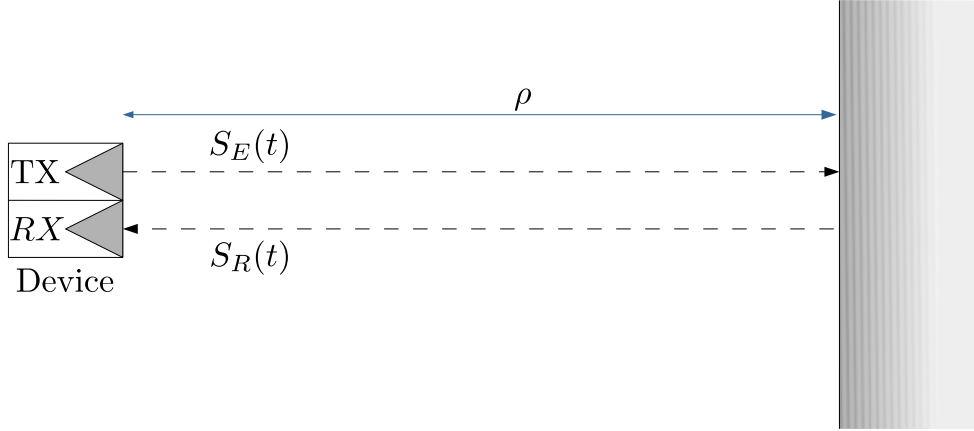


Figure 2.4: ToF measurement basic principle. An electro-magnetic signal is emitted by the transmitter TX , back-reflected by a scene surface at a distance ρ from the device and finally captured by the receiver RX after a time interval τ .

about stereo vision are presented by examining the case of a simplified binocular stereo system while referring to [119] for a more detailed explanation including N-views stereo systems.

A pictorial representation of a binocular stereo system can be seen in Figure 2.5 where the left camera, denoted with R , has been chosen as the reference camera while the right camera, denoted with L is the target camera.

Both cameras can be modeled with a pin-hole camera model like the one described in Section 2.2.1, each with its own camera and sensor coordinate systems. In particular, let R-CCS and T-CCS be the camera coordinate systems associated to the reference and target cameras respectively, $\mathbf{P}_R = (x_R, y_R, z_R)^T$ the coordinates of a scene point P with respect to R-CCS and $\mathbf{P}_T = (x_T, y_T, z_T)^T$ the coordinates of the same point given with respect to T-CCS. Moreover, let R-SCS and T-SCS be the 2D coordinate systems defined on the reference and target camera sensor planes such that the coordinates of a point p_R on the reference camera sensor plane are denoted with $\mathbf{p}_R = (u_R, v_R)^T$ while the coordinates of a point p_T on the target camera sensor plane are denoted with $\mathbf{p}_T = (u_T, v_T)^T$.

Both reference and target cameras are assumed to be calibrated with intrinsic parameter matrices \mathbf{K}_R and \mathbf{K}_T respectively.

Looking at Figure 2.5, any point P in the scene can be projected onto the reference and target camera sensors following the projection rules their associated pin-hole models, resulting in a pair of conjugate or corresponding points p_R and p_T . As will be clearer in a moment, the search for such conjugate pairs is at the core of any

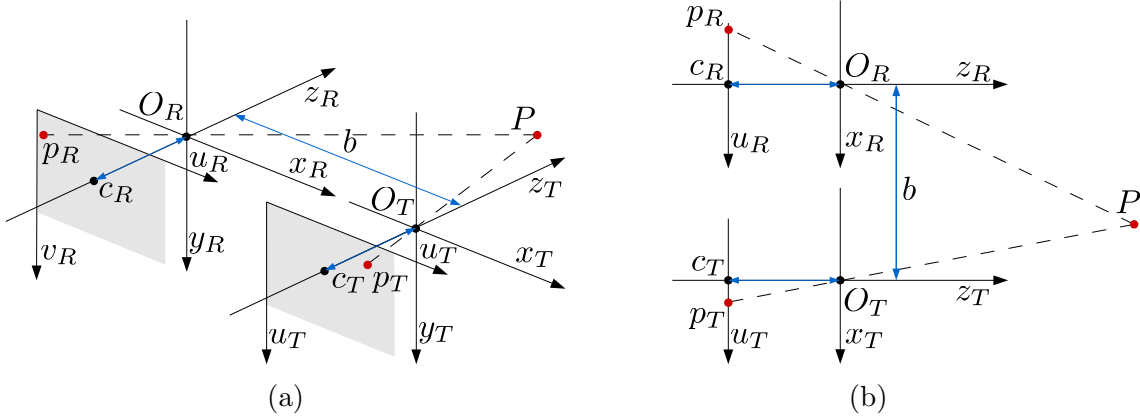


Figure 2.5: A simple sketch representation for a binocular stereo vision system where each camera is described using a pin-hole camera model: perspective view (a), side view (b).

stereo vision algorithm, allowing to estimate the depth of P using triangulation.

In what follows, a relationship between the depth of P and the coordinates of its associated conjugate pair p_R and p_T is derived, providing in particular a simple equation to compute z_R from \mathbf{p}_R , \mathbf{p}_T and the relative pose of the reference camera with respect to the target one.

To let the derivation be as simpler as possible, some assumptions are made. Specifically, the two cameras are assumed to be substantially identical, sharing the same intrinsic parameter matrix $\mathbf{K}_R = \mathbf{K}_T = \mathbf{K}$ where K is

$$\mathbf{K} = \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix} \quad (2.10)$$

Moreover, the two cameras are assumed to have parallel optical axes, their sensors to lie on the same plane and be aligned so that p_R and p_T have identical vertical coordinates $v_R = v_T$. Whenever this is not the case, a rectification procedure must be applied beforehand.

Within these settings, the disparity of p_R , denoted with d , is defined as the difference between its horizontal coordinate and the horizontal coordinate of its conjugate point p_T , i.e.

$$d = u_R - u_T \quad (2.11)$$

With this notation, the coordinates of p_R and p_T with respect to R-SCS and T-SCS respectively can be rewritten as

$$\mathbf{p}_R = \begin{pmatrix} u_R \\ v_R \end{pmatrix} \quad \mathbf{p}_T = \begin{pmatrix} u_T = u_R - d \\ v_T = v_R \end{pmatrix} \quad (2.12)$$

Using extended vectors $\tilde{\mathbf{p}}_R = (u_R, v_R, 1)^T$ and $\tilde{\mathbf{p}}_T = (u_T, v_T, 1)^T$ and inverting (2.5) allows one to write the coordinates of P with respect to R-CCS and T-CCS as

$$\begin{aligned} \mathbf{P}_R &= z_R \mathbf{K}^{-1} \tilde{\mathbf{p}}_R \\ \mathbf{P}_T &= z_T \mathbf{K}^{-1} \tilde{\mathbf{p}}_T \end{aligned} \quad (2.13)$$

Finally observe that the way the two cameras are aligned allows to express the rigid transformation (\mathbf{R}, \mathbf{t}) mapping T-CCS to R-CCS in a much simpler form than the one holding for general rotation matrix \mathbf{R} and translation vector \mathbf{t} . Indeed, under the given assumptions, \mathbf{R} is equal to the identity matrix while $\mathbf{t} = (-b, 0, 0)^T$, resulting in the following straightforward relation between \mathbf{P}_R and \mathbf{P}_T

$$\mathbf{P}_R = \mathbf{P}_T + \begin{pmatrix} -b \\ 0 \\ 0 \end{pmatrix} \quad (2.14)$$

Combining (2.13) and (2.14), the depth z_R of P given with respect to R-CCS can be expressed as

$$z_R = \frac{bf}{d} \quad (2.15)$$

For a rectified stereo system it is straightforward to derive d from (2.11) once the the conjugate pair is given, while b is a known fixed parameter characterizing the system. From 2.15 it is clear that, in order to estimate a dense depth map for the observed scene, it is crucial to be able to associate the correct conjugate point (pixel) to every point (pixel) in the reference camera sensor plane (image). The problem of extracting conjugate points from a pair of stereo images is known as the stereo pair correspondence or matching problem, which is the subject of the following section.

2.4.1 The Correspondence Problem

The computation of a depth map from a stereo image pair via triangulation assumes conjugate points \mathbf{p}_R and \mathbf{p}_T are already available. For this reason, the stereo correspondence problem (sometimes called the stereo matching problem), i.e. the problem of finding such conjugate pairs between the reference and target images, represents a

key step for any stereo vision algorithm. Still nowadays no ultimate solution has been proposed to this problem, which keeps being considered a challenging one mainly for two reasons. On one side, due to occlusions, some of these pairs may not exist. Usually, not every pixel in the reference image has a conjugate pixel in the target image. On the other side, there might be cases for which finding such a pair might result extremely difficult or even impossible. Scene appearance plays an important role in determining the feasibility of such task. Consider the extreme case where every pixel in both reference and target images encode for the same color, which can happens e.g. when the stereo system is placed at a close distance from a white uniform wall. Clearly, no meaningful correspondence between points in the two images could be set in this case, being every pair relating any point in the reference image to any point in the target image potentially feasible.

A simple taxonomy of approaches to the correspondence problem divide them into sparse and dense as well as local and global ones.

Dense stereo vision approaches aim finding the conjugate pixel for every pixel in the reference image, provided it exist. On the contrary, sparse approaches usually limit this search to a much smaller subset of all existing pairs, whether or not it covers all pixels in the reference image.

A more interesting distinction is the one between local and global methods. Local methods look for a conjugate point by maximizing local similarities. Specifically, for a given point p_R in the reference image and candidate point p_T in the target image a cost function is evaluated that compares a support window $N(p_R)$ centered at p_R with a corresponding support window $N(p_T)$ centered at p_T , penalizing dissimilarities between the two windows.

Assuming undistorted and rectified stereo image pairs, only pixels in the target image that are located at the same row of p_R are picked as candidates. Furthermore, candidate pixels corresponding to disparity values higher or lower than a threshold are typically excluded from the search. In this case, the search for a conjugate point can be cast to the search for a suitable disparity in a range of candidate disparities $\{d_{min}, \dots, d_{max}\}$. To speed the process a winner-takes-all strategy is typically enforces, selecting a conjugate point among candidates minimizing the cost function.

Cost aggregation methods [87] add an additional processing step by summing or averaging the costs computed at neighboring points inside the support window. The rationale behind this strategy is the underlying assumption that the disparity map is piecewise smooth except in correspondence of depth discontinuities.

To cope with discontinuities between smooth regions some approaches have been proposed that properly adapt the size or shape of the support window [105] or aggregate the costs by a weighted average where weights are adjusted based on the

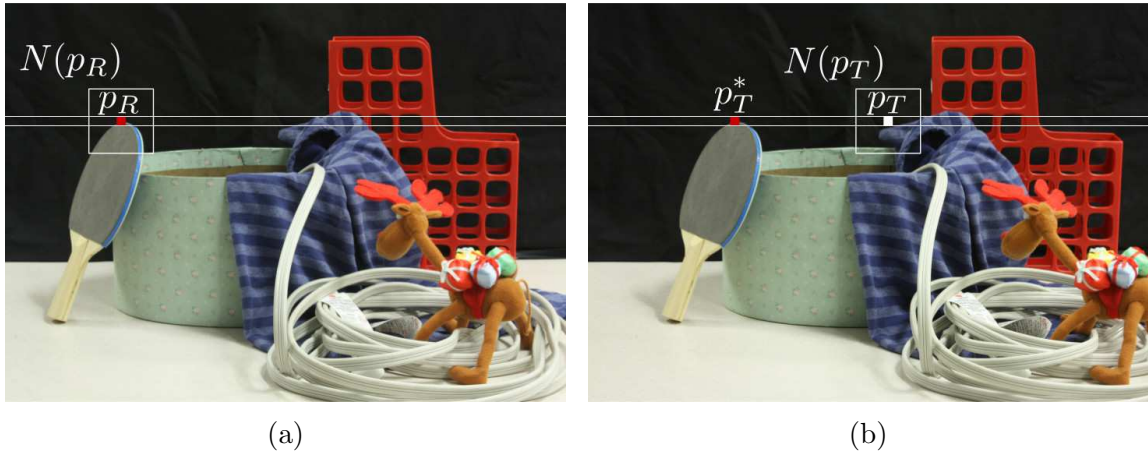


Figure 2.6: For a given rectified stereo pair and pixel p_R on the reference image (left), stereo matching local methods look for candidate pixels p_T in the target image (right) at the same row as p_R . For each candidate, a cost function is evaluated that penalizes dissimilarities between the content of a support region $N(p_R)$ surrounding p_R and the content of a corresponding support region $N(p_T)$ surrounding p_T . Images from the Middlebury Stereo Datasets 2014 [86].

window content [118].

Global methods usually aim at estimating the entire disparity map by minimizing a suitable cost function both accounting for local matches as well as enforcing some form of smoothness between disparity hypotheses at different points [100].

In general, global methods allow for finding more accurate matchings at a cost of an increased computational effort. As a drawback, the entire system is possibly prevented to run at real-time, thus making it inadequate for a number of possible applications.

The semi-global stereo matching algorithm by Hirschmuller [44] tries to combine advantages from both local and global approaches, achieving an accuracy comparable to top-ranked global algorithm while maintaining its complexity linear in the number of pixels and disparity levels.

2.4.2 A Locally Global Disparity Refinement

As mentioned in Section 2.4.1, several approaches have been proposed to solve the stereo correspondence problem using both local or global methods. Assuming a rectified binocular stereo pair, it is straightforward to derive the disparity of a point

p_R in the reference image once its conjugate point p_T has been found in the target image. Given the disparity, a corresponding depth value can easily be computed via triangulation as in (2.15).

Local methods based on Cost Aggregation (CA) (see Section 2.4.1) rely on a crucial assumption i.e. disparity values vary smoothly within neighboring points except at depth discontinuities. At the same time, different from global approaches, they do not enforce any constraint to account for mutual dependencies between disparity assumptions at different points. In other words, the aggregated disparity is estimated at a given point in a way that is completely unaware about the aggregate disparities estimated at other neighboring points, despite the piecewise smoothness assumption suggests that a strong correlation should exist between such assumptions.

The approach proposed by Mattoccia in [65] aims at explicitly model these dependencies and refine the disparity values estimated via local approaches by explicitly enforcing such dependencies.

To grasp the rationale behind the strategy proposed in [65], first consider a point p_R in the reference image for which a disparity value d has been estimated. Provided the disparity piecewise smoothness assumption holds, if p_R has disparity d then all neighboring points in a support window $S(p_R)$ centered at p_R are likely to have the same disparity as well, except for discontinuities. The same can be said about any other point in the reference image and in particular about any neighboring point $q_R \in S(p_R)$. Namely, for a given neighbor q_R , all points within its support $S(q_R)$ are implicitly assumed to have the same disparity as the one assigned to q_R by the CA algorithm. Moreover, if p_R has disparity d then its conjugate point p_T in the target image is implicitly assumed to have disparity $-d$ as well as its neighboring points $q_T \in S(q_T)$ following an analogous reasoning.

Provided a disparity value is estimated for each pixel in the reference image by the CA aggregation algorithm, as many assumptions are implicitly made about p_R disparity as the number of points in the reference image whose supports contain p_R . For fixed size support windows $n \times n$ pixels wide, n^2 assumptions are made, all potentially different.

The idea is to force these assumptions to be consistent with each other by applying a two-step procedure. First, a suitable function $\mathcal{P}_{p_R \rightarrow q_R}^R(d)$ is defined encoding for the plausibility of the disparity assumption d made by a point p_R in the reference image on a neighboring point q_R within its support

$$\mathcal{P}_{p_R \rightarrow q_R}^R(d) = e^{-\frac{\Delta_{p_R, q_R}}{\gamma_s}} e^{-\frac{\Delta_{p_R, q_R}^\psi}{\gamma_c}} e^{-\frac{\Delta_{p_T, q_T}}{\gamma_s}} e^{-\frac{\Delta_{p_T, q_T}^\psi}{\gamma_c}} e^{-\frac{\Delta_{q_R, q_T}^\omega}{\gamma_t}} \quad (2.16)$$

where Δ accounts for spatial proximity, Δ^ψ and Δ^ω account for color similarity

and γ_s , γ_c and γ_t control the behavior of the distributions (a detailed explanation is provided in [65]).

Similarly, a function $\mathcal{P}_{p_T \rightarrow q_T}^T(d)$ is defined to encode for the disparity assumptions made by points in the target image.

In the second step all assumptions made about the disparity of a given point are gathered by properly accumulating the plausibilities evaluated in the first step. More precisely, the following accumulated plausibilities $\Omega_{p_R}^R(d)$ and $\Omega_{p_T}^T(d)$ are computed

$$\begin{aligned}\Omega_{p_R}^R(d) &= \sum_{q_R \in S(p_R)} \mathcal{P}_{q_R \rightarrow p_R}^R(d) \\ \Omega_{p_T}^T(d) &= \sum_{q_T \in S(p_T)} \mathcal{P}_{q_T \rightarrow p_T}^T(d)\end{aligned}\tag{2.17}$$

and an overall plausibility $\Omega_{p_R}^{RT}(d)$ is obtained for each point p_R in the reference image via cross-check

$$\Omega_{p_R}^{RT}(d) = \Omega_{p_R}^R(d)\Omega_{p_T}^T(-d)\tag{2.18}$$

Finally, a winner-takes-all strategy is applied to select, for each point p_R , the disparity value maximizing the overall plausibility (2.18), i.e.

$$d^* = \arg \max_{d \in \{d_{min}, \dots, d_{max}\}} \Omega_{p_R}^R(d)\tag{2.19}$$

Chapter 3

ToF and Stereo Data Fusion

3.1 Introduction

There exist many different devices and algorithms for real-time depth estimation including active lighting devices and passive systems exploiting only regular cameras. The first family includes Time-of-Flight (ToF) sensors (see Section 2.3) and structured light cameras while the most notable example of the second family are the (binocular) stereo setups (see Section 2.4).

None of these solutions is completely satisfactory, active devices like Time-of-Flight and structured light cameras are able to robustly estimate the 3D geometry independently of the scene content but they have a limited spatial resolution, a high level of noise and a reduced accuracy on low reflective surfaces. Passive stereo vision systems, although widely used for the simple technology, have various limitations, in particular their accuracy strongly depends on the scene content and the acquisition is not very reliable on regions with uniform or repetitive texture. On the other side, passive stereo vision systems have a high resolution and a limited amount of noise. The characteristics of the two approaches are complementary and the fusion of data from the two systems has been the subject of several research studies in the last years.

In this chapter, a depth estimation algorithm combining together ToF and stereo data is proposed [27]. An effective solution for this task needs two fundamental tools: the estimation of the reliability of the data acquired by the two devices at each location and a fusion algorithm that exploits this information to properly combine the two data sources.

ToF sensors are typically affected by several sources of noise [119]. The reliability of ToF data has traditionally been estimated by using simple to sophisticated models

to describe such noise sources. As an example, shot noise can be estimated from the amplitude and intensity of the received signal, while other depth estimation issues specific of their working principles, like the presence of mixed pixels or the multi-path error, represent a much bigger challenge. Multi-path error due to light rays scattered multiple times before reaching the sensor is particularly difficult to be directly estimated and removed. The idea is to use a deep ConvNet to regress the confidence of ToF data rather than rely on traditional noise models.

Stereo data confidence is typically estimated with different metrics based on the analysis of the shape of the cost function [48]. While these metrics are able to capture the effects of the local matching cost computation, the same cannot be applied for the most recent stereo vision techniques. The latter typically exploit complex global optimization schemes whose behavior is hard to be captured by standard metrics. For this reason, coherently with the approach used for ToF data, a deep learning framework is adopted to for stereo confidence estimation.

Confidence estimates are exploited inside an extended version of the Local Consistency (LC) framework [15, 64] able to exploit such information to perform the fusion of the two data sources.

3.2 Related Works

Binocular stereo vision systems can estimate depth data from two standard images by exploiting the well known triangulation principle (see Section 2.4). A significant amount of research studies focused on this family of 3D data acquisition systems and a detailed review can be found in [104]. The depth estimation accuracy of these systems depends on many factors, including not only the specific matching algorithm used to estimate the disparity map but also the photometric content of the scene. In particular, the estimation is prone to errors in regions with fewer details, e.g. a planar wall with no texture, or repetitive patterns. For this reason it is important to estimate the reliability of the computed data. An exhaustive review about techniques for confidence estimation in stereo vision systems can be found in [48]. Notice how the confidence information for stereo systems used to be computed with deterministic algorithms based on the analysis of the matching cost function and only recently deep learning approaches have been exploited for this task [78, 89, 79].

ToF cameras have also attracted the attention of the research community working on depth acquisition systems [36, 83, 120, 76, 50, 31], since they can acquire depth information in real-time and many low cost devices using ToF principles are currently available in the consumer market. Differently from stereo vision systems, ToF cameras can estimate accurately the depth values also in regions without tex-

ture or with repetitive patterns since they do not rely uniquely on the scene content for depth estimation. On the other side, these devices have various limitations as the low resolution and high noise levels. Furthermore, as already mentioned, they are affected by systematic errors as multi-path interference, mixed pixels and noisy estimation on low reflective regions. In [50] it is possible to find a detailed analysis of the various error sources while [31] offers an insight on the effects of the low reflectivity of the scene on the depth estimation.

ToF cameras and stereo vision systems are based on different depth estimation principles and they have complementary strengths and weaknesses, suggesting that a fusion of the data acquired from the two sources can lead to a more reliable depth estimation. Different works on stereo-ToF depth fusion can be found in the literature, e.g., [30] and [119] present two complete reviews of the different approaches.

The combination of a ToF camera with a stereo vision system in order to estimate and then fuse two depth maps of the scene has been used in several works [55, 31, 24, 53]. In order to perform the fusion Zhu et al. [127, 126, 128] used a MAP-MRF Bayesian formulation where a global energy function is optimized with belief propagation. Dal Mutto et al. [11] also used a probabilistic formulation and computed the depth map with a ML local optimization. In a more recent version of the approach [14] a global MAP-MRF optimization scheme has been used in place of the ML optimization.

Nair et al. [71] instead used a variational fusion framework. An interesting contribution of this approach is the use of confidence measures for the ToF and stereo vision systems in order to control the process. Evangelidis et al. [20] estimate the depth information by solving a set of local energy minimization problems. Another solution [15] consists in using a locally consistent framework [65] (see Section 2.4.2) to fuse the two data sources. This approach has been improved in [64] by extending the LC framework driving the fusion process with the depth map confidences in order to take into account the different nature and reliability of the data sources. The same approach is also applied here. However, instead of computing confidence data based on heuristic cues a deep ConvNet is used instead for such estimation. More details about [64] can be found in the next Section 3.4.

Finally, it is worth to mention that a strictly related problem is the fusion of the data delivered by a ToF camera with data from a single color camera [17, 117, 116, 88, 26, 18]. For this task different strategies have been proposed, including methods based on bilateral filtering [117, 116], edge-preserving schemes [26] and methods exploiting confidence information for ToF data [88].

3.3 Proposed Method

The proposed algorithm starts by reprojecting ToF data on the stereo camera viewpoint and then upsamples the data to the spatial resolution of the stereo setup by using a combination of segmentation clues and bilateral filtering [15]. Then, confidence information for ToF depth data is estimated. For this task an ad-hoc Convolutional Network (ConvNet) has been developed that takes in input multiple clues, i.e., the stereo and ToF disparities, the ToF amplitude and the difference between the left image and the right one warped according to disparity information, providing a hint of the stereo matching accuracy, and jointly estimates both stereo and ToF confidence measures.

It is customary to use a good amount of data for the training of the ConvNet with the corresponding ground truth information. At the time of writing there are no available datasets containing these data and furthermore the acquisition of accurate ground truth data for real world 3D scenes is a challenging operation.

For this reason 55 different 3D synthetic scenes have been rendered using *Blender* [4], with examples of various acquisition issues including reflections and global illumination. Realistic stereo and ToF data have been simulated on the rendered scenes using LuxRender [62] and a simulator realized by Sony EuTEC starting from the simulator presented by Meister et al. [67]. This have been used to train the ConvNet that proved to be able to accurately estimate a confidence measure for both stereo and ToF depth acquisitions even in challenging situations like mixed pixels on boundaries and multi-path artifacts affecting ToF estimations. Finally, the two data sources are fused together. The proposed fusion algorithm has been derived from [64]. The framework extends the LC method [65] to combine the confidence measures of the data produced by the two devices. It computes a dense disparity map with subpixel precision by combining the two sources of information enforcing the local consistency of the measures weighted according to the computed confidence information. More details are given in Section 3.4.

The considered acquisition system is made of a ToF camera and a stereo vision system each producing an estimation of depth data from the corresponding viewpoint. The goal of the proposed method is to provide a dense depth map from the point of view of one of the color cameras of the stereo setup.

Two acquisition systems are assumed to have already been jointly calibrated (this can be done using ad-hoc techniques for this kind of setups, e.g., the one proposed in [11]). The algorithm includes four steps (see Figure 3.1):

1. The depth information acquired from the ToF sensor is firstly reprojected to the reference color camera viewpoint. Since ToF sensors have typically a lower

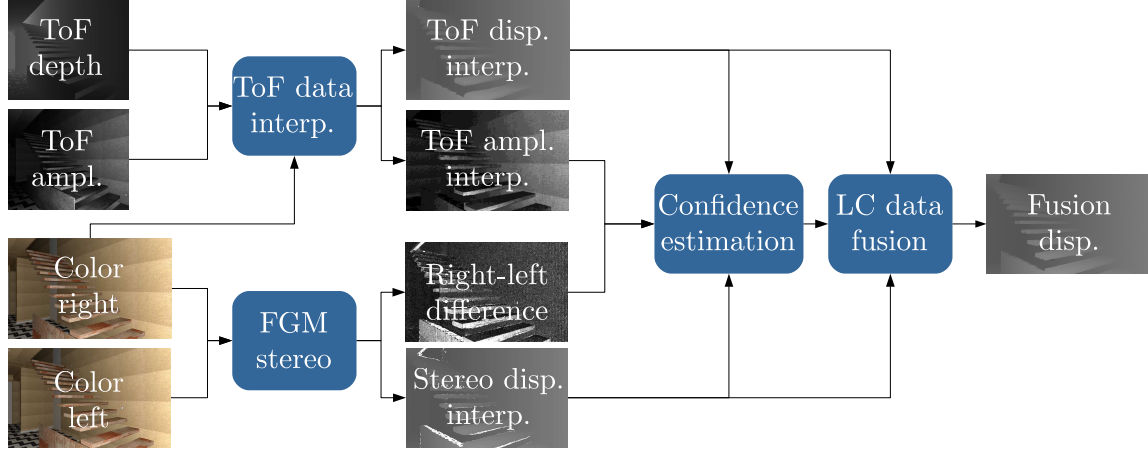


Figure 3.1: Flowchart of the proposed approach.

resolution than color cameras, it is also necessary to interpolate the ToF data. For this task the approach proposed in [15] has been used. More in detail the color image is firstly segmented using Mean-Shift clustering [8] and then an extended cross bilateral filter with an additional segmentation-based term besides the standard color and range ones is used to interpolate the data and to produce a high resolution depth map aligned with the color camera lattice. Since the fusion algorithm works in disparity space, the computed depth map is also converted to a disparity map with the well known inversely proportional relationship between the two quantities. For more details on the ToF reprojection and upsampling, the reader is referred to [15].

2. A high resolution depth map is computed from the two color views by applying a stereo vision algorithm. The proposed approach is independent of the stereo algorithm used to compute the disparity map, however for the current implementation the Semi-Global Matching (SGM) algorithm [44] has been used. This algorithm performs a 1D disparity optimization on multiple paths that minimizes an energy term made of point-wise or aggregated matching costs and a regularization term.
3. Confidence information for the stereo and ToF disparity maps are estimated using the Convolutional Network architecture of Section 3.6.
4. The upsampled ToF data and the stereo disparity are finally fused together using an extended version of the LC technique [65].

3.4 Locally Consistent ToF-Stereo Data Fusion

The Local Consistency (LC) method [15, 64] allows to exploit knowledge about the reliability of ToF and stereo data in order to better combine them into a more accurate disparity map. This method is based on the locally global approach presented by Mattocchia in [65] and already discussed in Section 2.4.2, which was originally conceived as a fast and effective tool to refine stereo matching data. In [15] this approach has been extended in order to be used with multiple disparity hypotheses, e.g. from a stereo system and da ToF camera, as is the case considered here. The idea behind is to start by considering each point p_R of the reference image that has at least one valid range measurement provided by the ToF camera or stereo system. The plausibility of the measure originated from each sensor is computed similarly to Equation 2.16 thus accounting for both color and spatial consistency of the data. Multiple plausibilities are then propagated to neighboring points and accumulated into overall plausibilities, cross-checked and normalized. A winner-takes-all strategy is finally used to compute the optimal disparity value. Although the extension proposed in [15] produces reasonable results, it has the limitation that it assigns the same weight to the two data sources, without accounting for their specific reliability. The approach proposed by [64] tries to overcome this drawback by assigning different weights to the plausibilities, depending on the confidence estimation for the considered depth acquisition system computed at the considered point, i.e.

$$\Omega_{p_R}(d) = \sum_{q_R \in \mathcal{N}(p_R)} \left(P_T(q_R) \mathcal{P}_{q_R \rightarrow p_R}^T(d) + P_S(q_R) \mathcal{P}_{q_R \rightarrow p_R}^S(d) \right) \quad (3.1)$$

where $\Omega_{p_R}(d)$ is the plausibility at point p_R in the reference image for depth hypothesis d , $\mathcal{P}_{q_R \rightarrow p_R}^T(d)$ is the plausibility propagated by neighboring points q_R according to ToF data and $\mathcal{P}_{q_R \rightarrow p_R}^S(d)$ is the one according to stereo data.

Finally, for any given point p , $P_T(p)$ and $P_S(p)$ are the ToF and stereo confidence values at that location.

Different from [64] where confidence information comes from a deterministic algorithm based on the noise model for the ToF sensor and from the cost function analysis for the stereo system, in the proposed approach the confidence is estimated with a ConvNet whose architecture is detailed in Section 3.6.

3.5 SYNTH3: A ToF and Stereo Synthetic Dataset

A synthetic dataset called *SYNTH3* has been developed for supervised machine learning applications. The dataset is split into two parts, a training set and a test

set. The training set contains 40 scenes, including 20 unique scenes plus 20 more scenes obtained from the former by rendering them from different viewpoints. It is worth noticing that, even if the size of the dataset is quite small if compared to the size of datasets used at the moment in other fields, it is still the largest dataset for stereo-ToF fusion currently available, with scenes encompassing a large range of different characteristics. Additional 15 unique scenes are included in the test set.

Each synthetic scene is realized using the *Blender* 3D rendering software [4]. *Blender* scenes have been downloaded from the *BlendSwap* website [5], appropriately adjusted and finally rendered into virtual cameras thus generating the stereo-ToF data examples included in the dataset. The various scenes contain furnitures and objects of several shapes relative to different environments e.g., living rooms, kitchen rooms or offices. Furthermore, some outdoor locations with non-regular structure are also included in the dataset. In general, they appear realistic and suitable for the simulation of Stereo-ToF acquisition systems. The depth range across the scenes goes from about 50[cm] to 10[m], providing a large range of measurements.

Each scene has been rendered by simulating a stereo system with characteristics resembling the ones of the ZED stereo camera [122] and a ToF camera with characteristics similar to a Microsoft Kinect v2 [90, 119]. The stereo system is made by two Full-HD (1920×1080) color cameras with a baseline of 12[cm] and optical axes and image planes parallel to each other so that to virtually acquire already rectified stereo image pairs. Also the image plane and optical axis of the Kinect sensor are parallel to those of the ZED cameras and the Kinect sensor is placed under the right camera of the stereo system at a distance of 4[cm]. The considered acquisition system is depicted in Figure 3.2 that shows the relative positions of the 2 cameras and ToF sensor. Table 3.1 summarizes the parameters of the acquisition system.

	Stereo setup	ToF camera
Resolution	1920×1080	512×424
Horizontal FOV	69°	70°
Focal length	3.2[mm]	3.66[mm]
Pixel size	2.2[μm]	10[μm]

Table 3.1: Parameters of the stereo and ToF subsystems.

For each scene, the following data are provided: 1) the 1920×1080 color image acquired by the left camera of the stereo system, 2) the 1920×1080 color image acquired by the right camera of the stereo system, 3) the depth map estimated by the ToF sensor on the synthetic scene, 4) the relative amplitude map of the ToF sensor.

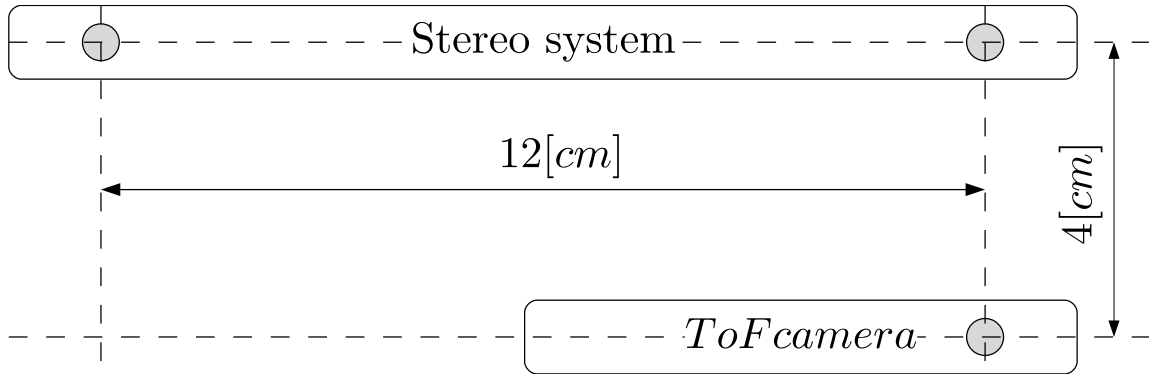


Figure 3.2: Representation of the Stereo-ToF acquisition system. The Figure shows the relative position of the color cameras and ToF camera.

The color images have been generated directly in *Blender* using the 3D renderer *LuxRender* [62]. The data captured by the Kinect ToF camera have been obtained by using the *ToF-Explorer* simulator developed by Sony EuTEC. The *ToF-Explorer* simulator was first designed according to the ToF simulator presented by Meister et al. in [67] that accurately simulate the data acquired by a real ToF camera including different sources of error as shot noise, thermal noise, read-out noise, lens effect, mixed pixels and the interference due to the global illumination (multi-path effect). The ToF simulator uses as input the scene information generated by *Blender* and *LuxRender*.

Moreover, the dataset contains also the scene depth ground truth relative to the point of view of both the Kinect and the right color camera of the stereo system. The dataset is publicly available at [99].

3.6 ConvNet for Joint ToF and Stereo Confidence Estimation

A Convolutional Network (ConvNet) model can be used to regress the confidence information required by the LC fusion algorithm of Section 3.4 in order to produce the final disparity map. The proposed ConvNet takes as input both ToF and stereo clues and outputs a confidence map for each of the two devices.

Such clues can be easily computed through a fast pre-processing stage from the ToF and stereo disparity maps, the ToF amplitude and the color image pair. The process operates by first applying suitable projections and color conversions to the

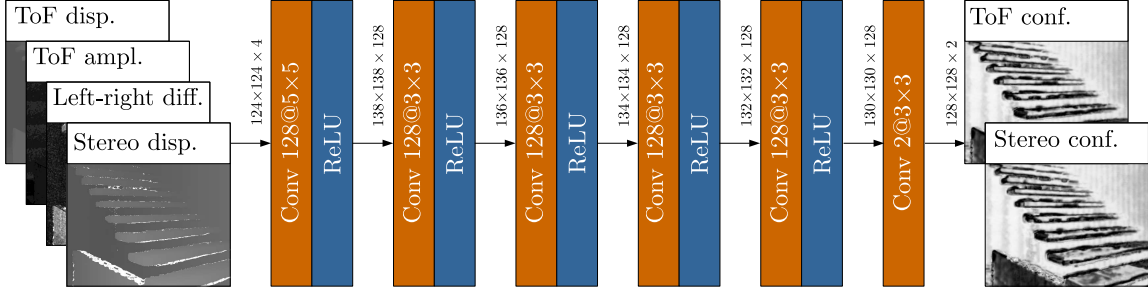


Figure 3.3: Architecture of the proposed Convolutional Network (ConvNet). The size of the outputs produced at the end of each convolutional layer is also reported for the case where a single 4-channel training patch of size 142×142 is fed as input to the ConvNet.

raw data:

- $D_{T,i}$ the ToF disparity map reprojected on the reference camera of the stereo vision system.
- $A_{T,i}$ the ToF amplitude image reprojected on the reference camera of the stereo vision system.
- $D_{S,i}$ the stereo disparity map.
- $I_{R,i}$ the right stereo image converted to grayscale.
- $I_{L',i}$ the left stereo image converted to grayscale and reprojected on the right camera using the disparity computed by the stereo algorithm.

The four clues are then derived as follows. The first clue, $\Delta'_{LR,i}$, is extracted from the left and right grayscale images $I_{L',i}$ and $I_{R,i}$ in a two-step procedure. First, the absolute difference between their scaled versions is computed:

$$\Delta_{LR,i} = \left| \frac{I_{L',i}}{\mu_{L,i}} - \frac{I_{R,i}}{\mu_{R,i}} \right| \quad (3.2)$$

where the scaling factors $\mu_{L,i}$ and $\mu_{R,i}$ are the averages calculated on the left and right images respectively. The value returned by Equation (3.2) is then divided by $\sigma_{\Delta_{LR}}$, the average of the standard deviations calculated for each $\Delta_{LR,j}$ for j varying across the scenes in the training set:

$$\Delta'_{LR,i} = \Delta_{LR,i} / \sigma_{\Delta_{LR}} \quad (3.3)$$

The other three clues $D'_{T,i}$, $D'_{S,i}$ and $A'_{T,i}$ are obtained straightforwardly from ToF and stereo disparities and ToF amplitude by applying a normalization similar to the one in Equation (3.3), that is:

$$D'_{T,i} = D_{T,i}/\sigma_{D_T} \quad (3.4)$$

$$D'_{S,i} = D_{S,i}/\sigma_{D_S} \quad (3.5)$$

$$A'_{T,i} = A_{T,i}/\sigma_{A_T} \quad (3.6)$$

where σ_{D_T} , σ_{D_S} and σ_{A_T} are the average of the standard deviations calculated for each disparity or amplitude representation in the training set. Finally, the four clues $\Delta'_{LR,i}$, $D'_{T,i}$, $D'_{S,i}$ and $A'_{T,i}$ are concatenated in a four-channel input image which is fed to the ConvNet in order to produce two confidence maps P_T and P_S for ToF and stereo data respectively.

The inference process is better explained by Figure 3.3 that shows the architecture of the proposed ConvNet. It contains a stack of six convolutional layers each followed by a ReLU non-linearity with the exception of the last one. The first five convolutional layers have 128 filters each, the first layer has a window size of 5×5 pixels while all others have a window size of 3×3 pixels. The last convolutional layer has only two filters, producing as output a two-channels image where the two channels contain the estimated ToF and stereo confidence respectively. Notice that, in order to produce an output with the same resolution of the input, no pooling layers have been used. At the same time, to cope with the boundary effect and size reduction introduced by the convolutions, each input image is padded by 7 pixels along their spatial dimensions, where the padded values are set to be equal to the values at the image boundary.

3.7 Experimental Results

The proposed fusion algorithm has been trained and evaluated on the training set and test set of the SYNTH3 dataset (see Section 3.5) respectively.

As pointed out in Section 3.5, the test set contains 15 different scenes. The thumbnails of the various scenes are shown in Figure 3.4, notice how they contain different challenging environments with different acquisition ranges, complex geometries and strong reflections. Also different materials, both textured and un-textured have been used. The acquisition setup and the camera parameters are the same of the training set discussed in Section 3.5. Ground truth data have been computed by extracting the depth map from the *Blender* rendering engine and then converting

it to the disparity space. The algorithm takes in input the 512×424 depth and amplitude maps from the ToF sensor and the two 960×540 color images from the cameras. The color cameras resolution has been halved with respect to the original input data in the dataset. The output is computed on the point of view of the right camera at the same (higher) resolution of color data and it has been cropped to consider only on the region that is framed by all the three sensors.









































Input Scene		ToF		Stereo		Fusion	
Color view	Ground truth	Disparity	Error	Disparity	Error	Disparity	Error
							
							
							
							
							

Figure 3.4: Results of the proposed fusion framework on 5 sample scenes (one for each row). In error images, gray pixels correspond to points excluded since they are not valid on one of the disparity maps. The intensity of red pixels is proportional to the absolute error. (*Best viewed in color*)

In order to train the ConvNet of Section 3.6, a large set of training examples has been generated by randomly extracting a number of patches from each scene in the training set. Each patch has a size of 128×128 pixels (that becomes 142×142 after padding). During this process, a set of standard transformations have also been applied to augment the number of training examples and ease regression, namely rotation by ± 5 degrees, horizontal and vertical flipping. In the experiments, 30

patches from each of the 40 scenes included in the training set have been extracted, and considering also their transformed versions at the same corresponding location a total of 6000 patches have been obtained.

Both ToF and stereo ground truth confidence maps have been generated from the absolute error of the two disparity estimations against the disparity ground truth of the scene, that is available in the dataset. More specifically, each ground truth confidence map has been computed by first clipping all values above a given threshold in the corresponding disparity absolute error map, then dividing by the same threshold in order to obtain values in the range $[0, 1]$.

The network has been trained to minimize a loss function defined as the Mean Squared Error (MSE) between the estimated ToF and stereo confidence maps and their corresponding ground truth. To this aim, a standard Stochastic Gradient Descent (SGD) optimization has been employed with momentum 0.9 and batch size 16. An initial set of weight have been derived with Xavier’s procedure [28]. As for the learning rate, an initial value of 10^{-7} has been used, subject to a constant decay by a coefficient of 0.9 every 10 epochs. The network has been implemented using the MatConvNet framework [108]. The training of the network took about 3 hours on a desktop PC with an Intel i7-4790 CPU and an NVIDIA Titan X (Pascal) GPU.

Before evaluating the performances of the fusion scheme, the confidence information computed with the proposed ConvNet used to control the fusion process is analyzed. Some confidence maps computed with the proposed ConvNet are shown in Figure 3.5. The second column shows the ToF confidence, it is noticeable how the ConvNet is able to estimate the areas of larger error by assigning low confidence (darker pixels in the images). A first observation is that in most of the confidence maps it is possible to see how the error is larger in proximity of the edges. It is a well-known issue of ToF sensors due to the limited resolution and due to the mixed pixel effect. Furthermore, by looking for example at rows 2 and 4, it is visible how the ConvNet has also correctly learned that the ToF error is higher on dark surfaces due to the lower reflection (e.g., on the dark furniture in row 2 or on the black squares of the checkerboard in row 4, or on some of the rocks in row 1). The multi-path is more challenging to be detected, but row 4 shows how the confidence is lower on the wall edges or behind some stairs element in row 3. Concerning the stereo confidence the results are also good. Also in this case the limited accuracy on edges is detected and a low confidence is assigned. Furthermore, some surfaces with uniform or repetitive patterns have a lower confidence, e.g., some rocks in row 1.

The computed confidence information is then used to drive the fusion process. Figure 3.4 shows the output of the proposed algorithm on some sample scenes. Column 1 and 2 show a color view of the scene and the ground truth disparity data.

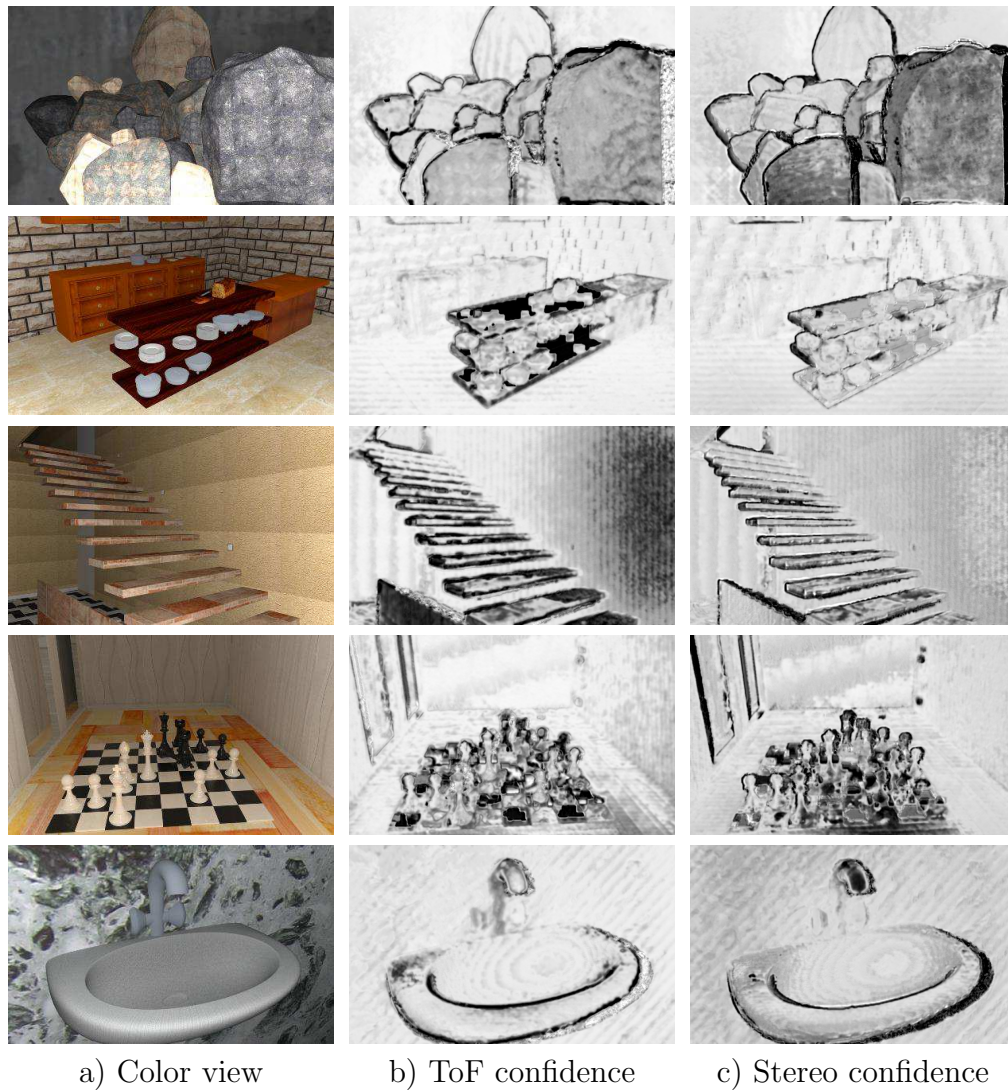


Figure 3.5: Confidence information computed by the proposed deep learning architecture for some sample scenes: a) Color view; b) Estimated ToF confidence; c) Estimated stereo confidence. Brighter areas correspond to higher confidence values, while darker pixels to low confidence ones.

The up-sampled, filtered and reprojected ToF data are shown in column 3, while column 4 contains the corresponding error map. Columns 5 and 6 show the disparity estimated by the stereo vision algorithm and the corresponding error map. Concerning stereo data, the OpenCV implementation of the SGM stereo algorithm has been used with pointwise Birchfield-Tomasi metric, 8 paths for the optimization and a window size of 7×7 pixels. The fused disparity map and its relative error are shown in columns 7 and 8. Starting from ToF depth data, this is the more accurate of the two data sources, the filtered and interpolated data is quite accurate, even if there are issues in proximity of edges that are sometimes not too accurate. Also low-reflective surfaces like the black checkers in row 4 are very noisy and sometimes not acquired at all. The multi-path affects some regions like the steps of the stairs. Stereo based disparity maps usually have sharper edges but there are several artifacts due to the well-known limitations of this approach. The fusion algorithm reliably fuse the information coming from the two sensors providing a depth map with less artifacts on edges and free from the various problems of the stereo acquisition.

A quantitative performance evaluation is shown in Table 3.2 and confirms the visual analysis. The table shows the RMS in disparity space averaged on all the 15 scenes. For a fair comparison, only pixels with a valid disparity value in all the compared disparity maps (stereo, ToF and fused disparities) have been considered valid. By looking at the averaged RMS values, the ToF sensor has a high accuracy with a RMS of 2.19, smaller than the RMS of 3.73 of the stereo system. This is a challenging situation for fusion algorithms since it is difficult to improve the data from the best sensor without affecting it with errors from the less efficient one. However confidence data helps in this task and the proposed approach is able to obtain a RMS of 2.06 with a noticeable improvement with respect to both sensors. Comparison with state-of-the-art approaches is limited by the use of the new dataset and the lack of available implementations of the competing approaches. However, the approach has been compared with the highly performing method of Marin et al. [64]. This approach has a RMS of 2.07, higher than the one of the proposed method even if the gap is limited and the results comparable. The method of [64] outperforms most state-of-the-art approaches, so also the performances of the proposed method are expected to be competitive with the better performing schemes, with the advantage that the proposed approach does not involve heuristics.

Method	RMS Error
Interpolated ToF	2.19
SGM Stereo	3.73
Proposed ToF-Stereo Fusion	2.06
Marin et al. [64]	2.07

Table 3.2: RMS in disparity units with respect to the ground truth for the ToF and stereo data, the proposed method and [64]. The error has been computed only on non-occluded pixels for which a disparity value is available in all the methods.

Chapter 4

Segmentation and Semantic Labeling

4.1 Introduction

Recent achievements in the computer vision field allowed to obtain a relevant improvement in algorithms dealing with the semantic segmentation task, boosted by two key advancements in particular. The first is the development of more powerful machine learning algorithms, specially deep learning techniques, that allowed to better understand the semantic content of the images. The second is the introduction of consumer depth sensors that allowed to easily acquire the 3D geometry of the scene, a very useful source of information overcoming several limitations and ambiguities of color information.

Clustering techniques, e.g., normalized cuts spectral clustering [93], are an effective approach for segmentation well-suited for the extension to the joint segmentation of color and geometry information [12]. However, the normalized cuts algorithm has a bias towards producing regions of similar sizes and for this reason it is challenging to properly separate all the objects avoiding at the same time to over-segment the scene.

The problem can be solved by exploiting an over-segmentation performed with normalized cuts followed by an iterative region merging approach scheme. This work follows this rationale and uses together two different cues in order to decide which segments must be merged. The first is a segment similarity measure obtained from the descriptors computed by a Convolutional Network (ConvNet). The other is obtained, for a given couple of segments, by fitting a Non-Uniform Rational B-Spline (NURBS) on each segment taken separately and on their union. The fitting

accuracies are then compared and the two segments are merged whenever their union results in an increased fitting accuracy [74]. Notice how this idea allows to detect if the two segments are part of the same scene surface (and thus are candidate to be merged) and to properly handle also non-planar object and surfaces.

The proposed approach has been presented in [69] first, then in [72] where an enhanced classification algorithm has been proposed that also addresses the semantic labeling task besides segmentation. In particular, a deeper Convolutional Network (ConvNet) architecture has been employed in [72], which has been fed with surface curvatures together with fitting error (this is the first time this kind of data is used in a deep learning framework), other than color and HHA descriptors [35] (disparity, height and orientation angle) instead than orientation data.

4.2 Related Works

Segmentation of RGB-D data has been the subject of many research works (a recent review is contained in [119]). Clustering techniques are commonly used for image segmentation and they have been exploited for the combined segmentation of color and geometry by using multi-channel feature vectors [13, 12]. The method of [51] performs multiple clusterings with K-means and combines them together.

Region splitting and rowing methods are another commonly used approach. The approach of [97] starts from an over-segmentation and combines segments corresponding to the same planar region by exploiting a method based on Monte Carlo Markov Chain and Rao Blackwellization. The method of [73] exploits region splitting and iteratively refines the segmentation by recursively splitting regions that do not correspond to a single surface. The work of [74] uses the same criteria in a bottom-up approach starting from an over-segmentation of the scene. Gupta et al. [33] use a hierarchical segmentation starting from edge detection information. The method of [94] starts with an over-segmentation computed with watersheds and then exploits a hierarchical approach. Hasnat et al. [37, 38] start from a joint clustering method on the color, geometry and orientation data and then apply a region merging algorithm searching for planar regions. Finally, [102] uses dynamic programming to extract planar surfaces.

A closely related problem is semantic segmentation, i.e., joint segmentation and labeling of the segments. This problem is typically solved by using machine learning approaches. Ren et al. [84] exploit an over-segmentation with Markov Random Fields followed by a tree-structured algorithm. The works of [16] and [60] instead use Conditional Random Fields (CRF). The method of [52] also uses a CRF model that captures planar surfaces and dominant lines. Another work based on CRFs is

[42], that combines them with decision forests. The approach of [16] combines CRF with mutex constraints based on geometry data while the approach of [60] combines 2D segmentation, 3D geometry data and contextual information. The work of [3] is instead based on a proposal process that generates spatial layout hypotheses followed by a sequential inference algorithm.

Recently, deep learning algorithms have been exploited for the semantic segmentation task [45, 10, 61]. One of the first solutions exploiting deep learning is [10], that uses a multiscale Convolutional Network. The method of [61] is able to achieve a very high accuracy by exploiting Fully Convolutional Networks. The approach of [35] uses a ConvNet working on geometric features. Wang et al. [111] use two different ConvNets, one for color and one for depth, and a feature transformation network able to separate the information common to the two modalities from the one specific of each modality. The work of [19] jointly solves the semantic labeling together with depth and normal estimation using a multiscale ConvNet. Finally the method of [110] uses deep learning to extract superpixel features that are then classified with SVMs.

The combination of segmentation and semantic labeling has been considered in [107], that employs multiple segmentations to generate the regions to be used for object detection and recognition. Multiple segmentations are used also by [7] that deals with the problem of object segmentation using a sequence of constrained parametric min-cut problems.

Even if several different approaches for this task have been proposed, the proposed method has some original features not present in previous works, in particular the usage of surface fitting cues and the strict coupling between the semantic labeling and the segmentation, with the idea of exploiting the deep learning descriptors to control the iterative merging together with fitting cues. This allows it to obtain very accurate results even if the deep learning framework is simpler than some of the related works. Furthermore, while many related works strongly rely on the planar surface assumption the proposed model properly accounts for arbitrarily shaped regions.

4.3 Proposed Method

The proposed method is organized in 3 main blocks as shown in Figure 4.1. Color and depth data are used to compute a set of nine-dimensional vectors containing the 3D location, the surface normal information and the color representation for each sample. Then the algorithm computes an over-segmentation of the scene using the 9D vectors exploiting a spectral clustering algorithm derived from [12, 74] (Section 4.4). After

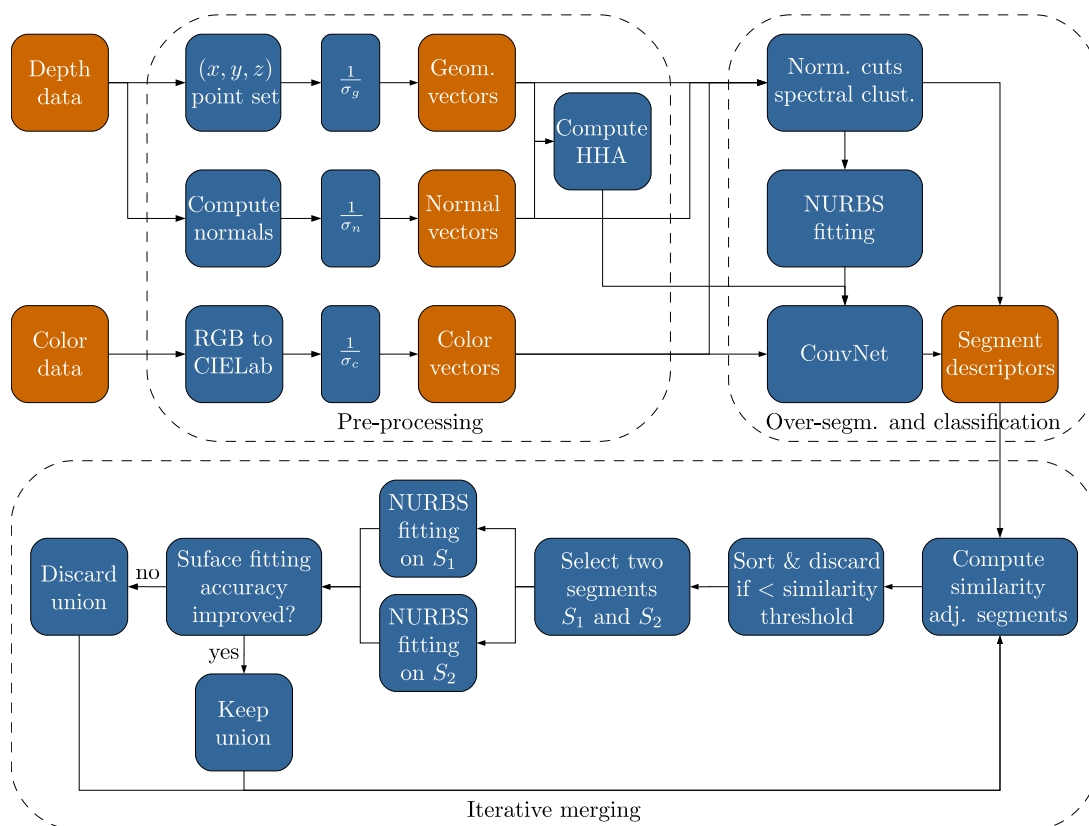


Figure 4.1: General architecture of the proposed method.

performing the over-segmentation, a NURBS surface is fitted over each segment using the approach detailed in Section 4.5. The fitting error and the curvature information for the fitted surfaces are also extracted. This information is fed to a Convolutional Network together with color and HHA descriptors. The network (Section 4.6) is trained for the semantic labeling task and computes a descriptor vector for each sample representing the probabilities of the various classes at the pixel location. The descriptors are aggregated inside each segment in order to obtain a unique descriptor for the segment. The third step is an iterative region merging algorithm (Section 4.7). It firstly analyzes the segments and computes an adjacency map. In the map two segments are marked as adjacent if they are connected and have compatible color and geometry data on their shared boundary. The similarity between ConvNet descriptors is then used to sort the couples of adjacent segments. Couples with a low similarity score according to the ConvNet descriptors are discarded and the remaining ones are processed in order of similarity. After selecting a couple, a NURBS surface is fitted over the merged region obtained by joining the two segments and the accuracy of the fitting is compared with the ones of the two segments. If the fitting error decreases after the merging (a hint that the two segments belong to the same surface) the merging is performed, otherwise the operation is discarded. The algorithm proceeds iteratively until there are no more segments to merge. Finally the probability vectors from the ConvNet are used to assign a label to each of the final segments in order to get also the semantic information. The obtained results are presented in Section 4.8.

4.4 Over-segmentation

To segment the acquired scene, a multi-dimensional vector enclosing the color and spatial information is built for every pixel p_i of the input image with valid depth information. The first three components $L(p_i), a(p_i), b(p_i)$ contain the color information in the perceptually uniform CIELab space. Then, the 3D position $x(p_i), y(p_i), z(p_i)$ and the surface normal $n_x(p_i), n_y(p_i), n_z(p_i)$ are considered (the 3D coordinates are calculated based on the sensor calibration information while for normal computation the approach of [46]) has been used. To achieve a consistent representation for these different types of information, the color, geometry and orientation average standard deviations σ_c, σ_g and σ_n are computed from the input image and depth map. Then, each of the 3 set of components is normalized by the corresponding standard deviation, providing normalized vectors $[\bar{L}(p_i), \bar{a}(p_i), \bar{b}(p_i)]$, $[\bar{x}(p_i), \bar{y}(p_i), \bar{z}(p_i)]$ and $[\bar{n}_x(p_i), \bar{n}_y(p_i), \bar{n}_z(p_i)]$ and finally the nine-dimensional representation:

$$\mathbf{p}_i^{9D} = [\bar{L}(p_i), \bar{a}(p_i), \bar{b}(p_i), \bar{x}(p_i), \bar{y}(p_i), \bar{z}(p_i), \bar{n}_x(p_i), \bar{n}_y(p_i), \bar{n}_z(p_i)]. \quad (4.1)$$

These 9D vectors representing the acquired scene are then segmented [12, 74] by means of Normalized Cuts spectral clustering [93] with Nyström acceleration [23]. The algorithm parameters are set in order to create a large number of segments (over-segmentation), since the final result will be produced by the merging procedure of Section 4.7.

The following step is the approximation of each segment with a Non-Uniform Rational B-Spline (NURBS) surface [77]. This is used twice in the proposed method: firstly, in order to produce an additional set of input cues for the ConvNet classifier (Section 4.6), secondly, in order to evaluate if segments produced by the merging operations correspond to a single scene object (Section 4.7).

4.5 NURBS Fitting

By means of NURBS it is possible to approximate each segment (including non-planar regions) with a continuous parametric surface $\mathbf{S}(u, v)$, computed by solving an over-determined system of linear equations in the least-squares sense. The reader is referred to [74] for the details on how the NURBS parameters are selected (e.g., the degrees) and how the linear system has been set up. It is worth noting that, in this formulation, the number of surface control points, corresponding to the degrees of freedom of the model, is proportional to the number of pixels in the segment to approximate. This prevents the fitting accuracy to be better on smaller segments, favoring over-segmentation in the final result [74]. Moreover, the usage of NURBS surfaces provides a geometric model suitable for arbitrary shapes, differently from several competing schemes [102, 97] that are more appropriate for scenes where most surfaces are planar.

After fitting the NURBS surfaces, two additional clues can be associated to each sample. The first one is the fitting error, i.e., the distance between each 3D position acquired by the sensor and the corresponding location on the fitted surface. This will be used both as an input for the ConvNet classifier and to recognize if a segment contains a single object (for segments, the Mean Squared Error will be considered). Notice how a large fitting error is a hint of the fact that a segment covers multiple objects, as exemplified in Figure 4.2. The second one is given by the two principal curvatures (i.e., the maximum and minimum local curvature values, see [106]) at each pixel location. These quantities are intrinsically related to the geometric shape

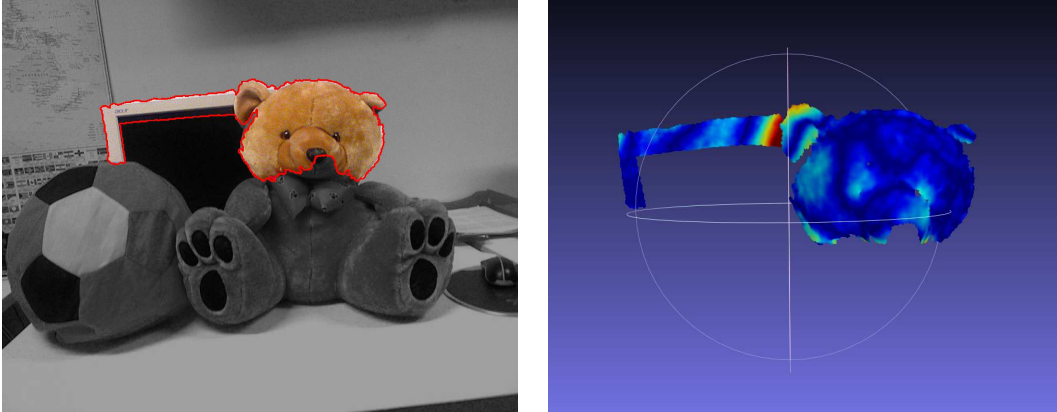


Figure 4.2: Fitting error for a NURBS surface approximating a segment containing 2 different objects. The red areas correspond to larger fit error. Notice how the large fit error between the teddy head and the monitor reveals that the two segments do not actually belong to the same object.

of the fitted surface, hence they have been used as an additional input channel for the ConvNet classifier.

4.6 A ConvNet for Semantic Labeling

A machine learning stage is employed in this step in order to produce classification data for the input scene, that is used not only to produce the semantic labels but also to decide which regions of the over-segmentation should belong to the same segment in the final segmentation.

The idea is to exploit the output of a Convolutional Network (ConvNet) trained for semantic image segmentation in order to compute a pixel-wise high-level description of the input scene. Specifically, a descriptor vector is associated to each pixel by considering the final layer of the network, a standard softmax classifier. This information is then used to compute a similarity score between couples of adjacent segments and, at the same time, to provide the input image with semantic labels. In particular, the proposed similarity score is exploited to drive the merging procedure detailed in Section 4.7, using it both to decide whether any two adjacent segments should be merged together as well as to determine the order in which candidate couples of segments are selected for the merging operations.

The ConvNet takes in input various cues:

- Color data, represented by the 3 components in the RGB color space.
- The geometry information. It is represented using three channels containing, for each sample, the horizontal disparity h_1 , the height above the floor h_2 , and the angle of the normal with the vertical direction a . This representation, typically abbreviated with HHA, has been introduced by [35] and provided better performances than the direct usage of geometry and orientation information.
- Surface fitting information, represented with a 3D vector containing the fitting error f and the two principal curvatures c_1 and c_2 .

For each point of the scene, a 9D vector is used to store this information as

$$\mathbf{p}_i^{cn} = [R(p_i), G(p_i), B(p_i), h_1(p_i), h_2(p_i), a(p_i), f(p_i), c_1(p_i), c_2(p_i)]. \quad (4.2)$$

Finally, the vectors are arranged over the image pixel lattice to produce, for each scene in the dataset, a 9-channel input representation.

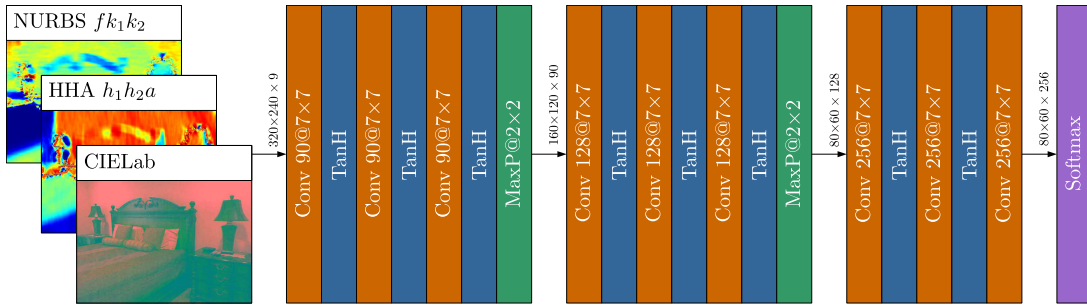


Figure 4.3: Layout of the proposed Convolutional Network.

An overview of the employed network structure is shown in Figure 4.3. The network has been constructed starting from the architecture employed in [21, 9, 69] by extending each of the original layers into a group of 3 layers. In order to avoid a too large increase in computation time no multi-scale input representation has been used. A sequence of convolutional layers is applied in order to extract a local representation of the input.

More in detail, each 9-channel input image passes through nine convolutional layers, arranged into three main blocks each one containing three layers (see Figure 4.3). Each block corresponds to one of the layers in the previous approach [69] and works with a constant resolution and number of filters. Moving from one block

to the next instead the resolution is reduced of a factor of 2 and the number of filters increases. Every block is made up to three convolutional layers (CONV) each followed by a hyperbolic tangent activation function (TANH). The first two blocks have also a final max-pooling (MAXP) layer, while the last convolutional layer of the last block does not have any activation function. Finally, a pixel-wise softmax classifier is applied on top of the last convolutional layer.

In order to reduce the computation time, input data are fed to the ConvNet at the reduced resolution of 320×240 . The convolutional layers have 90 filters in the first block, 128 in the second block and 256 in the last one (notice that the layers in the last blocks work at a lower resolution, thus an higher number of filters can be used without affecting too much the computation time). All filters have a size of 7×7 pixels, while the final softmax classifier has a weight matrix of size 256×14 and no bias.

The first layer filters are arranged into 9 groups so that filters in the i -th group are connected to the i -th input channel only. Also, local contrast normalization is applied to each input channel independently, allowing filter weights in the first convolutional layer to converge faster.

The network is trained to produce a semantic segmentation of the input image by labeling each pixel in the scene with one out of 14 different semantic labels. To this aim, a multi-class cross-entropy loss function is minimized throughout the training process.

4.7 Region Merging

The next step is the merging procedure, that starts from the initial over-segmentation and iteratively joins couples of segments to finally obtain the objects of the scene. The process is visualized in the bottom half of Figure 4.1 and summarized in Algorithm 1.

The procedure first identifies the couples of segments that are suitable to be merged. To this aim, it creates an adjacency matrix, storing for each couple of segments whether they are *adjacent* (that is, candidate for merging) or not. Two segments are considered as *adjacent* if the following conditions hold (see [74] for additional details):

1. They must be neighboring on the grid given by the depth map.
2. The depth information must be compatible along the shared boundary. Precisely, the difference ΔZ_i between the depth values on the two sides of the edge

is computed for each point P_i in the common boundary C_C . This difference must be smaller than a threshold T_d along at least half of the boundary, i.e.,:

$$\frac{|P_i : (P_i \in C_C) \wedge (\Delta Z_i \leq T_d)|}{|P_i : P_i \in C_C|} > 0.5 \quad (4.3)$$

3. Also the color information must be consistent. A condition similar to the one used for the depth data is required for the color difference in the CIELab space ΔC_i with a threshold T_c :

$$\frac{|P_i : (P_i \in C_C) \wedge (\Delta C_i \leq T_c)|}{|P_i : P_i \in C_C|} > 0.5 \quad (4.4)$$

4. The same is required for the orientation information, the angle between the two normal vectors $\Delta \theta_i$ being compared to a threshold T_θ :

$$\frac{|P_i : (P_i \in C_C) \wedge (\Delta \theta_i \leq T_\theta)|}{|P_i : P_i \in C_C|} > 0.5 \quad (4.5)$$

If the above conditions are fulfilled, the two segments are considered as adjacent. If necessary in order to reduce computation time, this procedure can be dropped and replaced by the assumption that all the connected segments are adjacent with a limited impact on the algorithm performances.

Subsequently, for each couple of adjacent segments the similarity $b_{i,j}$ is computed from the information inferred during the machine learning stage. The exploited idea is that, apart from predicting the semantic labels, the output of the softmax classifier can also be used to produce a descriptor vector associated to each segment and, in the end, to compute a similarity score for any couple of segments. For each pixel p_i , a descriptor vector $\mathbf{c}_i = [c_i^1, \dots, c_i^{14}]$ is extracted from the output of the softmax classifier. Notice that a linear interpolation is applied in order to resize the output from its actual size (i.e., $80 \times 60 \times 14$ pixels) to the size of the input image. Each descriptor vector can be considered as a discrete probability distribution (PDF) associated to the corresponding pixel, since its elements are non-negative values summing up to 1.

A probability density function $\mathbf{s}_i = [s_i^1, \dots, s_i^{14}]$ can be associated to each segment S_i as well, by simply computing the average of the PDFs associated to the pixels belonging to the segment, i.e.,

$$\mathbf{s}_i = \frac{\sum_{j \in S_i} \mathbf{c}_j}{|S_i|}. \quad (4.6)$$

Given two segments S_i and S_j , an effective approach in order to estimate their similarity is to compute the Bhattacharyya coefficient between their PDFs \mathbf{s}_i and \mathbf{s}_j respectively, i.e.,

$$b_{i,j} = \sum_{t=1, \dots, 14} \sqrt{s_i^t s_j^t}. \quad (4.7)$$

As an example, Figure 4.4 depicts the proposed similarity score between touching segments on a sample image. The lower is the similarity $b_{i,j}$ between segments, the darker is the color of the boundary between them. As can be seen in Figure 4.4a, different objects usually have a low similarity value while segments belonging to the same object typically share higher $b_{i,j}$ values (lighter boundaries). Notice how in Figure 4.4b the boundaries between segments at the very end of the merging stage (see Section 4.7) are more likely to correspond to low similarity scores.



Figure 4.4: Similarity values $b_{i,j}$ computed on a sample scene: a) $b_{i,j}$ values between segments in the initial over-segmentation; b) $b_{i,j}$ values between segments at the end of the merging procedure. The color of the boundary between any two touching segments is proportional to their similarity score (white corresponds to high $b_{i,j}$ values and black to low ones).

The couples are placed in a priority queue Q_A sorted based on their similarity values $b_{i,j}$, and the ones with similarity $b_{i,j}$ smaller than a threshold T_{sim} are discarded so that they will not be considered for the merging operations ($T_{sim} = 0.77$ was used

in the results). The aim is to prevent segments with low similarity to be merged, since it is reasonable to expect that they correspond to different objects or portions of the scene.

The algorithm then processes the couple with the highest similarity score. Let S_{i^*} and S_{j^*} be the two segments and let $S_{i^* \cup j^*}$ be their union. A NURBS surface is fitted on each of the two regions i^* and j^* (see Section 4.5) and the fitting error, i.e., the Mean Squared Error (MSE) between the actual and the fitted surface, is computed for both segments providing the values e_{i^*} and e_{j^*} . The fitting error $e_{i^* \cup j^*}$ on segment $S_{i^* \cup j^*}$ is also computed and compared to the weighted average of the errors on S_{i^*} and S_{j^*} :

$$e_{i^*}|S_{i^*}| + e_{j^*}|S_{j^*}| > e_{i^* \cup j^*}(|S_{i^*}| + |S_{j^*}|) \quad (4.8)$$

If Equation (4.8) is satisfied, i.e., the fitting error is reduced, then the two segments are joined, otherwise the union of the two segments is discarded. If the joining of S_{i^*} and S_{j^*} is performed, all the couples including any of the two segments are deleted from Q_A . The priority queue is then updated by considering $S_{i^* \cup j^*}$ adjacent to all segments that were previously adjacent to S_{i^*} or to S_{j^*} and by adding the corresponding couples if their similarity score is greater than or equal to T_{sim} . In order to compute the similarity of the new couples, the descriptor vector $\mathbf{s}_{i^* \cup j^*}$ associated to $S_{i^* \cup j^*}$ is first calculated with Equation (4.6), then Equation (4.7) is used.

The next couple to be processed is then removed from the front of the queue and the algorithm iterates until no more couples are present in the queue.

After getting the final segmentation, a semantic label is also computed for each segment by checking the descriptors of all the pixels in the segment (computed in Section 4.6) and assigning the most common class to the segment. Notice that average and max pooling have also been considered for this task, but they did not lead to better performances than this simple strategy.

Algorithm 1 summarizes the whole merging procedure, while some examples of intermediate steps are shown in Figure 4.5 and in the videos available as additional material.

4.8 Experimental Results

The proposed approach has been tested on the NYU Depth Dataset V2 (NYUDv2) [94]. The NYUDv2 dataset is made of 1449 indoor scenes acquired with a first generation Kinect. For each scene a color view and a depth map are provided. The updated ground truth labels from [34] are used, since the original ones have missing

Algorithm 1 The merge algorithm.

```

 $Q_A \leftarrow$  Priority queue, sort by similarity value
for each couple of adjacent segments  $A_{i,j} = \{S_i, S_j\}$  do
  if  $b_{i,j} \geq T_{sim}$  then
     $Q_A.push(A_{i,j})$ 
  end if
end for
while  $Q_A \neq \emptyset$  do
   $A_{i^*,j^*} \leftarrow Q_A.pop()$ 
  Compute fitting error on merged segment  $S_{i^* \cup j^*}$ 
  if Equation (4.8) is satisfied then
    Remove all  $A_{i,j}$  such that  $i = i^* \vee j = j^*$  from  $Q_A$ 
    for each  $S_k$  adjacent to  $S_{i^* \cup j^*}$  do
      if  $b_{i^* \cup j^*, k} \geq T_{sim}$  then
         $Q_A.push(A_{i^* \cup j^*, k})$ 
      end if
    end for
  end if
end while

```

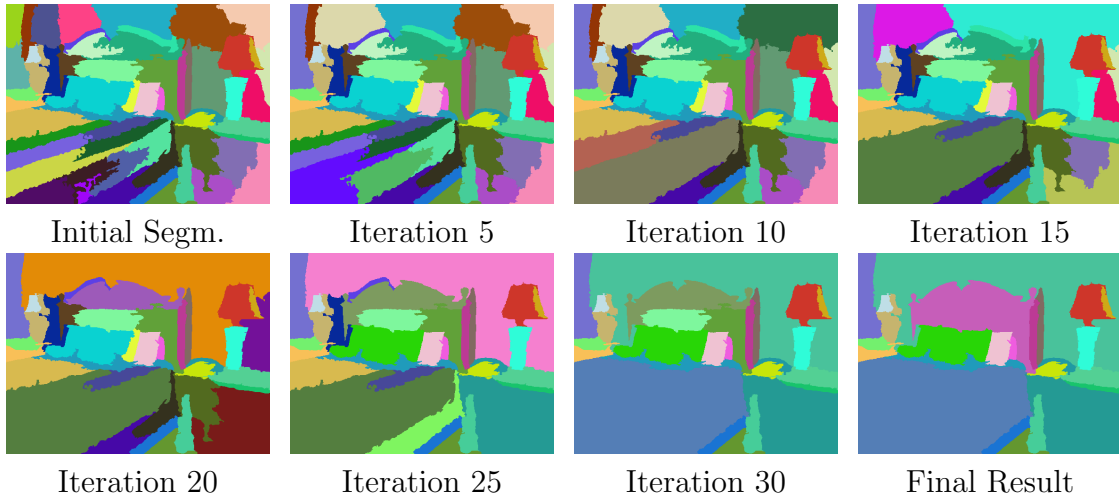


Figure 4.5: Some steps of the merging procedure on the scene of Figure 4.6, row 5. The initial over-segmentation, the output after 5, 10, 15, 20, 25, 30 iterations and the final result (iteration 32) are shown.

areas. The original 894 categories have been clustered into 14 classes as proposed in [9] since this grouping is used for the evaluation of competing approaches. The dataset has been divided in two subsets using the standard train/test separation with 795 and 654 scenes respectively. For the semantic labeling, results on the test set are provided, this being the approach used by all the competing approaches. For segmentation instead most approaches are evaluated on the complete dataset. To get the results in this case two independent evaluations have been done: in the first experiment the standard train/test subdivision has been used as before while in the second the train and test sets have been swapped.

Notice that no expansion of the dataset has been used in the training. Concerning the parameters, $\sigma = 3$ has been used for the normalized cuts algorithm while $T_d = 0.2 m$, $T_c = 10$ and $T_\theta = 4^\circ$ have been set as threshold values for the adjacency computation. Stochastic gradient descent and a quadratic regularization with coefficient 0.001 have been used to optimize the weight of the ConvNet. The learning rate has been set to an initial value of 0.01, then updated with an adaptive decay policy, reducing it of a factor 0.7 after 10 epochs without improvement. The training of the ConvNet network on the NYUDv2 dataset required around 36 hours on a workstation with an Intel i7-970 CPU at 3.2 Ghz and an NVIDIA Tesla K40 GPU.

The proposed method produces a segmentation with semantic labels and it can be exploited both as a segmentation algorithm and as a semantic classification one. This section is split in two parts evaluating the proposed algorithm for the two considered tasks.

4.8.1 Evaluation of the Segmentation Accuracy

Results of comparison of the proposed approach with state-of-the-art methods on the NYUDv2 dataset are presented in Table 4.1 (the results of some competing approaches have been collected from [37] and [74]). The works of [37] and [38], among the compared approaches, are based on clustering and region merging, the MRF scene labeling approach of [84], a variation of [22] that exploits also geometry data, the method of [102] exploiting dynamic programming and the multi-layer clustering strategy of [51]. Other approaches taken into consideration for the comparison are [12] based on normalized cuts, the region merging approach of [74] and the method of [69] that represents the starting point for this work. The method of [74] exploits an iterative merging scheme driven by surface fitting but does not exploit any machine learning clue, so it can be used as a reference to evaluate the impact on the performances due to NURBS surface fitting (i.e., roughly corresponding to the difference between [12] and [74]) and due to the ConvNet descriptors (i.e., the further

improvement from [74] to the proposed work).

The results have been compared with ground truth data using 2 different metrics (see [1] for details). The first is the Variation of Information (VoI) and the second the Rand Index (RI). The mean VoI score of the proposed approach is 1.92. This is the best value among all the considered approaches and the gap is significant. The only method getting close to the proposed one is [69] (i.e., the previous version of this work based on the same segmentation algorithm with a simpler semantic classification stage). The mean score according to the RI metric is 0.91. This value is better than most compared approaches, i.e., [22], [102], [12], [74], [37] and [84], and is the same of the two best competitors, i.e., [38] and [69]. Another advantage is that the method proposed here does not rely on the assumption of planar surfaces (NURBS can handle complex shapes), while many competitors (e.g., [37], [38] and [102]) exploit this assumption thus getting accurate results on the NYUDv2 dataset (that has many planar surfaces), but reducing their generalization capabilities on scenes with non-planar surfaces.

Table 4.1: Average VoI and RI values on the NYUDv2 dataset (1449 scenes). The Table shows the data for some state-of-the-art methods from the literature and for the proposed method. Note that lower values are better for VoI while higher ones are better for RI.

<i>Method</i>	<i>VoI</i>	<i>RI</i>
Hasnat et al. (2014) [37]	2.29	0.90
Hasnat et al. (2016) [38]	2.20	0.91
Ren et al. [84]	2.35	0.90
Felzenszwalb et al. [22]	2.32	0.81
Taylor et al. [102]	3.15	0.85
Khan et al. [51]	2.42	0.87
Dal Mutto et al. [12]	3.09	0.84
Pagnutti et al. [74]	2.23	0.88
Minto et al. [69]	1.93	0.91
Proposed method	1.92	0.91

Some visual examples of segmentations performed with the proposed approach are displayed in Figure 4.6. The sequence of merging steps for a couple of sample scenes is shown in the videos available at http://lttm.dei.unipd.it/paper_data/iet_semantic. The images show that the algorithm is able to properly segment different challenging scenes. The background and the larger surfaces are divided in

several segments in the initial over-segmentation but they are properly merged by the iterative algorithm since the ConvNet descriptors are a very useful clue in order to detect if segments are part of the same object or region. On the other side the algorithm is able to correctly recognize and keep separate most of the scene objects. It is also possible to observe that the objects' edges are precise and there are no small segments close to edges as in some competing methods. However there are a few minor mistakes specially on small objects.

4.8.2 Evaluation of the Classification Accuracy

The proposed approach provides also a semantic label for each segment. In order to evaluate the accuracy of this labeling, it has been compared with some competing approaches on the NYUDv2 test set. The compared state-of-the-art approaches are the methods of [10] that uses a multi-scale ConvNet, of [43] that uses a hierarchy of super-pixels to train a random forest classifier, of [110] that uses deep learning to extract super-pixels features, of [111] exploiting two different ConvNets, of [42] using Random Forest and CRFs and finally [19] using a multi-scale deep learning architecture.

Table 4.2 reports the results: two different metrics have been considered, the per-pixel accuracy, counting the percentage of correctly classified pixels and the average class accuracy, obtained by computing the percentage of correctly classified pixels for each class independently and averaging the values. Notice that the second number is smaller since classes with a low number of samples are typically harder to recognize.

The proposed deep learning architecture achieves a mean pixel accuracy of 64.4% on the test set. By taking the segmentation output of Subsection 4.8.1 and assigning a single label to each segment as described in Section 4.7 it is possible to refine the labeling and increase the accuracy to 67.2%. This is a very good result outperforming all the compared approaches except [19]. Notice that [19] achieves very high performances by exploiting a much more complex deep learning architecture. In any case the method proposed here is the one that gets closer to it, while even the very recent methods of [43] and [111] have lower performances than ours.

The results are also confirmed by the average class accuracy. The ConvNet output accuracy is of 51.7%, a remarkable result outperforming all compared approaches except [111] and [19]. By refining it with the segmentation the accuracy increases to 54.4%, outperforming also [111]. Table 4.3 reports also the accuracy for each class, notice how it is very high on several classes and quite low only for a few classes. In particular the accuracy is lower on classes without a well defined geometric structure, like the *Objects* class, that includes many different things inside or the

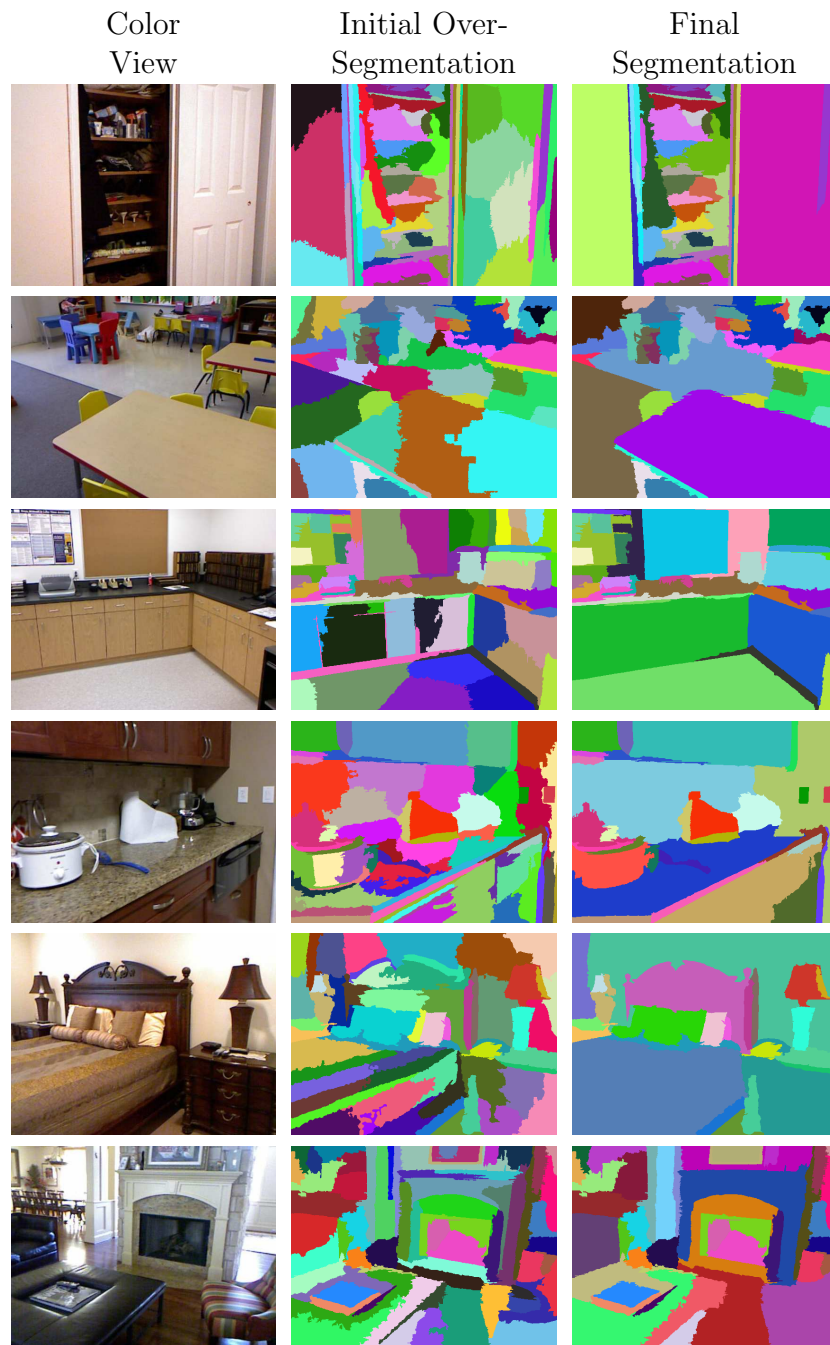


Figure 4.6: Initial over-segmentation and output of the proposed approach for some example scenes. The displayed scenes are the number 72, 330, 450, 846, 1110 and 1313 of the NYUDv2 dataset.

Picture/wall deco class, that is associated to a quite flat and not very descriptive geometry. Another issue is that the dataset is not balanced and there are uncommon classes for which a limited amount of training data is available, e.g., the *TV/monitor* class, that accounts for just 1% of the samples got the worst score in the results. On the other side the *Floor* and *Wall* classes have a very regular structure are detected with an high accuracy. Table 4.2 also report the improvement obtained by refining the output with the segmentation, notice how it improves for all classes except a small loss on the *Picture/wall deco* and *Object* classes. This is due to the fact that the segmentation improves the boundary accuracy and removes isolated detections typically due to noise (but they can seldom correspond to small objects).

A visual evaluation of the results on some sample scenes is shown in Figure 4.7, notice how the classification is accurate even in challenging situations (e.g., closed windows) and how the refinement with segmentation largely improves the edges accuracy. However a few errors are present, e.g., beds exchanged with sofas that have a similar visual appearance.

Different approaches have been tried than simply selecting the most common label for the segmentation-based refinement, however the average pooling scheme lead to the same pixel accuracy of 67.2% with just small differences on the accuracy of the single classes, while the max-pooling scheme lead to less satisfactory performances with a 65.9% accuracy.

The implementation of the approach has not been optimized, currently the processing of a scene requires on average less than 2 minutes. Furthermore most computation time is spent on the initial over-segmentation (87s) that could be replaced with a simpler superpixel segmentation scheme.

Table 4.2: Average pixel and class accuracies on the test set of the NYUDv2 dataset (654 scenes) for some state-of-the-art methods from the literature and for the proposed method.

<i>Approach</i>	<i>Pixel Accuracy</i>	<i>Class Accuracy</i>
Coupric et al. [10]	52.4%	36.2%
Hickson et al. [43]	53.0%	47.6%
A. Wang et al. [110]	46.3%	42.2%
J. Wang et al. [111]	54.8%	52.7%
A. Hermans et al. [42]	54.2%	48.0%
D. Eigen et al. [19]	75.4%	66.9%
Proposed method (ConvNet output)	64.4%	51.7%
Proposed method (with segmentation)	67.2%	54.4%

Table 4.3: Average accuracy for each of the 13 classes on the test set of the NYUDv2 dataset for the proposed approach (the *unknown* class has not been considered consistently with the evaluation of all the compared approaches).

<i>Class</i>	<i>Accuracy (ConvNet)</i>	<i>Accuracy (with segmentation)</i>	<i>Accuracy Improvement</i>
Bed	58.0%	64.1%	6,1%
Objects	43.2%	41.8%	-1,4%
Chair	35.4%	38.4%	3,0%
Furniture	64.7%	70.2%	5,5%
Ceiling	62.8%	64.2%	1,4%
Floor	92.2%	93.7%	1,5%
Picture / wall deco	30.5%	26.8%	-3,7%
Sofa	55.8%	66.5%	10,7%
Table	42.0%	46.0%	4,0%
Wall	83.7%	86.3%	2,6%
Window	53.9%	55.8%	1,9%
Books	23.8%	24.0%	0,2%
Monitor / TV	26.2%	29.1%	2,9%
Average	51.7%	54.4%	2,7%

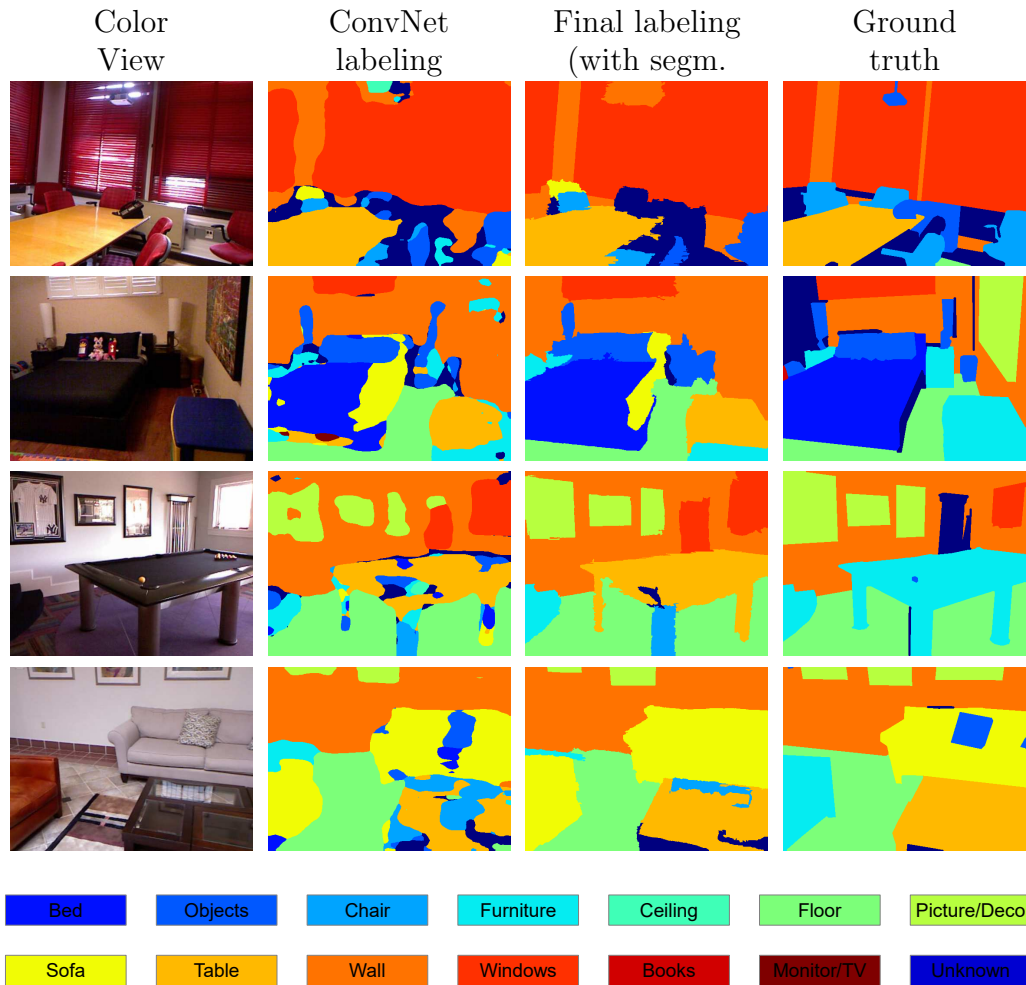


Figure 4.7: Semantic labeling of some sample scenes from the NYUDv2 dataset. The figure shows the color images, the labeling from the Convolutional Network, the refined labeling exploiting segmentation data and the ground truth for scenes 39, 280, 433 and 462.

Chapter 5

3D Shape Recognition

5.1 Introduction

The recent introduction of consumer depth cameras has made 3D data acquisition easier and widely increased the interest in methods for the automatic classification and recognition of 3D shapes. This has been a long term research task, however algorithms dealing with this problem have achieved a completely satisfactory performance only recently, specially thanks to the introduction of deep learning techniques. Differently from standard images, that can be straightforward sent to Convolutional Networks (ConvNets), the processing of 3D point clouds with deep learning techniques requires first of all to represent the data into a form that is suitable for the deep learning algorithms. In this chapter a novel method [70] is presented for the classification of 3D shapes based on the idea of representing the data with multiple 2D structures and then exploiting a multi-branch ConvNet. Three different representations are proposed. The first, derived from the approach presented in [121], is given by a set of different depth maps obtained by rendering the input shape from six different viewpoints, which is a quite standard approach. The second representation is a novel volumetric descriptor that captures the density, i.e., the amount of filled voxels, along directions parallel to the 3D axes. Finally, parametric NURBS surfaces are also fit on the objects, then the two principal curvatures are calculated at each surface location, obtaining 2D maps that describe the local curvature of the shape.

The three representations are used as input for the neural network in Section 5.4: the ConvNet has 15 branches, each branch analyzing a different data source. Specifically, there are 6 branches for the depth maps, 3 for the volumetric data and finally 6 for the curvature data. Each branch contains 4 (for depth and surface data) or 5 (for the volumetric densities) layers that progressively reduce the resolution

until a single description vector is obtained for each of them. In order to reduce the complexity weights have been shared by dividing the depth and curvature branches in two groups, one containing the four side views and one for the top and bottom views. Finally, the classification outputs are concatenated into a single vector which is fed to a final linear layer that produces the shape classification.

5.2 Related Works

The retrieval and classification of 3D shapes is a long term research field. Many different schemes based on global representations and local shape descriptors have been proposed in the past. For an overview of the field see review papers like [101, 32, 57]. As for many other classification tasks, the introduction of deep learning approaches has allowed large improvements and completely changed the way of dealing with this problem. Several different deep learning techniques and in particular ConvNets have been proposed. The fundamental issue with these methods is that 3D representations do not lay on a regular structure as 2D images, making it necessary to convert the data into a representation suitable for the network structure or to adapt the network model.

A first family of approaches is based on the idea of rendering the 3D model from different viewpoints and then use the obtained silhouettes, images or depth maps as input to a standard convolutional network. The work of [96] exploits a spherical parametrization to represent the mesh in a geometry image containing curvature information that is fed to a ConvNet. The method of [92] exploits the idea of representing the 3D object with a panoramic view and uses an ad-hoc ConvNet structure for this kind of images. In the scheme of [49] pairs of views of the object are used together with a second ConvNet for the selection of the best viewpoints. Another approach exploiting this strategy is [98] that extracts a set of color views from the 3D model and combines the information into a single shape descriptor using a ConvNet architecture. Multiple depth maps rendered from the object have been exploited in [121], which are taken as a starting point for the depth-based component of the proposed method.

A second possibility is to use volumetric representations instead, together with three-dimensional ConvNets applied on the voxel structure. In [114] a Convolutional Deep Belief Network is exploited to represent input shapes as probability distributions on a 3D voxel grid. A highly performing method based on the voxel representation is [6], which exploits a variation of the ResNet architecture.

PointNet is introduced in [25] where density occupancy grids are fed as input to a ConvNet for 3D shape classification. The approach of [66] relies on a 3D ConvNet fed

with volumetric occupancy grids while [113] jointly exploits Volumetric Convolutional Networks and Generative Adversarial Networks (GANs).

A comparison between the volumetric and the multi-view scheme is presented in [81], also proposing various improvements to both approaches.

Finally, some approaches exploit non-standard deep learning architectures in order to deal with unstructured data. The approach of [59] exploits field probing filters to extract the features and optimizes not only the weights of the filters as in standard ConvNets but also their locations. Another scheme of this family is the one of [54], which presents a deep learning architecture suited for the Kd-tree representation of volumetric data. A deep network able to directly process point cloud data has been presented in [80].

5.3 Surface and Volume Representations

The proposed algorithm works in two stages: a pre-processing step that constructs the input data followed by a multi-branch ConvNet that performs the classification. The proposed data representation is described in this section while the ConvNet architecture will be the subject of Section 5.4. Three different data representations are considered:

1. A multi-view representation made of a set of six depth maps extracted from the 3D model.
2. A volumetric representation obtained by measuring the number of filled voxels along directions parallel to the 3D space axes.
3. A surface representation given by the curvatures of NURBS surfaces fitted over the 3D model.

5.3.1 Multi-View Representation

In order to build this representation the bounding box of the input 3D model is first computed. Then the 3D model is rendered from six different viewpoints, each corresponding to one of the six faces of the bounding box. The depth information from the z-buffer is extracted for each of the six views, thus obtaining six different depth maps for each object (see Figure 5.1). The output depths have a resolution of 320×320 pixels which, following experiments, proved to be a reasonable trade-off between the accuracy of the representation and the computational effort required

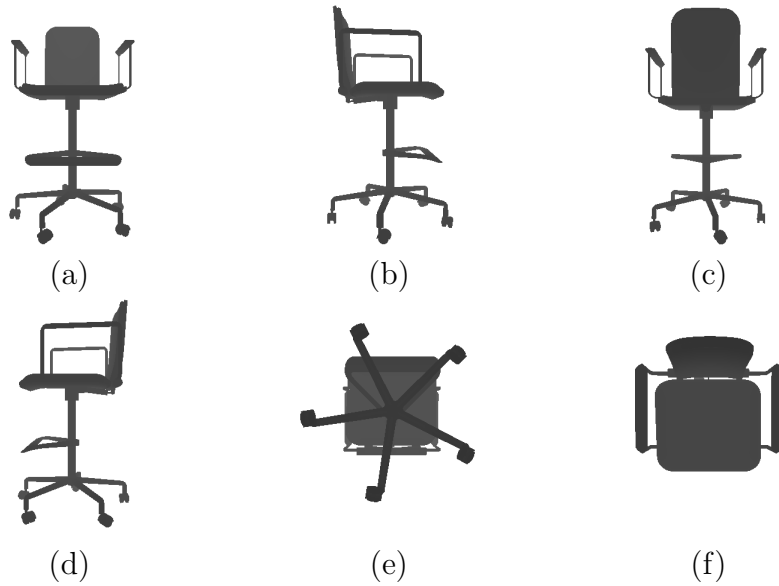


Figure 5.1: Example of the six depth maps used for the analysis of a *chair* 3D model: Four side views, namely front view (a), left view (b), back view (c) and right view (d), and two additional views namely bottom view (e) and top view (f).

to train the ConvNet. The six depth maps represent the input for the proposed classifier. Their usage makes it possible to capture a complete description of the 3D shape without considering a full volumetric structure that would require a larger amount of data due to the higher dimensionality. Furthermore, the depth map conveys a greater information content if compared with the silhouettes of the shape. Notice that, assuming the object is lying on the ground, the six views can be divided into four side views with a similar structure and the bottom and top views. Indeed, for many real world objects it is reasonable to assume they can rotate around the vertical axis while being constrained to lay on the ground. Furthermore, the fact that typically the four side views have a similar content while the top and bottom capture a different representation will be exploited in the construction of the neural network. Finally, local contrast normalization is applied to each input depth map independently.

5.3.2 Volume Representation

Volumetric representations have been exploited in various 3D classification schemes like the ones of [114, 113, 66]. Unfortunately, the performance of approaches exploit-

ing the full volumetric representation is affected by the fact that the 3D structure containing the voxel data uses a considerable amount of memory and requires 3D convolutional filters with a higher number of parameters. The increased dimensionality and requirements are typically compensated by using low resolution and simpler networks, but this also impacts on the performance. In order to exploit the information given by the volumetric data and at the same time preserve the simpler and faster operations of 2D representations a novel data representation has been introduced. The idea is to build a set of three density maps representing the density of filled voxels along the directions corresponding to each of the three axes. More in detail, the X -axis representation is built by quantizing the YZ -plane into 32×32 cells and counting how many filled voxels are encountered by going down along the X -axis from each location (i.e., letting X vary after fixing the value of the Y and Z coordinates). The representation for the Y -axis and Z -axis are built in the same way by swapping the axes (i.e., fix X and Z and let Y vary or fix X and Y and let Z vary). A visual example on a table model is shown in Figure 5.2. Notice how, for example, the Z profile (i.e., the top-to-bottom profile) captures the table surface (low density) and four high density spots corresponding to the four legs of the table. Finally, as for depth information, local contrast normalization has been applied to the data.

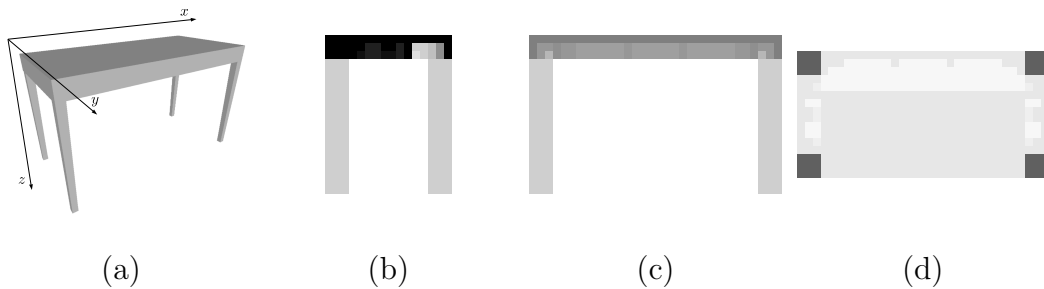


Figure 5.2: Example of a *table* 3D model (a) and corresponding voxel density maps computed along the x -axis (b), y -axis (c) and z -axis (d) respectively.

5.3.3 Surface Representation

The third data representation is based on geometric properties of parametric surfaces that approximate the objects shape. The idea is to consider the six views of the first representation and to obtain a Non-Uniform Rational B-Spline (NURBS) fitting surface for each of them. In order to perform this task the 3D points corresponding

to each depth sample have been approximated with a continuous parametric surface $S(u, v)$, computed by solving an over-determined system of linear equations in the least-squares sense. Notice that the u, v parametric range of the NURBS surface corresponds to the rectangular grid structure of the depth map. The NURBS degrees in the u and v directions have been set to 3, while the weights are all equal to 1, i.e., the fitted surfaces are non-rational (splines). The same surface fitting algorithm presented in [69, 75] is used here, see these publications for more details on this task. Notice how the usage of NURBS surfaces provides a geometric model that is well suitable to describe arbitrary shapes, not only planar ones. After fitting the surfaces their two principal curvatures k_1 and k_2 are determined at each sample location. An example of the resulting information is visible in Figure 5.3, that shows the two curvature maps for a sample object. Such data are locally normalized and then used as the last input for the ConvNet classifier.

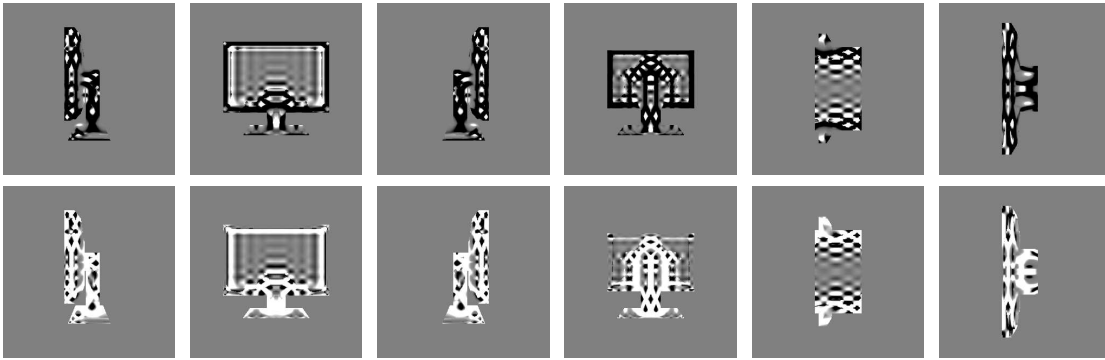


Figure 5.3: Example of curvature maps for a 3D model of the *monitor* class. The first row shows the k_1 curvature maps for the six views, while the second row shows the data relative to k_2 . The data have been scaled for visualization purposes (dark colors correspond to negative values and bright colors to positive ones).

5.4 3D Shape Recognition with ConvNet

The proposed classifier takes in input the three representations and gives in output a semantic label for each scene. For this task an ad-hoc ConvNet architecture with multiple branches has been developed.

The structure of the network for this task is summarized in Figure 5.4. It is made of two main parts, namely a set of branches containing convolutional layers and a linear classification stage combining the information from the different branches. The

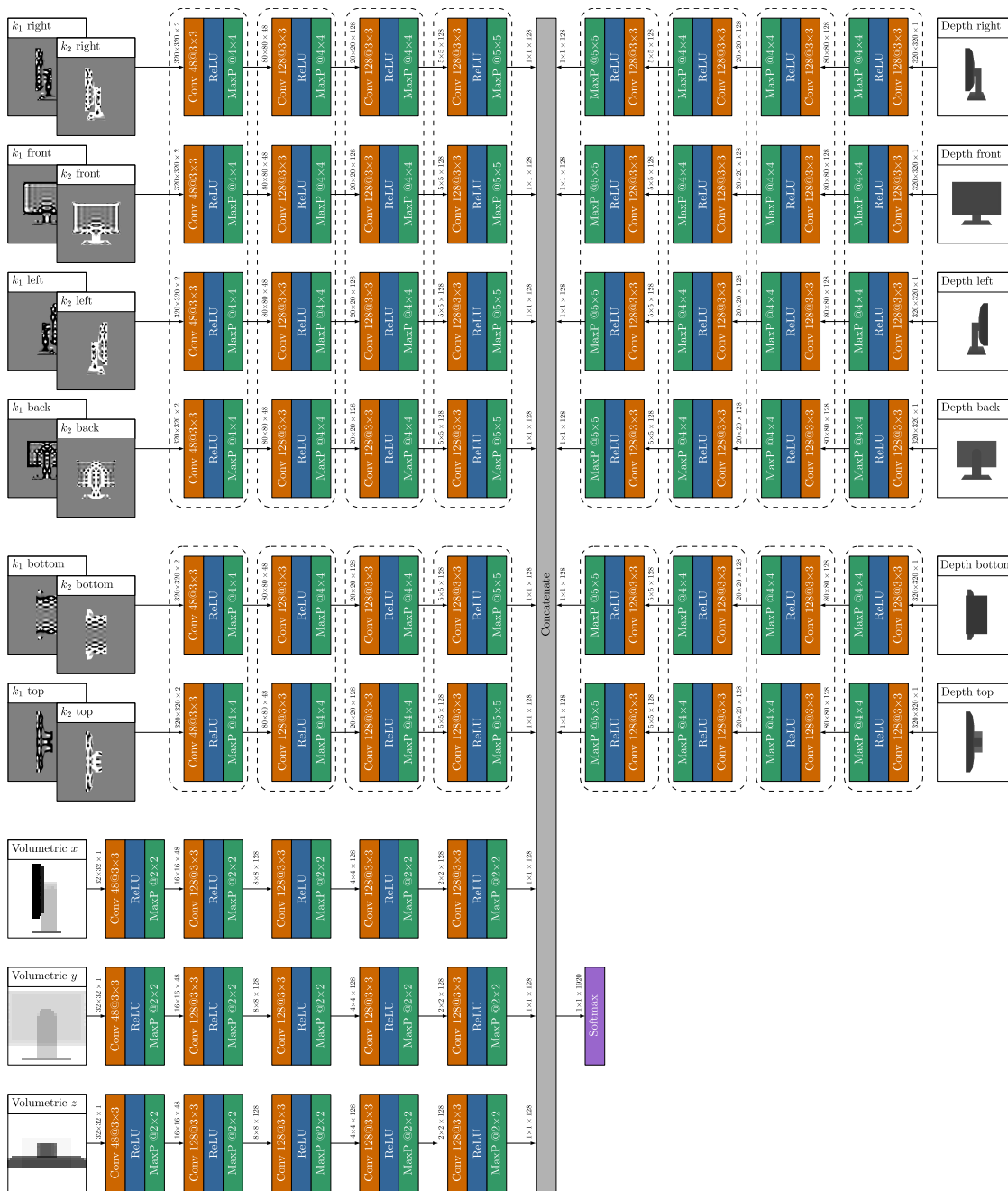


Figure 5.4: Layout of the proposed multi-branch Convolutional Network.

architecture of the network has been designed in order to produce a single reliable classification output for each 3D object and to be suitable for the multi-modal input of Section 5.3. Its structure stems from the semantic segmentation architectures presented in [21, 9, 69], but greatly differs from them due to the different task and the particular nature of the exploited data. In the first part there are 15 different branches, divided in 3 groups (see Figure 5.4). The first group has 6 branches and processes the depth information. Each branch takes in input a single depth map at the resolution of 320×320 pixels and extracts a classification vector for the input by applying a sequence of convolutional layers. More in detail, each branch has 4 convolutional layers (CONV), each followed by a rectified linear unit activation function (RELU) and a max-pooling layer (MAXP). The first layer has 48 filters while the second, third and fourth ones have 128 filters, all being 3×3 pixels wide. The max-pooling stages subsample the data by a factor of 4 in each dimension in the first three stages and of 5 in the last one. The rationale behind these values is to progressively reduce the resolution of the data until a single descriptor vector is obtained for each depth map. Notice that, by using this approach only the first layer works at the full resolution, the latter being significantly reduced in the next layers up to the point where a single classification hypothesis is obtained for the whole depth map in the last layer. This scheme allows to limit the computational resources required for the training since in the inner layers the resolution is strongly reduced. The final output is a 128-elements descriptor for each depth map. Concerning the weights of the convolutional filters there are two approaches commonly used. The first is to have independent weights for each branch of the network. This allows to better adapt the network to the various views, however it leads to a large amount of parameters thus increasing the computational complexity and the risk of over-fitting. Furthermore, this makes also the approach more dependent on the pose of the model since, changing the pose, data can move from one view to another. Other approaches share the weights across the various branches [21, 9], thus reducing the complexity but also the discrimination capabilities of the network. A key observation is that the captured data are typically similar in the four side views but different for the top and bottom ones. Thus a hybrid solution between the two approaches has been used, with a shared set of weights for the four side views and a different set for the top and bottom ones. This proved to be a good trade-off between the two solutions, providing a good accuracy with a reasonable training time and a partial invariance at least to the rotation along the vertical axis. Notice that the approach assumes that the objects are laying on the ground in order to distinguish between the side and top or bottom views, however this is a reasonable assumption for most real world objects. The branches in the second set deal with volumetric data. In this case

Table 5.1: Summary of the properties of the various layers in the ConvNet.

Input	Branches		Layers				
			L1	L2	L3	L4	L5
Depth	6	Filters	48	128	128	128	-
		Conv	3×3	3×3	3×3	3×3	-
		Pooling	4×4	4×4	4×4	5×5	-
NURBS	6	Filters	48	128	128	128	-
		Conv	3×3	3×3	3×3	3×3	-
		Pooling	4×4	4×4	4×4	5×5	-
Volumetric	3	Filters	48	128	128	128	128
		Conv	3×3	3×3	3×3	3×3	3×3
		Pooling	2×2	2×2	2×2	2×2	2×2

there are 3 branches, associated to the X -axis, Y -axis and Z -axis respectively. The input data have a lower resolution, namely 32×32 pixels (see Section 5.3.2 for the rationale behind this choice). Each branch has 5 convolutional layers, each one with a RELU activation and a max-pooling stage. There are 48 convolutional filters in the first layer and 128 in the other ones. The filters are still 3×3 , however this time the max-pooling stages subsample the data by a factor of 2 due to the lower starting resolution. In this case weights are independent for each branch since the three profiles capture different data. Notice also that, given the low resolution, sharing the weights among the three branches would not bring any substantial reduction in the training effort to be performed. Finally, the third set of branches is devoted to the surface fitting data. There is a set of coefficients contained in a 2-channel 320×320 pixels map for each 3D view, the structure being very similar to the one of first group. In this case, there are two channels corresponding to the two principal curvatures instead of one only. Aside from this, the network architecture is exactly the same as the one of the first group.

The 128-elements descriptor vectors produced by each one of the 15 channels are then concatenated in a $15 \times 128 = 1920$ elements vector and fed to a final softmax classifier with weight matrix of size $1920 \times n_c$ and no bias, where n_c is the considered number of classes. Parameter n_c is set equal to 10 and 40 depending on the dataset used for experimental results. The network is trained as described in Section 5.5 to produce a labeling of each 3D shape by assigning it one out of the n_c different categories. To this aim, a multi-class cross-entropy loss function is minimized throughout the training process. Epochs have been limited to 100, even

if the optimal solution is typically reached earlier.

5.5 Experimental Results

The performance of the proposed approach has been evaluated on the Princeton *ModelNet* dataset [114], a large-scale dataset containing 3D object models along with their ground truth categories. Results are presented for both the 10-class subset *ModelNet10* and the larger 40-class subset *ModelNet40*. In particular, ConvNet described in Section 5.4 has been trained and tested on the two subsets independently. To this aim, standard training and test splits have been adopted as provided along with each subset data. Specifically, results on the *ModelNet10* subset have been obtained by training the network on 3991 samples, leaving aside 908 samples for the test. Similarly, training and testing on *ModelNet40* has been performed using 9843 and 2468 samples respectively.

In both cases the training has been carried out by minimizing a multi-class cross-entropy loss function with the Stochastic Gradient Descent (SGD) algorithm. The Theano framework [103] has been used for the implementation of the optimization algorithm.

Starting from the *ModelNet10* subset, the impact of each one of the three data representations separately if first evaluated. The results reported in Table 5.2 suggest that the depth maps extracted from the 3D model rendering carry the largest information content, achieving an average accuracy of 93.2% when used alone. A remarkable accuracy can also be obtained by feeding the ConvNet with volumetric profiles only, correctly predicting 91.2% of the models in the test set. Despite this value is lower than the one derived using depth data, it is still noteworthy that it has been obtained with a low resolution data representation (32×32 pixels). Even if volumetric profiles size equal to just 1% of depth data, results demonstrate that is still possible to correctly classify most of the 3D models using this representation alone. Finally, the accuracy that can be obtained using only surface (NURBS) curvature data is 90.9%, lower than the other two descriptors but still noticeable, proving also the effectiveness of this representation. By combining all the three representations together, an average accuracy of 93.6% has been achieved on the test set, higher than the one obtained by taking each representation separately.

An in-depth analysis of the performance is shown in Table 5.3, which contains the confusion matrix of the proposed approach on the *ModelNet10* dataset and in Table 5.4, which reports the average accuracies obtained on each separate class.

Notice how the proposed method is able to achieve a very high accuracy on most classes. Some of them are almost perfectly recognized, e.g. the *bed*, *chair*, *monitor*

and *toilet* classes. On the other side, some critical situations also exist such as the confusion between the *night stand* and *dresser* classes, an expected issue since these two classes have similar shapes and the disambiguation is difficult in some samples even for a human observer. Another challenging recognition scenario is given when it comes to distinguish between the *table* and *desk* classes. In several instances these classes share a typical structure made by a flat surface supported by the legs. Nonetheless, most samples in these classes are correctly recognized even if some errors are present.

Table 5.2: Average accuracies on the *ModelNet10* and *ModelNet40* datasets for the proposed method when using the three different data representations taken separately as well as their combination.

<i>Approach</i>	<i>ModelNet10</i>	<i>ModelNet40</i>
Depth maps	93.2%	88.0%
Volumetric	92.2%	86.9%
NURBS	90.9%	85.2%
Combined	93.6%	89.3%

The performance of the proposed approach has also been compared with some recent state-of-the-art approaches on the *ModelNet10* dataset (all the compared approaches can be dated back to the last two years). The comparison is reported in Table 5.5: the average accuracy of 93.6% obtained by combining all three data representations is higher than most of previous works. Specifically, only [6] and [54] outperform the proposed approach, the second one by a limited performance gap.

The approach has been evaluated also on the larger *ModelNet40* subset that, as expected, proved to be more challenging due to the larger number of classes and higher variety of models. The results on this subset are reported in the last column of Table 5.5. The depth information alone allows to obtain an accuracy of 88.0%, a lower value than the one achieved on *ModelNet10*. Yet the loss (about 5%) is quite limited, especially if considering that the model is expected to discriminate between 4 times more categories. Accuracy undergoes a similar drop also when using volumetric data, being able to correctly recognize 86.9% of the models compared to 92.2% on the *ModelNet10* subset. As for NURBS data, the test gave an accuracy of 85.2%, consistently with the results obtained with depth and volumetric data. Notice how the relative ranking of the three representations is the same as for *ModelNet10*, being depth the most accurate, followed by volumetric data and finally NURBS curvatures. Finally, the combined use of the three representations led to an accuracy

Table 5.3: Confusion matrix for the proposed approach on the *ModelNet10* dataset. Values are given in percentage.

	bath tub	bed	chair	desk	dresser	monitor	night st.	sofa	table	toilet
bath tub	92	8	0	0	0	0	0	0	0	0
bed	0	100	0	0	0	0	0	0	0	0
chair	0	0	100	0	0	0	0	0	0	0
desk	0	1	0	86	0	0	7	1	5	0
dresser	0	0	0	0	85	1	14	0	0	0
monitor	0	0	0	0	1	99	0	0	0	0
night st.	0	0	0	0	13	0	80	0	7	0
sofa	0	0	0	1	0	0	2	97	0	0
table	0	0	0	7	0	0	0	0	93	0
toilet	0	0	1	0	0	0	0	0	0	99

Table 5.4: Accuracy of the proposed approach on the various classes of the *ModelNet10* dataset. The number of samples belonging to each class is also reported.

Class	Acc.	Samples	Class	Acc.	Samples
bath tub	92%	50	monitor	99%	100
bed	100%	100	night st.	80%	86
chair	100%	100	sofa	97%	100
desk	86%	86	table	93%	100
dresser	85%	86	toilet	99%	100

Table 5.5: Average accuracies on the *ModelNet10* and *ModelNet40* datasets for some state-of-the-art methods from the literature and for the proposed method.

<i>Approach</i>	<i>ModelNet10</i>	<i>ModelNet40</i>
3DShapeNets [114]	83.5%	77.0%
DeepPano [92]	85.5%	77.6%
VoxNet [66]	92.0%	83.0%
Klokov and Lempitsky [54]	94.0%	91.8%
Zanuttigh and Minto [121]	91.5%	87.8%
LightNet [125]	93.39%	86.90%
Xu and Todorovic [115]	88.00%	81.26%
Pairwise [49]	92.8%	90.7%
3D-GAN [113]	91.0%	83.3%
VRN Ensemble [6]	97.14%	95.54%
Geometry Image [96]	88.4%	83.9%
GIFT [2]	92.4%	83.1%
ECC [95]	90.0%	83.2%
FusionNet [40]	93.11%	90.8%
PANORAMA-NN [91]	91.1%	90.7%
<i>Proposed Method</i>	93.6%	89.3%

of 89.3%. Notice that, in this case, the gap with respect to the various representations taken separately is larger, revealing the effectiveness of the combined use of multiple representations particularly when dealing with more challenging tasks. The drop with respect to *ModelNet10* when all representations are used is just around 4%.

The average accuracy for each single class is shown in Table 5.6, while some examples of correctly and wrongly classified objects are shown in Figure 5.5 and 5.6 respectively. The accuracy is high on most classes except for a few of them, e.g., the *flower pot* and *radio* classes. These correspond to classes with a limited amount of training samples and a large variability between the samples for which the algorithm is not able to properly learn the structure. Inter-class similarities are also more common given the larger number of fine-grain classes present in the subset. In general, classes sharing a similar appearance are more challenging to disambiguate, leading the model to confuse e.g. *table* instances with *desk* instances. Similarly, *flower pot* instances are often misclassified as *plant* or *vase* while a number of *cup* instances have been wrongly assigned to the *vase* category (some examples in these classes are shown in Figure 5.6).

The comparison with competing approaches on the *ModelNet40* dataset is also reported in Table 5.5. In this case the proposed approach ranks 6th out of 15 compared methods, a very good performance even if the relative ranking is slightly lower than in the previous case, specially considering that all the compared approaches are very recent and from top conferences and journals.

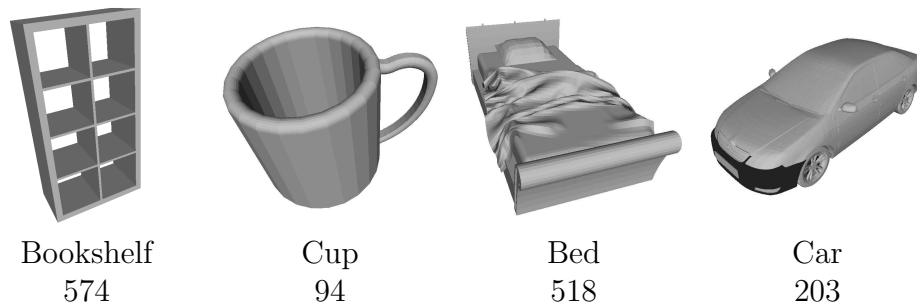


Figure 5.5: Examples of 3D models from the *ModelNet40* dataset correctly recognized by the proposed approach.

Finally concerning the training time, it is about 22 hours for the *ModelNet10* dataset and 51 hours for the larger *ModelNet40* dataset. The tests have been performed on a system equipped with an 3.60 GHz Intel i7-4790 CPU and an NVIDIA TitanX (Pascal) GPU and refer to the complete version of the approach with all the three representations.

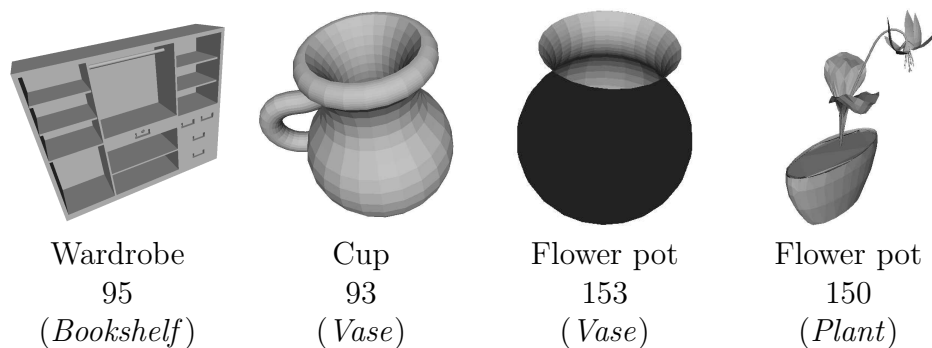


Figure 5.6: Examples of 3D models from the *ModelNet40* dataset wrongly recognized by the proposed approach. The predicted categories are reported in parenthesis.

Table 5.6: Accuracy of the proposed approach on the various classes of the *ModelNet40* dataset. The number of samples belonging to each class is also reported.

Class	Acc.	Samples	Class	Acc.	Samples
airplane	100%	100	laptop	100%	20
bathtub	90%	50	mantel	94%	100
bed	97%	100	monitor	99%	100
bench	80%	20	night st.	76%	86
bookshelf	97%	100	person	100%	20
bottle	96%	100	piano	89%	100
bowl	100%	20	plant	91%	100
car	97%	100	radio	60%	20
chair	96%	100	r. hood	94%	100
cone	90%	20	sink	70%	20
cup	65%	20	sofa	96%	100
curtain	80%	20	stairs	75%	20
desk	84%	86	stool	90%	20
door	95%	20	table	77%	100
dresser	80%	86	tent	90%	20
flower pot	15%	20	toilet	97%	100
glass box	96%	100	tv st.	84%	100
guitar	93%	100	vase	74%	100
keyboard	100%	20	wardrobe	80%	20
lamp	75%	20	xbox	70%	20

Chapter 6

Conclusions

Three problems have been faced, namely ToF-stereo data fusion, semantic segmentation of RGB-D images and 3D shape classification.

In all three cases, a ConvNet has been used to directly tackle the problem or one of its sub-part. In Chapter 3 a deep ConvNet trained on a synthetic dataset is employed to estimate the reliability of ToF and stereo data obtaining reliable confidence maps identifying the most critical acquisition issues of both sub-systems. The solution to the fusion problem is then demanded to a different algorithm, the LC algorithm [64], which fuses the ToF and stereo disparities exploiting their estimated confidences. In Chapter 4 a multi-scale ConvNet is used to extract suitable descriptors from color and geometry clues. Once such descriptors have been extracted, a similarity measure is computed from them and then exploited together with surface fitting errors to drive the merging operations iteratively applied to go from an initial over-segmentation to the final result. Finally, the 3D shape recognition problem in Chapter 5 is fully addressed (except for a pre-processing stage) by a multi-branch ConvNet that takes as input suitable surface and volume representations of a given 3D model and outputs its predicted class.

3D data have also been exploited throughout all three approaches, i.e. depth and volumetric data has been used for the classification task, while color and depth data have been jointly exploited in the ToF-stereo fusion and RGB-D semantic segmentation.

Results suggest that, for the tasks being considered, both the use of deep ConvNet models as well as the joint exploitation of geometry clues together with color ones represent a successful strategy. In particular, the approaches proposed in Chapter 4 and 5 for semantic segmentation and 3D shape recognition respectively proved to achieve a state-of-the-art performance. As for ToF-stereo fusion, the gap with

other state-of-the-art approaches is still small. In this specific case however, tests made by driving the fusion with ground truth confidence maps revealed a substantial improvement, suggesting that the use of a more powerful ConvNet and likely more accurate confidence estimates would led to better results.

Bibliography

- [1] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2011.
- [2] Song Bai, Xiang Bai, Zhichao Zhou, Zhaoxiang Zhang, and Longin Jan Latecki. Gift: A real-time and scalable 3d shape search engine. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5023–5032, 2016.
- [3] Dan Banica and Cristian Sminchisescu. Second-order constrained parametric proposals and sequential search-based structured prediction for semantic segmentation in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3517–3526, 2015.
- [4] Blender website. <https://www.blender.org/>, Accessed July 31st, 2017.
- [5] Blend swap website. <https://www.blendswap.com/>, Accessed July 31st, 2017.
- [6] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236*, 2016.
- [7] Joao Carreira and Cristian Sminchisescu. Cpmc: Automatic object segmentation using constrained parametric min-cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1312–1328, 2012.
- [8] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

- [9] Camille Couprie, Clément Farabet, Laurent Najman, and Yann LeCun. Indoor semantic segmentation using depth information. *arXiv preprint arXiv:1301.3572*, 2013.
- [10] Camille Couprie, Clément Farabet, Laurent Najman, and Yann Lecun. Convolutional nets and watershed cuts for real-time semantic labeling of rgbd videos. *Journal of Machine Learning Research*, 15(1):3489–3511, 2014.
- [11] Carlo Dal Mutto, Pietro Zanuttigh, and Guido M Cortelazzo. A probabilistic approach to tof and stereo data fusion. *3DPVT, Paris, France*, 2, 2010.
- [12] Carlo Dal Mutto, Pietro Zanuttigh, and Guido M Cortelazzo. Fusion of geometry and color information for scene segmentation. *IEEE Journal of Selected Topics in Signal Processing*, 6(5):505–521, 2012.
- [13] Carlo Dal Mutto, Pietro Zanuttigh, Guido M Cortelazzo, and Stefano Mattocchia. Scene segmentation assisted by stereo vision. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*, pages 57–64. IEEE, 2011.
- [14] Carlo Dal Mutto, Pietro Zanuttigh, and Guido Maria Cortelazzo. Probabilistic tof and stereo data fusion based on mixed pixels measurement models. *IEEE transactions on pattern analysis and machine intelligence*, 37(11):2260–2272, 2015.
- [15] Carlo Dal Mutto, Pietro Zanuttigh, Stefano Mattocchia, and Guido Cortelazzo. Locally consistent tof and stereo data fusion. In *Computer Vision–ECCV 2012. Workshops and Demonstrations*, pages 598–607. Springer, 2012.
- [16] Zhuo Deng, Sinisa Todorovic, and Longin Jan Latecki. Semantic segmentation of rgbd images with mutex constraints. In *Proceedings of the IEEE international conference on computer vision*, pages 1733–1741, 2015.
- [17] James Diebel and Sebastian Thrun. An application of markov random fields to range sensing. In *Advances in neural information processing systems*, pages 291–298, 2006.
- [18] Jennifer Dolson, Jongmin Baek, Christian Plagemann, and Sebastian Thrun. Upsampling range data in dynamic environments. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1141–1148. IEEE, 2010.

- [19] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2650–2658, 2015.
- [20] Georgios D Evangelidis, Miles Hansard, and Radu Horaud. Fusion of range and stereo data for high-resolution scene-modeling. *IEEE transactions on pattern analysis and machine intelligence*, 37(11):2178–2192, 2015.
- [21] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013.
- [22] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59(2):167–181, 2004.
- [23] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the nystrom method. *IEEE transactions on pattern analysis and machine intelligence*, 26(2):214–225, 2004.
- [24] A Frick, F Kellner, B Bartczak, and R Koch. Generation of 3d-tv ldv-content with time-of-flight camera. In *3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video, 2009*, pages 1–4. IEEE, 2009.
- [25] Alberto Garcia-Garcia, Francisco Gomez-Donoso, Jose Garcia-Rodriguez, Sergio Orts-Escolano, Miguel Cazorla, and J Azorin-Lopez. Pointnet: A 3d convolutional neural network for real-time object class recognition. In *Neural Networks (IJCNN), 2016 International Joint Conference on*, pages 1578–1584. IEEE, 2016.
- [26] Valeria Garro, Carlo Dal Mutto, Pietro Zanuttigh, and Guido M Cortelazzo. A novel interpolation scheme for range data with side information. In *Visual Media Production, 2009. CVMP'09. Conference for*, pages 52–60. IEEE, 2009.
- [27] Agresti Gianluca, Minto Ludovico, Giulio Marin, and Pietro Zanuttigh. Deep learning for confidence information in stereo and tof data fusion. In *Computer Vision–ECCV 2017 Workshops*. Springer, 2017.
- [28] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

- [29] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.
- [30] Marcin Grzegorzek, Christian Theobalt, Reinhard Koch, and Andreas Kolb. *Time-of-Flight and Depth Imaging. Sensors, Algorithms and Applications: Dagstuhl Seminar 2012 and GCPR Workshop on Imaging New Modalities*, volume 8200. Springer, 2013.
- [31] Sigurjon Arni Gudmundsson, Henrik Aanaes, and Rasmus Larsen. Fusion of stereo vision and time-of-flight imaging for improved 3d estimation. *International Journal of Intelligent Systems Technologies and Applications*, 5(3-4):425–433, 2008.
- [32] Yulan Guo, Mohammed Bennamoun, Ferdous Sohel, Min Lu, and Jianwei Wan. 3d object recognition in cluttered scenes with local surface features: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2270–2287, 2014.
- [33] Saurabh Gupta, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation. *International Journal of Computer Vision*, 112(2):133–149, 2015.
- [34] Saurabh Gupta, Pablo Arbelaez, and Jitendra Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 564–571, 2013.
- [35] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision*, pages 345–360. Springer, 2014.
- [36] Miles Hansard, Seungkyu Lee, Ouk Choi, and Radu Patrice Horaud. *Time-of-flight cameras: principles, methods and applications*. Springer Science & Business Media, 2012.
- [37] Md Abul Hasnat, Olivier Alata, and Alain Trémeau. Unsupervised rgb-d image segmentation using joint clustering and region merging. *J-STSP*, 6(5):505–521, 2012.

- [38] Md Abul Hasnat, Olivier Alata, and Alain Trémeau. Joint color-spatial-directional clustering and region merging (jcsd-rm) for unsupervised rgb-d image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 38(11):2255–2268, 2016.
- [39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [40] Vishakh Hegde and Reza Zadeh. Fusionnet: 3d object classification using multiple data representations. *arXiv preprint arXiv:1607.05695*, 2016.
- [41] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1106–1112. IEEE, 1997.
- [42] Alexander Hermans, Georgios Floros, and Bastian Leibe. Dense 3d semantic mapping of indoor scenes from rgb-d images. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2631–2638. IEEE, 2014.
- [43] Steven Hickson, Irfan Essa, and Henrik Christensen. Semantic instance labeling leveraging hierarchical segmentation. In *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*, pages 1068–1075. IEEE, 2015.
- [44] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2008.
- [45] Nico Höft, Hannes Schulz, and Sven Behnke. Fast semantic segmentation of rgb-d scenes with gpu-accelerated deep neural networks. In *Joint German/Austrian Conference on Artificial Intelligence (Künstliche Intelligenz)*, pages 80–85. Springer, 2014.
- [46] Stefan Holzer, Radu Bogdan Rusu, M Dixon, Suat Gedikli, and Nassir Navab. Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 2684–2689. IEEE, 2012.

- [47] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. *CoRR*, abs/1704.05526, 2017.
- [48] Xiaoyan Hu and Philippos Mordohai. A quantitative evaluation of confidence measures for stereo vision. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2121–2133, 2012.
- [49] Edward Johns, Stefan Leutenegger, and Andrew J Davison. Pairwise decomposition of image sequences for active multi-view recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3813–3822, 2016.
- [50] Timo Kahlmann and Hilmar Ingensand. Calibration and development for increased accuracy of 3d range imaging cameras. *Journal of Applied Geodesy*, 2(1):1–11, 2008.
- [51] Mahfuzur Rahman Khan, ABM Muhitur Rahman, GM Atiqur Rahaman, and Md Abul Hasnat. Unsupervised rgb-d image segmentation by multi-layer clustering. In *Informatcs, Electronics and Vision (ICIEV), 2016 5th International Conference on*, pages 719–724. IEEE, 2016.
- [52] Salman Hameed Khan, Mohammed Bennamoun, Ferdous Sohel, and Roberto Togneri. Geometry driven semantic labeling of indoor scenes. In *European Conference on Computer Vision*, pages 679–694. Springer, 2014.
- [53] Young Min Kim, Christian Theobalt, James Diebel, Jana Kosecka, Branislav Miscusik, and Sebastian Thrun. Multi-view image and tof sensor fusion for dense 3d reconstruction. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1542–1549. IEEE, 2009.
- [54] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. *arXiv preprint arXiv:1704.01222*, 2017.
- [55] Klaus-Dieter Kuhnert and Martin Stommel. Fusion of stereo-camera and pmd-camera data for real-time suited precise 3d environment reconstruction. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 4780–4785. IEEE, 2006.

- [56] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [57] Bo Li, Yijuan Lu, Chunyuan Li, Afzal Godil, Tobias Schreck, Masaki Aono, Martin Burtscher, Qiang Chen, Nihad Karim Chowdhury, Bin Fang, et al. A comparison of 3d shape retrieval methods based on a large-scale benchmark supporting multimodal queries. *Computer Vision and Image Understanding*, 131:1–27, 2015.
- [58] Ke Li and Jitendra Malik. Amodal instance segmentation. In *European Conference on Computer Vision*, pages 677–693. Springer, 2016.
- [59] Yangyan Li, Sören Pirk, Hao Su, Charles R Qi, and Leonidas J Guibas. Fpnn: Field probing neural networks for 3d data. In *Advances in Neural Information Processing Systems*, pages 307–315, 2016.
- [60] Dahua Lin, Sanja Fidler, and Raquel Urtasun. Holistic scene understanding for 3d object detection with rgb-d cameras. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1417–1424, 2013.
- [61] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.
- [62] Luxrender website. <http://www.luxrender.net>, Accessed July 31st, 2017.
- [63] Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in Neural Information Processing Systems*, pages 1682–1690, 2014.
- [64] Giulio Marin, Pietro Zanuttigh, and Stefano Mattoccia. Reliable fusion of tof and stereo depth driven by confidence measures. In *European Conference on Computer Vision*, pages 386–401. Springer, 2016.
- [65] Stefano Mattoccia. A locally global approach to stereo correspondence. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1763–1770. IEEE, 2009.
- [66] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems*

- (IROS), *2015 IEEE/RSJ International Conference on*, pages 922–928. IEEE, 2015.
- [67] Stephan Meister, Rahul Nair, and Daniel Kondermann. Simulation of Time-of-Flight Sensors using Global Illumination. In Michael Bronstein, Jean Favre, and Kai Hormann, editors, *Vision, Modeling and Visualization*. The Eurographics Association, 2013.
- [68] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, USA, 1969.
- [69] Ludovico Minto, Giampaolo Pagnutti, and Pietro Zanuttigh. Scene segmentation driven by deep learning and surface fitting. In *Computer Vision–ECCV 2016 Workshops*, pages 118–132. Springer, 2016.
- [70] Ludovico Minto, Pietro Zanuttigh, and Giampaolo Pagnutti. Deep learning for 3d shape classification based on volumetric density and surface approximation clues. In *VISAPP 2018*, 2018.
- [71] Rahul Nair, Frank Lenzen, Stephan Meister, Henrik Schäfer, Christoph Garbe, and Daniel Kondermann. High accuracy tof and stereo sensor fusion at interactive rates. In *Computer Vision–ECCV 2012. Workshops and Demonstrations*, pages 1–11. Springer, 2012.
- [72] Giampaolo Pagnutti, Ludovico Minto, and Pietro Zanuttigh. Segmentation and semantic labelling of rgb-d data with convolutional neural networks and surface fitting. *IET Computer Vision*, 2017.
- [73] Giampaolo Pagnutti and Pietro Zanuttigh. Scene segmentation from depth and color data driven by surface fitting. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 4407–4411. IEEE, 2014.
- [74] Giampaolo Pagnutti and Pietro Zanuttigh. Joint color and depth segmentation based on region merging and surface fitting. In *VISIGRAPP (4: VISAPP)*, pages 93–100, 2016.
- [75] Giampaolo Pagnutti and Pietro Zanuttigh. Joint color and depth segmentation based on region merging and surface fitting. In *Proc. of International Conference on Computer Vision Theory and Applications (VISAPP)*, 2016.
- [76] Dario Piatti and Fulvio Rinaudo. Sr-4000 and camcube3. 0 time of flight (tof) cameras: Tests and comparison. *Remote Sensing*, 4(4):1069–1089, 2012.

- [77] Les Piegl and Wayne Tiller. *The NURBS book*. Springer Science & Business Media, 2012.
- [78] Matteo Poggi and Stefano Mattoccia. Learning from scratch a confidence measure. In *BMVC*, 2016.
- [79] Matteo Poggi and Stefano Mattoccia. Learning to predict stereo reliability enforcing local consistency of confidence maps. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2017.
- [80] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *arXiv preprint arXiv:1612.00593*, 2016.
- [81] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5648–5656, 2016.
- [82] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- [83] Fabio Remondino and David Stoppa. *TOF range-imaging cameras*, volume 68121. Springer, 2013.
- [84] Xiaofeng Ren, Liefeng Bo, and Dieter Fox. Rgb-(d) scene labeling: Features and algorithms. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2759–2766. IEEE, 2012.
- [85] Frank Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957.
- [86] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German Conference on Pattern Recognition*, pages 31–42. Springer, 2014.
- [87] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42, 2002.

- [88] Sebastian Schwarz, Marten Sjoström, and Roger Olsson. Time-of-flight sensor fusion with depth measurement reliability weighting. In *3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), 2014*, pages 1–4. IEEE, 2014.
- [89] Akihito Seki and Marc Pollefeys. Patch based confidence prediction for dense disparity map. In *BMVC*, 2016.
- [90] John Sell and Patrick O’Connor. The xbox one system on a chip and kinect sensor. *IEEE Micro*, 34(2):44–53, 2014.
- [91] Konstantinos Sfikas, Theoharis Theoharis, and Ioannis Pratikakis. Exploiting the panorama representation for convolutional neural network classification and retrieval. In *Eurographics Workshop on 3D Object Retrieval*, 2017.
- [92] Baoguang Shi, Song Bai, Zhichao Zhou, and Xiang Bai. Deeppano: Deep panoramic representation for 3-d shape recognition. *IEEE Signal Processing Letters*, 22(12):2339–2343, 2015.
- [93] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.
- [94] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgb-d images. *Computer Vision–ECCV 2012*, pages 746–760, 2012.
- [95] M. Simonovsky and N. Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 29–38, July 2017.
- [96] Ayan Sinha, Jing Bai, and Karthik Ramani. Deep learning 3d shape surfaces using geometry images. In *European Conference on Computer Vision*, pages 223–240. Springer, 2016.
- [97] Natesh Srinivasan and Frank Dellaert. A rao-blackwellized mcmc algorithm for recovering piecewise planar 3d models from multiple view rgb-d images. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 5392–5392. IEEE, 2014.

- [98] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.
- [99] Synth3 dataset. http://lstm.dei.unipd.it/paper_data/deepfusio, Accessed October 15st, 2017.
- [100] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.
- [101] Johan WH Tangelder and Remco C Veltkamp. A survey of content based 3d shape retrieval methods. *Multimedia tools and applications*, 39(3):441, 2008.
- [102] Camillo J Taylor and Anthony Cowley. Parsing indoor scenes using rgb-d imagery. In *Robotics: Science and Systems*, volume 8, pages 401–408, 2013.
- [103] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.
- [104] Beau Tippetts, Dah Jye Lee, Kirt Lillywhite, and James Archibald. Review of stereo vision algorithms and their suitability for resource-limited systems. *Journal of Real-Time Image Processing*, 11(1):5–25, 2016.
- [105] Federico Tombari, Stefano Mattoccia, and Luigi Di Stefano. Segmentation-based adaptive support for accurate stereo correspondence. *Advances in Image and Video Technology*, pages 427–438, 2007.
- [106] Victor A Toponogov. *Differential geometry of curves and surfaces*. Springer, 2006.
- [107] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.
- [108] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. In *Proceeding of the ACM Int. Conf. on Multimedia*, 2015.
- [109] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

- [110] Anran Wang, Jiwen Lu, Gang Wang, Jianfei Cai, and Tat-Jen Cham. Multi-modal unsupervised feature learning for rgb-d scene labeling. In *European Conference on Computer Vision*, pages 453–467. Springer, 2014.
- [111] Jinghua Wang, Zhenhua Wang, Dacheng Tao, Simon See, and Gang Wang. Learning common and specific features for rgb-d semantic segmentation with deconvolutional networks. In *European Conference on Computer Vision*, pages 664–679. Springer, 2016.
- [112] Paul John Werbos. Beyond regression: New tools for prediction and analysis in the behavioral sciences. *Doctoral Dissertation, Applied Mathematics, Harvard University, MA*, 1974.
- [113] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016.
- [114] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015.
- [115] Xu Xu and Sinisa Todorovic. Beam search for learning a deep convolutional neural network of 3d shapes. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 3506–3511. IEEE, 2016.
- [116] Qingxiong Yang, Narendra Ahuja, Ruigang Yang, Kar-Han Tan, James Davis, Bruce Culbertson, John Apostolopoulos, and Gang Wang. Fusion of median and bilateral filtering for range image upsampling. *IEEE Transactions on Image Processing*, 22(12):4841–4852, 2013.
- [117] Qingxiong Yang, Ruigang Yang, James Davis, and David Nistér. Spatial-depth super resolution for range images. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.
- [118] Kuk-Jin Yoon and In So Kweon. Adaptive support-weight approach for correspondence search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):650–656, 2006.

- [119] Pietro Zanuttigh, Giulio Marin, Carlo Dal Mutto, Fabio Dominio, Ludovico Minto, and Guido Maria Cortelazzo. *Time-of-Flight and Structured Light Depth Cameras*. Springer, 2016.
- [120] Pietro Zanuttigh and Ludovico Minto. Deep learning for 3d shape classification from multiple depth maps. In *Image Processing, 2017. ICIP 2017. Proceedings. 2017 International Conference on*. IEEE, 2017.
- [121] Pietro Zanuttigh and Ludovico Minto. Deep learning for 3d shape classification from multiple depth maps. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, 2017.
- [122] Zed stereo camera. <https://www.stereolabs.com/>, Accessed July 31st, 2017.
- [123] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1612.03242*, 2016.
- [124] Wei Zhang. Shift-invariant pattern recognition neural network and its optical architecture. In *Proceedings of Annual Conference of the Japan Society of Applied Physics*, 1988.
- [125] Shuaifeng Zhi, Yongxiang Liu, Xiang Li, and Yulan Guo. Lightnet: A lightweight 3d convolutional neural network for real-time 3d object recognition. In *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2017.
- [126] Jiejie Zhu, Liang Wang, Jizhou Gao, and Ruigang Yang. Spatial-temporal fusion for high accuracy depth maps using dynamic mrfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):899–909, 2010.
- [127] Jiejie Zhu, Liang Wang, Ruigang Yang, and James Davis. Fusion of time-of-flight depth and stereo for high accuracy depth maps. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.
- [128] Jiejie Zhu, Liang Wang, Ruigang Yang, James E Davis, et al. Reliability fusion of time-of-flight depth and stereo geometry for high quality depth maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(7):1400–1414, 2011.