Sede Amministrativa: Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

CORSO DI DOTTORATO DI RICERCA IN: Ingegneria dell'Informazione

CURRICOLO: Scienza e Tecnologia dell'Informazione

CICLO: XXX

SEMANTIC MODELS OF SCENES AND OBJECTS FOR SERVICE AND INDUSTRIAL ROBOTICS

Coordinatore: Ch.mo Prof. Andrea Neviani
Supervisore: Ch.mo Prof. Stefano Ghidoni

Dottorando: Morris Antonello

Semantic Models of Scenes and Objects for Service and Industrial Robotics

A DISSERTATION PRESENTED
BY
MORRIS ANTONELLO
TO
THE DEPARTMENT OF INFORMATION ENGINEERING

FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN THE SUBJECT OF
INFORMATION SCIENCE AND TECHNOLOGY

COORDINATOR: PROF. ANDREA NEVIANI
ADVISOR: PROF. STEFANO GHIDONI

UNIVERSITY OF PADOVA
PADOVA, ITALY
JANUARY 2018

Thesis advisor: Prof. Stefano Ghidoni Morris Antonello

Semantic Models of Scenes and Objects for Service and Industrial Robotics

Abstract

What may seem straightforward for the human perception system is still challenging for robots. Automatically segmenting the elements with highest relevance or salience, i.e. the semantics, is non-trivial given the high level of variability in the world and the limits of vision sensors. This stands up when multiple ambiguous sources of information are available, which is the case when dealing with moving robots. This thesis leverages on the availability of contextual cues and multiple points of view to make the segmentation task easier. Four robotic applications will be presented, two designed for service robotics and two for an industrial context. Semantic models of indoor environments will be built enriching geometric reconstructions with semantic information about objects, structural elements and humans. Our approach leverages on the importance of context, the availability of multiple source of information, as well as multiple view points showing with extensive experiments on several datasets that these are all crucial elements to boost state-of-the-art performances. Furthermore, moving to applications with robots analyzing object surfaces instead of their surroundings, semantic models of Carbon Fiber Reinforced Polymers will be built augmenting geometric models with accurate measurements of superficial fiber orientations, and inner defects invisible to the human-eye. We succeeded in reaching an industrial grade accuracy making these models useful for autonomous quality inspection and process optimization.

In all applications, special attention will be paid towards fast methods suitable for real robots like the two prototypes presented in this thesis.

Supervisore: Prof. Stefano Ghidoni Morris Antonello

Modelli semantici di scene e oggetti per la robotica di servizio e industriale

Sommario

Il sistema percettivo umano si presta alla risoluzione di compiti che possono sembrare banali, ma che al contrario si rivelano essere delle sfide per i robot. La segmentazione automatica degli elementi di maggiore rilevanza o salienza, vale a dire la semantica, ne è un esempio in quanto è soggetta ai limiti dei sensori di visione e all'elevato grado di variabilità del mondo. In particolar modo ne abbiamo esperienza quando sono presenti più fonti di informazione, spesso ambigue, come nel caso di un robot in movimento. Questa tesi dimostra come si possa sfruttare la disponibilità di indizi contestuali e punti di vista diversi per rendere più facile l'attività di segmentazione. A dimostrazione verranno presentate quattro applicazioni robotiche, due progettate per la robotica di servizio e due per un contesto industriale. Verranno costruiti modelli semantici di scene domestiche arricchendo le ricostruzioni geometriche con delle informazioni semantiche che comprendono oggetti, elementi strutturali ed esseri umani. Il nostro approccio sfrutta il contesto, la molteplicità di fonti di informazioni e dei punti di vista, servendosi di esperimenti esaustivi condotti su diversi dataset per dimostrare come questi siano elementi cruciali per aumentare le prestazioni del robot. Inoltre, considerando scenari con robot che analizzano oggetti anziché esplorare l'ambiente, verranno costruiti modelli semantici di polimeri rinforzati in fibra di carbonio arricchendo i modelli geometrici con le misurazioni precise sull'orientazione delle fibre e i difetti

interni non visibili all'occhio umano. Siamo riusciti a raggiungere una precisione di livello industriale rendendo questi modelli utili per il controllo qualità automatico e l'ottimizzazione dei processi. In tutte le applicazioni, un'attenzione particolare sarà dedicata ai metodi più veloci, adatti a robot reali come i due prototipi presentati in questa tesi.

Contents

List of Figures

List of Tables

To Sabrina

Acknowledgments

I WOULD LIKE TO THANK my supervisors at the IAS-Lab, Prof. Stefano Ghidoni and Prof. Emanuele Menegatti, for having guided me through this journey. Furthermore, I would like to thank Prof. Matteo Matteucci and Prof. Domenico Sorrenti for their time as external reviewers. Special thanks go to Prof. Markus Vincze for having welcomed me in the Vision4Robotics Group. I would like to thank Sabrina for her patience and support. Last but not least, a thank you goes to my colleagues, family and friends, who made this journey easier and many moments memorable.

1 Introduction

Visual scene understanding [2–4] is a broad field of study encompassing the development of algorithms with the aim of making machines perceive and understand the contents of the scene as, or even more than what humans do. This kind of capabilities is crucial for making robots autonomous when performing complex operations, both in the domestic and industrial setting, or in the indoor and outdoor scenario.

Despite all the recent advances, what may seem straightforward for the human visual perception system is still challenging for robotic systems and matter of research. As humans, we are naturally capable of decomposing what we see into its semantically meaningful parts [5, 6] even in complex scenarios like in the examples reported in Figure 1.0.1. With just one glance lasting a fraction of second, we can perceive the gist of a scene, both indoor as in Figure 1.0.1(a) or outdoor. While

(a) Example of house view, whose components, object and room structures, can be segmented by our visual perception system.

(b) Example of fabric, whose components, the fibers, can be segmented by our visual perception system.

Figure 1.0.1: Examples of complex scenarios which can be tackled by our visual perception system. In this thesis, algorithms able to segment the main elements of interest and enrich scene or object models with them are the main concern.

driving, we can distinguish between road, cars, signs, sidewalks, building and people. Moreover, we can analyze object surfaces or, more specifically, pieces of fabric as in Figure 1.0.1(b) finding patterns, details or imperfections. As denoted in previous studies [7], information about basic features, the existence of surfaces and objects, and their three-dimensional disposition is all available preattentively to us, hence the information with the highest relevance or salience is subconsciously selected for further and more complete analysis by conscious processing.

In this thesis, we take some steps toward more autonomous and accurate robotic vision systems by studying how to enrich the geometric models of scenes and objects with their semantics, i.e. the properties of interest. Indeed, semantics has to be contextualized in the application since what is important depends on it. In our robotic applications, they will range from elements typically populating an indoor scene: objects, scene structures, i.e. the surfaces of the building like walls, floor and ceiling, [8] or humans [9] to object elements useful for quality control like fibers [10, 11] or object features like defects [12]. In any of these applications,

we will study how to segment the elements of interest exploiting the availability of moving robotic sensors and context information for higher performances.

A scene model can help a service robot to recognize the room or its configuration so as to take informed decisions [13–15]. Knowing that a remote control is more likely to be near a sofa or a TV, the robot can intelligently search for it skipping the bathroom and going directly to the living room. In addition, a semantic model can be further processed by combining it with text understanding techniques like DeepText [16] in order to aid machine-human interaction or support humans in every day tasks [2], e.g. for augmented reality or, more specifically, by describing visual scenes to blind people [17] with verbal sentences. Last but not least, the proposed models can be useful to perform automatic quality controls with robotic systems making them faster, more accurate and freeing humans to spend more time in creative and less repetitive tasks [11, 18].

1.1 Problem Statement

Building semantic models has received much attention by vision and robotics researchers at different scale levels: objects [19], indoor scenes [20] and entire cities [21]. Enriching geometric models with semantics, which may range from the color information in the most basic scenarios to material, object parts, object instances and scene structures is helpful for many applications like video games, quality control, caption generation or decision making, just to name a few.

Building such models with an autonomous robot is full of challenges, especially when multiple shots may be needed for observing the features of interest or to simply obtain a better measurement or observation. Among them, we can list:

- imperfect sensor systems which deliver noisy data and/or incomplete data with a limited field of view;
- the huge variety of objects, scenes and environmental conditions, which can be hardly captured in training datasets;

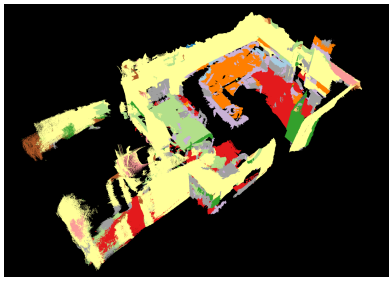
- ambiguous contributions from multiple points of view hindering the correct segmentation of the elements of interest;
- the need for fast methods useful in real robotic applications requiring fast response times and minimal power requirements.

In this thesis, we claim that we can segment the elements of interest, in particular fibers, defects, objects, scene structures or humans, and build semantic models in an accurate and efficient way if we fully exploit the robot capabilities, contextual information and, when possible, fuse the contributions from the multiple available points of view. In the following sections, we present the building capabilities behind our methods, followed by our key contributions and a comprehensive analysis of related works.

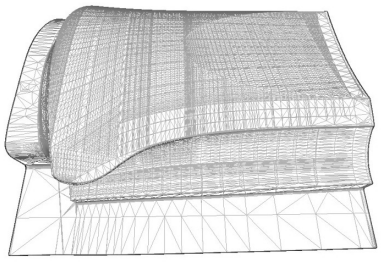
1.2 Common Framework

What is common in the four applications (Chapter 2-5) presented in this thesis is the existence of a geometric model to which semantics has to be mapped. This geometric model can be a map of the environment, in particular a 3D point cloud or a 2D occupancy map created with a Simultaneous Localization And Mapping (SLAM) algorithm, or a 3D model of an object in known pose, in particular a 3D mesh retrieved from a CAD model. Semantics can be a scene property like an object, a scene structure and a human, or an object property like texture, a fiber angle measurement and an inner defect. When segmenting semantics and building the model, contextual information can be taken into account in 2D by reasoning on the 2D image pixel neighbourhoods or in 3D by reasoning on 3D point neighbourhoods. This can be achieved in an efficient way in 2D by means of pixel coordinates but also in 3D if the data is organized and, again, can be accessed by means of pixel coordinates or if we use appropriate data structures. In Figure 1.2.1, two examples of semantic models are reported: the 3D geometric and semantic models of an indoor scene, in this case a living room, see Figure 1.2.1(a)(b), which can help a service robot to take proper decisions, and the 3D geometric and semantic models

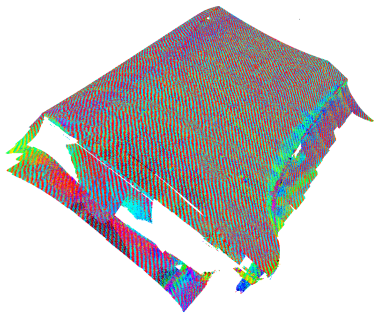
(a) 3D geometric model of a reconstructed scene.



(b) 3D semantic model of a reconstructed scene. Different object and scene structures are displayed with different colors.



(c) 3D geometric model of a complex carbon fiber preform.



(d) 3D semantic model of a complex carbon fiber preform. Different fiber orientations are displayed with different colors.

Figure 1.2.1: Two examples of geometric and semantic models.

of a carbon fiber preform, see Figure 1.2.1(c)(d), useful for quality control.

The underlying capabilities of the proposed methods are the back-projection from 2D to 3D and the forward-projection from 3D to 2D, both of which are displayed in Figure 1.2.2. Given a 2D image I_{ref}, the back-projection from 2D to 3D retrieves the 3D point on the 3D model in known pose corresponding to each 2D pixel. Given a 3D point, the forward-projection from 3D to 2D retrieves the respective 2D pixel in a 2D image. These operations may allow to perform geometri-

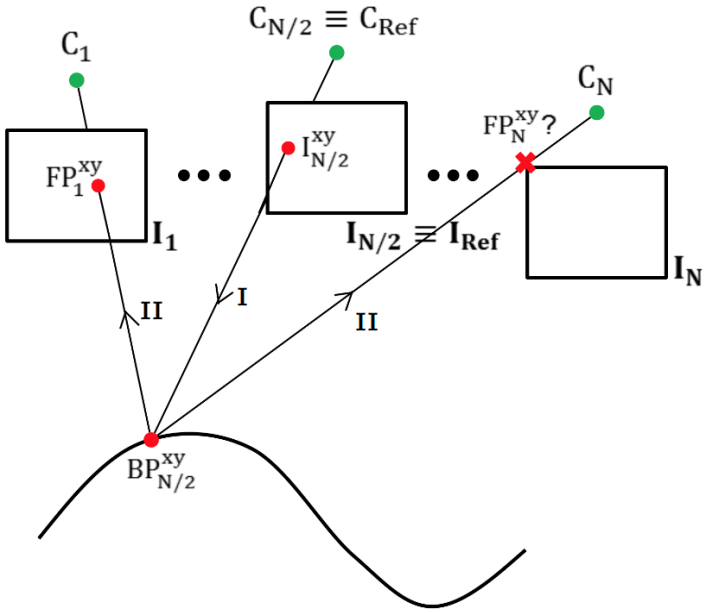


Figure 1.2.2: Multi-view image reprojection process for 6DoF sensor motion and 3D surfaces: examples of back-projection from a 2D image, I_{ref}, to a 3D model in known pose (I) and forward-projection from a 3D point to two 2D images, I_1 and I_N (II).

cal validation steps necessary to improve performances. Furthermore, these operations are useful for multi-view algorithms because, thanks to them, for each point in one view, the point correspondences in all the other views can be retrieved. This is shown in Figure 1.2.2, in which two examples of existent and missing forward-projection are reported. In the presented methods, these modules will be highly optimized exploiting the SSE2 vectorization in order to provide online performances useful in robotic applications. Most of the software has been developed in C++ with the Robot Operating System (ROS) [22], a framework widely used in robotics, and the libraries OpenCV [23] for classic computer vision and machine learning algorithms, the Point Cloud Library (PCL) [24] for 3D perception and Eigen [25] for linear algebra.

1.3 Contributions

We propose novel techniques to build accurate semantic models of scenes and objects from moving autonomous robots. In particular, on the one hand, we build semantic scene models segmenting objects, scene structures (Chapter 2) and humans (Chapter 3), on the other hand, we build semantic models of Carbon Fiber Reinforced Polymers (CFRPs) segmenting carbon fibers in part preforms (Chapter 4) and inner defects in final products (Chapter 5). The main contribution of the thesis is related to the fusion of multiple sources of information like multiple view points for object segmentation and recognition (Chapter 2). This is of great interest when perceiving the environment from a robot; indeed, a robot could plan to acquire new percepts from different perspectives to complement its current belief about the environment. As a further contribution of the thesis, these concepts are successfully applied in various real scenarios having a practical relevance (Chapters 3-5). In each scenario, we cope with different and difficult problems in computer vision, sometimes with specific solutions in order to handle specific problems in a practical and effective way. Indeed, in industrial applications, the application of machine learning techniques may be difficult because of the insufficient amount of data. Thus, our single contributions, in particular the underlying methods and results, are also briefly outlined in the following.

1.3.1 Semantic Segmentation and Mapping of Objects and Scene Structures - Chapter 2

Applications that provide location related services need to understand the environment in which humans live such that verbal references and human interaction are possible. We formulate this semantic labelling task as the problem of learning the semantic labels of the main scene elements from the perceived 3D structure. In this study, we successfully extend and fuse the fast state-of-the-art semantic segmentation approach based on 3D Entangled Forests and the fast state-of-the-art object detector You Only Look Once (YOLO). Furthermore, we propose a novel batch approach and a novel multi-view frame fusion technique to exploit multi-

ple views for improving the semantic labelling results. The batch approach works offline and is the direct application of the single-view 3D Entangled Forest classifier to scene reconstructions with multiple views. The multi-view frame fusion works in an incremental fashion accumulating the single-view results, hence allowing the online multi-view semantic segmentation of single frames and the offline reconstruction of semantic maps. Our experiments show the superiority of our approaches, which lead to a more accurate semantic labelling, in particular more accurate semantic segmentations and maps in a reasonable amount of time.

1.3.2 Semantic Segmentation and Mapping for Detecting Fallen People - Chapter 3

This chapter deals with the problem of detecting fallen people lying on the floor by means of a mobile robot equipped with a 3D depth sensor. The proposed algorithm is inspired by fast semantic segmentation techniques; the 3D scene is oversegmented into small patches and fallen people are then detected by means of two SVM classifiers: the first one labels each patch, while the second one captures the spatial relations between them. This novel approach showed to be robust and fast. Indeed, thanks to the use of small patches, fallen people in real cluttered scenes with objects side by side are correctly detected. Moreover, the algorithm can be executed on a mobile robot fitted with a standard laptop making it possible to exploit the 2D environmental map built by the robot and the multiple points of view obtained during the robot navigation. Additionally, this algorithm is robust to illumination changes since it does not rely on RGB data but on depth data only. All the methods have been thoroughly validated on the IASLAB-RGBD Fallen Person Dataset, which is published online as a further contribution. It consists of several static and dynamic sequences with 15 different people and 2 different environments.

1.3.3 Continuous Segmentation and Mapping of Carbon Fibers to 3D Models - Chapter 4

This chapter describes the problem of segmenting carbon fibers and mapping their 3D orientation measurements to 3D models of CFRPs with known position. This is achieved by means of an inspection robot equipped with a sensor estimating these orientations from multiple 2D images captured with different illumination. In particular, we address the more general problem of mapping large object surfaces with a moving sensor. We propose registration, mapping and filtering algorithms that enable the use of sensors that need multiple shots to perform a measurement in continuous motion. These methods exploit the knowledge of the part shape to inspect in a both efficient and accurate fashion, thus allowing to obtain a measurement quality comparable to that of static measurements, while guaranteeing fast sensor motion and thus short scanning times. Experiments on carbon fiber preforms of complex 3D shape demonstrates that this system accurately segment in real-time the 3D fibers of the outer layer of CFRPs. Accuracy assessments report a measurement accuracy less than one degree on flat surfaces and two degrees on generic 3D surfaces. Qualitative tests mapping the texture of glass fibers onto 3D part models have been performed too. The inspection robot system presented here has been demonstrated both as an in-line quality inspection robot for production of carbon fiber preforms and as a measurement device for improving the draping process in the prototyping of carbon fiber parts. In this chapter, we will introduce also our method for performing a fully automatic hand-eye calibration procedure, which is necessary for completing the work-cell calibration so as to know where the robot is with respect to the inspected part and map carbon fibers to the right points.

1.3.4 Thermographic Segmentation of Defects in Upper Layers of Carbon Fiber Parts - Chapter 5

Many defects affecting the production process of CFRPs are due to the wrong distribution of the thermosetting polymer in the upper layers. In this chapter, they

are effectively and efficiently segmented by automatically analyzing the thermographic images obtained by Pulsed Phase Thermography (PPT) and comparing them with a defect-free reference. The flash lamp and infrared camera needed by PPT are mounted on an industrial robot so that surfaces of CFRP automotive components, car side blades in our case, can be inspected in a series of static tests. The thermographic image segmentation is based on local contrast adjustment via Un-Sharp Masking (USM). Calibration procedures play again an important role: the high level of knowledge of the entire system allows to map each image pixel to the 3D part model improving the segmentation quality and allowing the calculation of the defective areas in 3D world units of measurement. This system could replace manual inspection leading to a substantial increase in efficiency.

1.4 Related Works and Advancements

As previously introduced, the problem of building semantic models has received much attention by vision and robotics researchers at different scale levels: objects, indoor, outdoor scenes and entire cities. Object models, optionally including textural properties, can be exploited for object recognition and tracking [1]. Use cases requiring semantic models of human faces and heads [19] can be content generation for movie production, video games, virtual make over or physical manufacturing of figurines, i.e. 3D printing. Semantic object models of woven material and fabric [11, 26, 27] are of concern for automatic quality control. Analogously, semantic indoor or outdoor scene models can enhance the restoration, analysis and cataloguing of historical buildings [28]. In other contexts, they can be fed to systems reasoning about contents and their representation in the domain of natural language [2, 20] or help robots or autonomous cars in their tasks [3, 13]. Semantic city models [21, 29] can be built for many applications like urban planning (roadwork, architectural or emergency planning), real estate valuation or video games. In any of these applications, the elements of interest, which are to be mapped to the 3D model, have to be segmented. In the past, 2D and 3D image segmentation has been widely studied. In Subsection 1.4.1, we review the literature on semantic

segmentation of objects and scene structures in images, single-view point clouds and whole scene reconstructions. The related state-of-the-art methods addressing the object detection problem will be recapped too. In Subsection 1.4.2, we review the fallen person segmentation problem. In Subsection 1.4.3, we review the literature on the segmentation of thin structures like carbon fibers, but also, more generally, fibers in woven materials and fabric. In Subsection 1.4.4, we review the literature on the thermographic image segmentation of defects, which are salient elements in the thermographic image under analysis.

1.4.1 Object and Scene Structure Segmentation in RGB-D Images or 3D Data

Semantic segmentation [30–32], i.e. the decomposition of a scene in its meaningful parts (objects and scene structures), is receiving lots of attention, because of its importance in scene understanding, robotics and autonomous vehicles. A related research topic is semantic mapping [33, 34], i.e. the construction of 3D scene representations describing scene geometry and semantic content. Semantic maps can be obtained by combining a Simultaneous Localization and Mapping (SLAM) algorithm with a semantic segmentation technique, which is usually applied to single views, or other recognition techniques as in [35, 36]. These research topics are of interest in indoor scenarios, e.g. in the previously cited works, and outdoor scenarios as well, e.g. in [37, 38].

Nowadays, Deep Neural Networks (DNNs), and in particular CNNs, are boosting many fields, including the semantic segmentation of RGB-D images. One of the attempts belongs to Couprie *et al.* [39], who proposed a multiscale CNN architecture to combine information at different perceptive field resolutions and were among the first to train a CNN with depth information for this task. Later, many other approaches have been proposed, e.g. [36, 40–42]. Unfortunately, these algorithms need to be implemented for a high-end GPU in order to achieve real-time performances. Hence, the recent approaches by D. Wolf *et al.* [43, 44], which are fast and light, are of greater interest. In [43], they propose

a framework consisting of three stages requiring less than two seconds per frame. First, it over-segments the scene in such a way that each segment contains at most one object. Second, a Random Forest classifier trained on local hand-crafted features leads to a coarse classification of each segment. Third, a Conditional Random Field leads to a finer classification by modelling simple contextual relations like spatial/temporal consistency and color/spatial relations between pairs of segments. In [44], D. Wolf *et al.* introduce the 3D Entangled Forest as an extension to the standard Random Forest. This classifier is able to model complex contextual features only in one single pass in less than one second per frame on a standard CPU, without complex graphical models, random fields or other post-processings as e.g. in [45]. In this thesis, the capabilities of this approach are further studied. First, it will be coupled with a light SLAM algorithm like the RGB-D modelling technique proposed in [1] or RGB-D SLAM [46], in particular the second version available in ROS [22]. Second, three ways to exploit the availability of multiple points of view will be proposed with the aim of enhancing the semantic segmentation and build semantic maps.

The closest approaches to the work we present in Chapter 2 have been proposed by J. Stückler *et al.* [47], A. Hermans *et al.* [33], Zhao *et al.* [48, 49] and, very recently, J. McCormac *et al.* [34], C. R. Qi *et al.* [50] and L. P. Tchapmi *et al.* [51]. Some of them [33, 34, 47] differ because of the adopted registration system: a Multi-Resolution Surfel Map-based SLAM, a camera tracking system without explicit loop closure and Elastic Fusion, which needs an high-end GPU. All those approaches fused the image segmentations adopting a Bayesian framework. Here, we registered predictions using different systems working on the CPU, i.e. the RGB-D modelling technique proposed in [1] or RGB-D SLAM [46], and we show that, by considering pixel context, our novel fusion scheme is more effective. Furthermore, we added a geometrical verification step, useful for improving the semantic segmentation of the single-frames by leaving out wrong contributions due to frame distortions or alignment errors. Zhao *et al.* [48, 49] worked on the 40 class problem instead of the 13 class problem but proposed two related offline methods. In their former work [48], after voxelization of the reconstructed scene,

each voxel is labelled with the most frequent label among the points in it. Instead, on the one hand, we propose also an online scheme for the semantic segmentation, and, on the other hand, when building a semantic map, we work on points, not voxels, and forward-project each of them to all the other frames to retrieve the right contributions. Finally, unlike them, our multi-view frame fusion improves over the single-view. In their latter work [49], they improve over the previous work using temporal information and higher-order cliques to enforce the label consistency. Anyway, the results on the standard training and test splits in which the dataset is divided are not reported so a direct comparison becomes difficult. Very recently, C. R. Qi *et al.* [50] and L. P. Tchapmi *et al.* [51] have proposed two interesting methods. In contrast with our methods, they are based on deep neural network and are able to work with point clouds, instead of regular 3D voxel grids or collections of images. C. R. Qi *et al.* [50] propose a novel type of neural network that directly consumes point clouds, which well respects the permutation invariance of points in the input. This network, named PointNet, provides a unified architecture for applications ranging from object classification, part segmentation, to semantic segmentation and mapping. L. P. Tchapmi *et al.* [51] present SEGCloud, an end-to-end framework to obtain 3D point-level segmentation that combines the advantages of NNs, trilinear interpolation(TI) and fully connected Conditional Random Fields (FC-CRF) to enforce global consistency. Our batch method working on pre-reconstructed scenes will be compared with both of them. Despite the different application, the work by Tateno *et al.* [52] is also of interest. They enhanced the 3D object recognition and pose estimation by computing a 3D descriptor directly on each 3D cluster and matching it with the single 3D descriptor computed on the fully 3D object model. Their clusters are derived from an incremental segmentation stage exploiting both the single frames and the current reconstructed scene model. Here, the focus is on different techniques and classes but, similarly, we introduce a geometric verification step.

Object detection is also related. Indeed, even if object detectors can provide only the candidate boxes containing the objects of interest, they can be followed by an appropriate segmentation technique [53, 54], thus leading to a final seman-

tic segmentation of the objects of interest. Since the introduction of neural networks, object detectors have also become increasingly faster and more accurate. As reviewed in [55, 56], in the past, the most successful approaches to object detection utilized a sliding window paradigm, in which a computationally efficient classifier like [57–59] tests for object presence in every candidate image window. The steady increase in complexity of the core classifiers has led to improved detection quality, but at the cost of significantly increased computation time per window. Thus, in order to reduce the search space, many top performing object detectors [60–62] work on detection proposals [63, 64], i.e. only a small subset of all the possible windows. Nevertheless, very recently, the state-of-the-art family of detection systems known as You Only Look Once (YOLO), YOLOv2 and YOLO9000 [65, 66] proved that object proposals are not necessary. In contrast to prior works, they apply a single neural network to the full image so its predictions are informed by global context in the image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. These methods are very fast: they process images in real-time with GPU acceleration and, using a lighter model, they can work at about 6-12 seconds per image on a CPU. In Chapter 2, we study how to improve the semantic labeling result by combining the strengths of our semantic segmentation and mapping techniques with such an object detector.

1.4.2 Fallen Person Segmentation in RGB-D Images

To the best of our knowledge, there exist just a few previous approaches trying to detect fallen people already lying on the floor: [67–69]. Both [67] and [68] are specifically designed for mobile robots. In [67], the authors propose a pipeline working on just single RGB images extending a deformable part-based model to the multi-view case for viewpoint invariant lying posture detection. Like us, [68] proposes a pipeline working on depth images. Putative candidates are found by means of a segmentation phase based on an Euclidean clustering. Then, they are

layered so as to face with occlusions and classified by means of a SVM using Histograms of Local Surface Normals. The downside of the approach is the Euclidean segmentation, in particular its distance threshold: if people fall on or near furniture, the segmented object may contain the user and parts of the furniture. On the contrary, in this thesis, we devise a novel approach which concatenates two SVM classifiers, specifically addressing this problem by taking inspiration from previous fast semantic segmentation techniques of scene structures and objects from RGB-D data [43, 44]. The first one labels each patch, while the second one captures the spatial contextual relations between them. Unfortunately, neither the code or dataset of [68] are available making a direct comparison difficult so we implemented our own baseline approach similarly based on the Euclidean segmentation. Finally, in [69], a method for detecting and locating the head of a person lying on the floor by means of a RGB-D sensor is proposed. It would allow to test vital signs on the fallen people, but has not been tested in real cluttered scenarios and requires the head to be visible. Remarkably, none of the previous approaches take advantage of the other functionalities available thanks to the mobile robot like 2D mapping, i.e. the actual knowledge of the environment, and navigation, i.e. the availability of multiple vantage points.

There exist also more specific approaches addressing the detection of falls without any segmentation. These include wearable devices, whose great popularity is linked to the spread of open-source platforms which are small, powerful and connectable to low-cost sensors [70]. In most cases, such sensors include accelerometers [71–73]. These technologies suffer from the difficulty of correctly distinguishing falls from common actions like sitting or lying down. Furthermore, the elderlies easily forget to wear them. Other approaches specifically addressing falls need the installation of environmental devices like microphones [74], infrared, vibration sensors [75] or cameras for person tracking as in recent commercial systems[1] and in [76–78]. Anyway, these approaches are less effective or, being invasive, less accepted.

[1] http://www.nively.com/

1.4.3 Carbon Fiber Segmentation in 2D Images of Carbon Fiber Reinforced Polymers

Previous automatic visual inspection systems for analyzing woven material and fabric are focused on the analysis of their textural properties [26, 79, 80]. The optical properties exhibited by carbon fibers, like specular reflection and light absorption, make this analysis difficult, especially the fiber segmentation and the measurement of the fiber orientation relative to the part [81]. As reviewed in [82], there exist approaches based on gradient vectors in a local neighborhood [83], directional evidence accumulation [84, 85], Gabor filters and steerable pyramids [86], but they share many drawbacks. They need the line-like structure to be visible and strongly rely on parameters like the size of the neighborhood, sub-window or kernel. In addition, they are time-consuming. More recently, an alternative fiber measurement system [87] based on Infrared Thermography and Pulsed Thermal Ellipsometry has been proposed. Nevertheless, it has been tested with final parts instead of preforms and the recording time at each position is in the order of 10 seconds, thus making a quick dense scan prohibitive. Finally, the measured fiber orientations have not been compared with a ground truth, so the precision of this method is unknown.

In contrast, in Chapter 4, we address all of these issues by adopting the photometric stereo system introduced in [82, 88], thus allowing a fast and accurate pixel-per-pixel estimation of the dominant orientations. This system exploits a sensory setup based on photo-metric stereo which takes into account the cone-shaped reflection model typical of carbon fibers. The sensor output is analyzed with a segmentation method tailored towards the typical properties of woven carbon fiber fabrics, that partitions the fabric into single segments for feature calculation and classification. To segment and measure carbon fibers with this system, we must acquire multiple shots of the same fibers with different illuminations. To do so with a moving robot able to inspect entire parts larger than the field of view of the camera, an appropriate image registration technique has to be devised otherwise fibers cannot be accurately segmented. In literature, there exist image registration

methods both using 3D and 2D cues. As reviewed in [89], the open-source Point Cloud Library (PCL) [24] incorporates methods for the initial alignment of point clouds using a variety of local shape feature descriptors as well as for refining initial alignments using different variants of the well-known Iterative Closest Point (ICP) algorithm [90]. Since these approaches require 3D data, they could not be exploited. Furthermore, there exist approaches extracting and matching features in 2D images. They can be based on the SIFT detector and descriptor [91], and RANSAC [92] for the subsequent alignment. Anyway, as we will discuss in Chapter 4, feature-based techniques can be highly unreliable, thus leading to wrong feature matches. Indeed, even if some of them, e.g. SIFT, are known to be robust to illumination changes, they are less robust if the position of the illumination source changes. Instead, we will develop a novel 6-Degrees-of-Freedom registration algorithm exploiting the availability of the robotic arm positions and of the 3D model of the inspected part. Unlike other works on mapping [93], given the availability of the 3D model, 3D reconstruction is not necessary and we consider mapping as the projection of the measured information onto the provided model in known pose. As a further contribution, basic contextual information enhance performances. For each voxel, i.e. volume element, only the best fiber estimation is kept.

1.4.4 Defect Segmentation in Thermographic Images of Carbon Fiber Reinforced Polymers

When defects are invisible and present in the inner layers, the superficial properties of the object cannot be exploited for quality inspection. Many works deal with the applicability of infrared thermography for Non Destructive Testing and Evaluation (NDT&E). Some of them compare multiple thermography techniques [94], while others focus on one of them, for example Pulse Phase Thermography (PPT) [95]. In any case, these techniques have been applied to various case studies, including detection of defects in stainless steel and aluminum [96] or glue deficiency in laminated wood [97] and for superficial defects in multi-layered composites used in

military applications [98]. Furthermore, infrared imaging in medicine has been around since the early 1970s and its application is being investigated for breast cancer [99], burn trauma, diabetes, vascular problems, and neurological problems [100]. In all quoted studies, apart [99], the focus in on the thermography technique itself: thermographic images are calculated and then inspected manually.

Instead, in Chapter 5, we perform automatic defect segmentation, that is achieved by means of thermo-image analysis and comparison to the reference. Especially, first, we segment a region of interest in each PPT phase image, knowing that only the comparison of the glued areas, which appear darker, is of concern, and, second, we compare the restricted PPT phase image under analysis with a defect-free reference. The methods to solve the segmentation problem will be simpler than those addressing similar problems like background/foreground extraction [53], image matting or saliency detection [101] and will not need user interaction. Indeed, the texture in the involved PPT phase images is also simpler than in the images usually considered when validating those techniques, for example outdoor pictures of people with hair blown by the wind. Indeed, segmentation is addressed by adapting an image sharpening technique well-known in photography, the UnSharp Masking (USM), which permits to segment high contrast region taking pixel context into account, and taking advantage of the high level of knowledge of the entire system provided by the calibrations [102], which makes it possible to map each point to the 3D model of the CFRP. With regard to the comparison problem, every PPT phase image will be compared with a defect-free reference previously taken from the same point of view, so that the comparison methods do not need to cover rotations, translations or scalings.

1.5 List of Publications

Parts of the content presented in this thesis have been previously published or submitted for publication in national and international conferences and journals. The following papers refer to the domestic applications presented in Chapter 2, Chap-

ter 3 and Appendix A. They are sorted by relevance as well as order of presentation:

- ***M. Antonello**, D. Wolf, J. Prankl, S. Ghidoni, E. Menegatti and M. Vincze*, "Multi-view 3D Entangled Forest for Semantic Segmentation and Mapping", to appear in Proceedings of IEEE International Conference on Robotics and Automation (ICRA) 2018, Brisbane (Australia), 2018;

- ***M. Antonello**, M. Carraro, M. Pierobon, and E. Menegatti*, "Fast and Robust Detection of Fallen People from a Mobile Robot", in Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), pp. 4159-4166, Vancouver (Canada), 2017;

- *M. Carraro, **M. Antonello**, L. Tonin and E. Menegatti, "*An Open Source Robotic Platform for Ambient Assisted Living", 2nd Italian Workshop on Artificial Intelligence and Robotics, pp. 3-18, Ferrara (Italy), September 2015;

- *G. Beraldo, **M. Antonello**, A. Cimolato, E. Menegatti and L. Tonin*, "Brain-Computer Interface meets ROS: A robotic approach to mentally drive telepresence robots", to appear in Proceedings of IEEE International Conference on Robotics and Automation (ICRA) 2018, Brisbane (Australia), 2018.

The following papers refer to the industrial applications presented in Chapter 4, Chapter 5, Appendix B and Appendix C. Again, they are sorted by relevance as well as order of presentation:

- *M. Munaro, **M. Antonello**, M. Antonello, and E. Menegatti*, "Model-based Image Registration for Continuous Mapping with a Quality Inspection Robot", submitted to IEEE Transactions on Industrial Informatics;

- ***M. Antonello**, M. Munaro and E. Menegatti*, "Efficient Measurement of fiber Orientation for Mapping Carbon fiber Parts with a Robotic System", in Proceedings of the 14th International Conference on Intelligent Autonomous Systems (IAS-14), pp. 757-769, Shangai (China), July 2016;

- ***M. Antonello**, A. Gobbi, S. Michieletto, S. Ghidoni and E. Menegatti*, "A Fully Automatic Hand-Eye Calibration System", in Proceedings of the IEEE International Conference on Intelligent Robots and Systems (ECMR) 2017, pp. 1-6, Paris (France), September 2017;

- *M. Munaro, **M. Antonello**, E. Menegatti and C. Eitzinger*, "FIBREMAP - Automatic Mapping of Fibre Orientation for Draping of Carbon Fibre Parts.", in Proceedings of the International CAE Conference, Pacengo del Garda (Italy), October 2015;

- *M. Munaro, **M. Antonello**, M. Moro, C. Ferrari, E. Pagello, and E. Menegatti*, "Fibremap: Automatic mapping of fibre orientation for draping of carbon fibre parts", in Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13), pp. 272-275, Padova (Italy), July 2014;

- ***M. Antonello**, S. Ghidoni and E. Menegatti*, "Autonomous Robotic System for Thermographic Detection of Defects in Upper Layers of Carbon Fiber Reinforced Polymers", in Proceedings of the IEEE Automation Science and Engineering (CASE) 2015, pp. 634-639, Gothenburg (Sweden), August 2015;

- *N. Castaman, E. Tosello, **M. Antonello**, N. Bagarello, S. Gandin, M. Carraro, M. Munaro, R. Bortoletto, S. Ghidoni, E. Menegatti and E. Pagello*, "RUR53: an Unmanned Ground Vehicle for Navigation, Recognition and Manipulation", submitted to Journal of Field Robotics.

1.6 Outline

The remainder of this thesis is organized as in the following. The next two chapters are dedicated to the problem of building semantic models of scenes. The proposed algorithms are designed to be fast, hence applicable to mobile robots like our

O-Robot [103] presented in Appendix A and our RUR53 [104] presented in Appendix B, with which we ranked third in the Grand Challenge of the first Mohamed Bin Zayed International Robotics Challenge (Abu Dhabi, March 2017). Chapter 2 deals with the classical semantic labelling task formulated as the problem of learning the semantic labels of objects and scene structures from the perceived 3D structure. Chapter 3 extends these techniques to the problem of detecting people lying on the floor with a purely 3D technique. All of our methods have been thoroughly evaluated on various datasets and compared with state-of-the-art techniques. The subsequent two chapters are dedicated to the problem of building semantic object models of Carbon Fiber Reinforced Polymers with a quality inspection robot. In particular, Chapter 4 deals with the problem of analyzing the surface of carbon fiber preforms, segmenting carbon fibers from multiple views and building augmented 3D models with fiber angle measurements. This is useful for finding defects and subsequent process optimization. A handy tool for performing the hand-eye calibration is also introduced. Further details on it are reported in Appendix C. Chapter 5 deals with the problem of segmenting and mapping defects in the glue disposition in the upper inner layers, which are not visible by the human eye. Furthermore, many results of this research have been successfully applied in the context of several national and international projects at the Intelligent Autonomous Systems Laboratory (IAS-Lab): the European projects COROMA (COgnitively enhanced RObot for flexible MAnufacturing of metal and composites), FibreMap, ThermoBot, the regional project Omitech and the international MohammedBin Zayed International Robotics Challenge (MBZIRC). Finally, in Chapter 6, we recap the main results achieved in this thesis and provide possible directions for further developments toward full scene understanding for service and industrial robots.

2

Semantic Segmentation and Mapping of Objects and Scene Structures

In this chapter, we cope with the challenging problem of segmenting objects and structural elements, e.g. walls, floor and ceiling, from RGB-D data recorded with a Kinect-like sensor and from whole scene reconstructions. As introduced in Section 1.4.1, this problem is known as semantic segmentation and mapping. Semantic segmentation [30–32], i.e. the decomposition of a scene in its meaningful parts, is receiving lots of attention in the research community, because of its importance in scene understanding, robotics and autonomous vehicles. Semantic mapping [33, 34] is a related research topic, i.e. the construction of 3D scene representations describing scene geometry and semantic content. Semantic maps can be obtained by combining a Simultaneous Localization and Mapping (SLAM) al-

gorithm with a semantic segmentation technique, which is usually applied to single views, or other recognition techniques as in [35, 36]. For robots, semantic segmentation, whether it is applied to single views or whole reconstructions, plays an important role since, for instance, the former could be applied during the robot exploration of the environment while the latter for planning tasks.

While many approaches have been developed over the last years, we focus on the 3D approach by D. Wolf *et al.* [44], which works on single camera views of indoor environments and relies on the 3D Entangled Forest classifier (3DEF), an extension of the Random Forest. This approach is able to model complex contextual features in a single pass in about one second, without requiring complex graphical models, random fields or other types of post-processing. In particular, their processing pipeline comprehends two stages. First, they over-segment the scene in such a way that each segment contains at most one object. Second, they infer the semantic label of each segment by means of the 3DEF classifier. In particular, the classification of each segment depends on learned geometric relations of neighbouring segments typically appearing in a scene. In this work, we explore the potential of this method when multiple views are available with the aim of enhancing the semantic segmentation of single frames and construct a semantic map like the two scenes shown in Figure 2.0.1(a)(b). Additionally, we study how these approaches can be combined with a real-time state-of-the-art 2D object detector like You Only Look Once (YOLO) [66] to further improve performances with top-down cues, both on single views and when multiple views are available.

The main contributions of this chapter are:

- the development of a 3D semantic mapping approach modifying the original segmentation and applying the 3DEF classifier to whole smooth pre-reconstructed scenes instead of single views;
- a novel 3D multi-view frame fusion technique which improves the semantic segmentation of single-frames and allows the creation of accurate semantic maps in an incremental way;

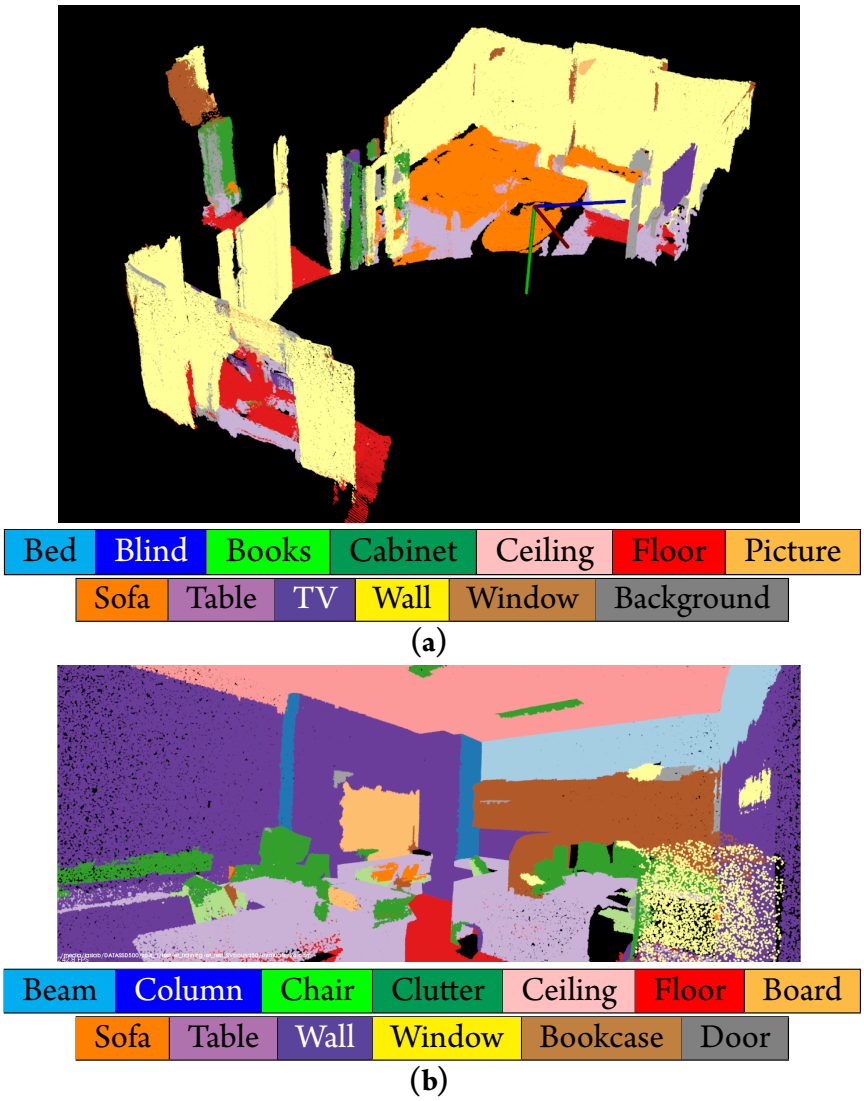


Figure 2.0.1: (a) Example of dense semantic map taken from the NYU dataset and obtained registering the single-frames with the rigid transformations from the RGB-D modelling technique in [1] and the best of our incremental methods. The map is semantically annotated with the 13 classes in the legend below it. (b) Another example of dense semantic map, this time taken from the S3DIS dataset and processed by our batch approach working on whole pre-reconstructed scenes. The 13-class problem involves a different set of labels.

- a novel approach for semantic segmentation and mapping combining in 3D the 3DEF classifier and the 2D YOLO detector for a further boost of performances;
- an optimized pipeline to acquire and annotate a semantic segmentation dataset.

These methods proved to be competitive with respect to the state-of-the-art and, if performances are considered with respect to computational resources or processing time, they lead to even better results.

The remainder of the chapter is organized as follows. Section 2.1 describes our approach for semantic segmentation and mapping with 3D Entangled Forests. First, the application of the original approach to whole scene reconstructions is described (Section 2.2). The original segmentation will be refined since it is unsuitable to segment difficult classes like boards, windows or paintings, which can be easily confused with structural elements like walls. Second, after discussing some limitations of the original approach, in Section 2.3, we present our multi-view frame fusion technique, which allows to improve the semantic segmentation of single frames and build accurate semantic maps. This technique could work also with other semantic segmentation methods. Third, in Section 2.4, we describe our approach to perform semantic segmentation combining the 3DEF classifier and the YOLO object detector for a further performance boost. In Section 2.5, the datasets exploited in this thesis are presented; they consist in: the standard datasets NYUDV1 [30] and NYUDV2 [31], the recent S3DIS dataset[105] and our COROMA dataset. The NYU datasets have been recorded with a Microsoft Kinect V1 sensor. They consist of hundreds of thousands of frames acquired in 79 different scenes and thousands of labeled frames acquired in 527 different scenes, respectively. The most widely used set of labels, introduced by the authors, contains 13 different label classes: scene structures and objects. Instead, the S3DIS dataset[105] is collected with the Matterport Camera[1], composed by 3 structured-light sensors. The dataset consists of 271 rooms from 6 large-scale indoor areas in

[1] https://matterport.com/

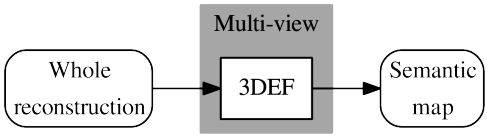


Figure 2.1.1: Overview of the first method for semantic mapping (3DEF-WRSM). It consists in the direct application of 3DEF to whole smooth reconstructions. For this purpose, the preliminary segmentation of the original approach has been modified.

3 different buildings. For each area, the 3D reconstructions of the whole buildings are provided as textured meshes, as well as the corresponding 3D semantic meshes. Its number of RGB and depth images is an order of magnitude up. Our COROMA dataset differs from the others for two reasons: it has been acquired with a Kinect One V2 and in an industrial scenario, in which different elements like machines, robots and tools are present. In Section 2.6, our methods are thoroughly evaluated. Finally, in Section 2.7, our main achievements are recapped and future directions of research identified.

2.1 Methods

In this section, we present our approach to enhance the semantic labelling and build semantic maps. It addresses two different problems: exploiting the availability of multiple points of view and integrating the top-down cues provided by an object detector. The first multi-view method (in the following: 3DEF Whole Reconstruction Semantic Mapping - 3DEF-WRSM) works offline and consists in the direct application of the 3DEF classifier [44] to whole smooth reconstructions. For this purpose, the preliminary segmentation of the original approach has been modified. An overview of the method is given in Figure 2.1.1. The other two multi-view methods are based on our frame fusion scheme: the first of the two enhances the semantic segmentation of single frames by cleverly retrieving the contributions from the adjacent views without storing the whole reconstruction (in the follow-

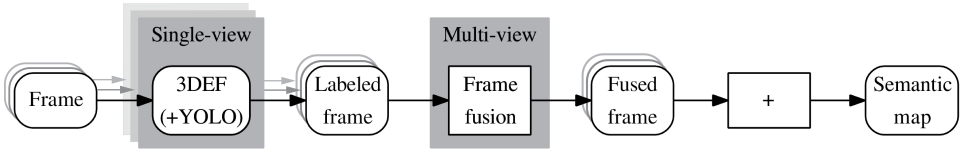
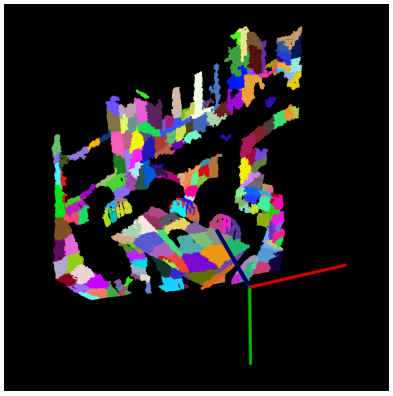


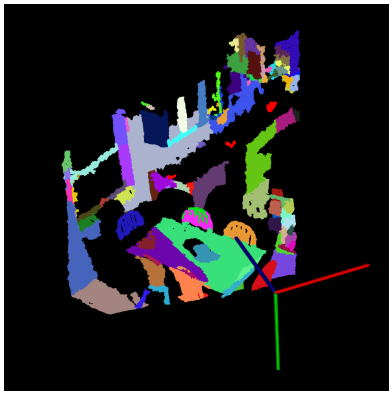
Figure 2.1.2: Overview of our multi-view frame fusion technique, which is the basis of the second (3DEF-FFSS) and third method (3DEF-FFSM) for exploiting multiple views. The difference stays in the number of considered frames: just a few in a neighbourhood or all. Respectively, they allow semantic segmentation, which is online, and mapping, which is offline. Here, for visualization purposes, just three frames are visualized. The 3DEF classifier can be replaced by our semantic segmentation method combining 3DEF and YOLO (3DEF+YOLO).

ing: 3DEF Frame Fusion Semantic Segmentation - 3DEF-FFSS), while the second of the two exploits the same frame fusion technique to build a semantic map (in the following: 3DEF Frame Fusion Semantic Mapping - 3DEF-FFSM). An overview of our frame fusion scheme is given in Figure 2.1.2. First, each frame is labelled by the single-view approach 3DEF. Then, each labelled frame is fused with the contributions from the other points of view. The difference between 3DEF-FFSS and 3DEF-FFSM stays in the number of frames considered: just a few in a neighbourhood or all respectively. Furthermore, 3DEF-FFSS can work online while 3DEF-FFSM works offline. All the presented methods can work with the RGB-D object modelling technique introduced in [1], which can provide nice and smooth reconstructions to 3DEF-WRSM and precise rigid transformations to 3DEF-FFSS and 3DEF-FFSM without requiring a GPU. 3DEF-FFSS and 3DEF-FFSM have been tested also on transformations retrieved with the popular RGB-D SLAM [46], which anyway cannot provide as smooth reconstructions as [1].

A further performance boost is achieved by combining the 3DEF classifier with our 2D semantic segmentation method based on the YOLO detector (in the following: 3DEF+YOLO). This is challenging because we aim at leveraging the best of both algorithms and fusing a 3D approach with a 2D approach. Of



(a) Over-segmentation in supervoxels, small 3D patches.



(b) Oversegmentation in merged supervoxels, larger and mostly planar segments.

Figure 2.2.1: Output of the preliminary over-segmentations.

course, the obtained semantic segmentation can be further improved by means of our multi-view fusion scheme leading to 3DEF+YOLO-FFSS and 3DEF+YOLO-FFSM.

2.2 Semantic Mapping of Whole Pre-reconstructed Scenes

The 3DEF approach in [44] operates on 3D point clouds, which can be recorded with an RGB-D sensor. It comprehends three phases:

- supervoxel over-segmentation in 3D patches, see Figure 2.2.1(a);
- fusion of similar adjacent segments into larger, mostly planar segments as shown in Figure 2.2.1(b). We modified this step;
- segment classification.

Here, we evaluated it on whole scene reconstructions, not just point clouds from single view points. We reconstructed them by means of the RGB-D modelling technique in [1], which can register point clouds from single view points and

smooth depth values thanks to the multiple view points. This way, we evaluated 3DEF-WRSM on smooth reconstructions, not presenting double walls or similar problems of other well-known tools like RGB-D SLAM. As an alternative, they can be acquired and processed directly with a Matterport Camera, whose technology allows to build 3D textured meshes of the scanned area. In the following, each phase is described.

The input point cloud is over-segmented into homogeneous 3D patches by means of the Voxel Cloud Connectivity Segmentation (VCCS) [106]. This solution aims at preserving the edges by finding patches not crossing object boundaries and, at the same time, it reduces the noise and the amount of data. This is a region growing method which incrementally expand patches, in particular supervoxels, i.e. volumetric over-segmentations of 3D point cloud data, from a set of seed points distributed evenly in space on a grid of fixed resolution R_{seed}. Expansion from the seed points is governed by a distance measure D calculated in a feature space consisting of spatial extent, color, and normals, as in the subsequent formula:

$$D = \sqrt{w_c D_c^2 + \frac{w_s D_s^2}{3R_{seed}^2} + w_n D_n^2}\,,$$

in which the spatial distance D_s is normalized by the seeding resolution, the color distance D_c is the euclidean distance in normalized RGB space, and the normal distance D_n measures the angle between surface normal vectors. Three weights can be controlled by the user: w_c, w_s and w_n. This method was proved to be more effective than existing 2D solutions.

In the subsequent step, this approach applies a region growing algorithm, which recursively merges two adjacent segments c_i and c_j into larger ones. The underlying idea is that bigger segments are better since the classifier features tend to be more reliable if calculated on bigger segments. This merging step is performed evaluating a distance function $d(c_i, c_j)$. In particular, given a threshold τ_{merge}, the constraint $d(c_i, c_j) < \tau_{merge}$ must hold. This distance function is a linear combination of the color, surface normal and point-to-plane distance between the seg-

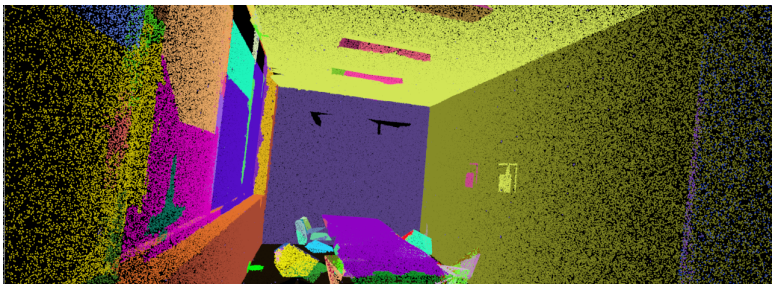
ments, as in the subsequent formula:

$$d(c_i, c_j) = w_c d_c(c_i, c_j) + w_n d_n(c_i, c_j) + w_p d_p(c_i, c_j) \,,$$

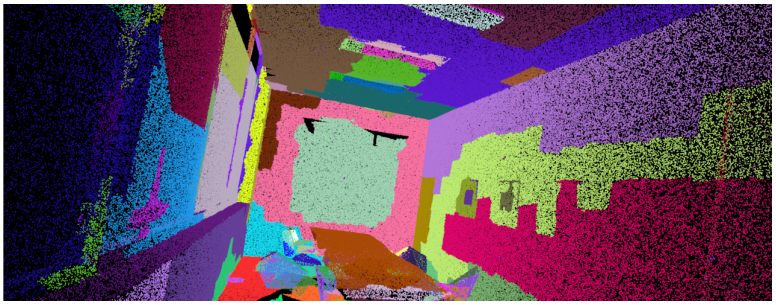
in which d_c is the color distance in Lab CIE 94 color space, d_n the surface normal difference indicated by the dot product $(1 - n_i n_j^T)$, d_p is the max of the point-to-plane distance from c_i to c_j and viceversa. The user can control three weights: w_c, w_n and w_p, normalized to sum up to 1. The algorithm stops if there are no more adjacent segments to be merged and returns the final set of segments $\mathcal{S}$. Empirically, we found out that, notwithstanding the choice of the parameters, this merging strategy is too aggressive on scene reconstructions. Indeed, in the event of contiguous objects with very similar color or geometrical features, the merging algorithm is unable to discriminate among different objects. For instance, pictures and boards are often confused with walls since they hang on them. As a first contribution, during the segment merging, the number of merged supervoxels in each cluster N_{ms} is updated. This way, a stop condition on N_{ms} can be easily evaluated so as to stop the merging and avoid too big segments. Empirically, a maximum bound of 50 proved to be a good choice. An example is reported in Figure 2.2.2.

For each segment generated by the over-segmentation, a feature vector x of length 18 is calculated. Besides simple color features, it includes fast geometric features. Some of them are calculated from the eigenvalues of the scatter matrix of the segment, which represent the variance magnitudes in the main directions of the spread of the segment points. Others are calculated from the Oriented Bounding Box (OBB) including all the segment points. A complete list of features is given in Table 2.2.1. Then, for each segment s_t, a set of close-by-segments s_i is selected on the basis of three constraints: point-to-plane distance, enclosed angles and Euclidean distance. During training and inference, this set can be used to evaluate five binary tests defining the entangled features, which are capable of describing complex geometrical relationship between segments in a neighbourhood. A complete

(a)



(b)



(c)

Figure 2.2.2: (a) Example of tricky scene; (b) The original method often confuses boards (this case), windows and pictures with walls since they hang on them; (c) The choice of setting a maximum bound over the number of merged supervoxels during the region-growing proved to be a good choice to avoid an aggressive merging. Please note that the different colors are random and do not refer to the semantic class. Classification has not yet been performed.

Table 2.2.1: List of unary features calculated for each 3D segment and their dimensionality.

Unary features	Dimensionality
Color mean and std. dev.	2
Compactness (λ_0)	1
Planarity ($\lambda_1 - \lambda_0$)	1
Linearity ($\lambda_2 - \lambda_1$)	1
Angle with floor (mean and std. dev.)	2
Height (top and bottom point)	2
OBB dimensions	3
OBB face areas	3
OBB elongations	3
Total dimensionality	18

Table 2.2.2: List of entangled features calculated for each 3D segment and their dimensionality.

Entangled features	Dimensionality
Existing segment	4
TopN segment	6
Inverse TopN segment	6
Node descendant	5
Common ancestor	5
Total dimensionality	26

list is given in Table 2.2.2. They are briefly explained as follows:

- *Existing Segment Feature*: this evaluates to true if the set of close-by-segments s_i is nonempty;
- *TopN Segment Feature* and *Inverse TopN Segment Feature*: these features take into account the class label distributions of the current tree nodes, which the candidate segments s_i have reached so far during classification. Two parameters are learned: a label l and the bound N. In particular, they evaluate

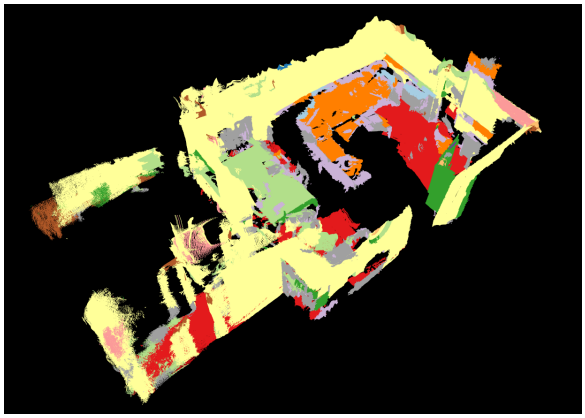
to true if a certain label l is among the most frequent N labels;

- *Node Descendant Feature* and *Common Ancestor Feature*: these features consider the path a target segment s_t or candidate segment s_i took through the tree during classification. Two parameters are learned: a label l and the bound M. They evaluate to true if a certain label l is encountered within M steps.

For further details, we refer to [43].

Many steps in the aforementioned pipeline require the preliminary alignment of the point cloud with respect to the floor. Indeed, unary features like the angle with floor or the height require the floor estimation in order to be calculated. The same holds for some of the constraints to find near-by segments and evaluate 3D entangled features. Here, with respect to the previous work [44], the floor plane is estimated from the reconstructed scenes. Hence, even when the floor is not visible, precise camera height, roll, pitch and yaw can be estimated for each frame, hence many cluster features are expected to be more significant. Despite this improvement and good qualitative results, see Figure 2.2.3, we will see that this method cannot reach the single-view performances on the NYU dataset. Indeed, given that just a few scenes are available in NYUDV1 [30] and only a few frames for each scenes are labeled in NYUDV2 [31], the forests are still trained on the single frames leaving room for improvement in the classifier training. Thus, we trained and tested 3DEF-WRSM also on the more recent, bigger and more precise S3DIS dataset, on which state-of-the-art performances can be reached. Anyway, the need of a big dataset is a non-negligible shortcoming for real applications so we developed two methods based on the fusion of frames from multiple view points, thanks to which single-view performances can be outperformed without retraining any classifier.

(a) Example of reconstructed scene.



(b) Labeling result.

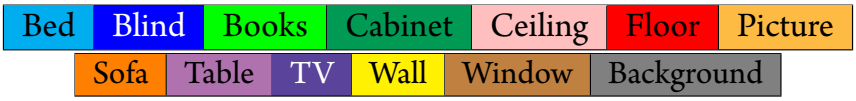


Figure 2.2.3: Example of reconstructed scene obtained by the RGB-D object modelling technique in [1] and fed to 3DEF-WRSM. The labelling result is reported too.

2.3 Multi-view Frame Fusion

This module operates on sequences of RGB-D frames, which may be acquired during the robot exploration of an environment, e.g. while searching for an object requested by the user. These frames may overlap and contain different views of the same entity (object or environmental structure) from varying angulations and distances. This module is decoupled into three steps which could be parallelized: the 3D reconstruction step, the semantic segmentation step and the multi-view frame fusion step. The 3D reconstruction step, here based on the RGB-D modelling technique in [1] and RGB-D SLAM [46], takes a new frame from a sequence of RGB-D frames and registers it to the 3D reconstruction returning its rigid transformation with respect to the reference frame. The semantic segmentation step is the original 3DEF approach applied to each frame. The multi-view frame fusion step, which is the focus of this section, fuses the semantic information for each point in order to exploit the availability of multiple points of view. The same process can be used to improve the semantic segmentation of single frames online or to build an accurate semantic map offline.

Given a sequence S of RGB-D frames I_i with i varying from 1 to N, a reference frame I_{ref} can be selected, e.g. with $\text{ref} = N/2$. Every 3D point P^{xy}, where x and y are the coordinates in the image reference system, belonging to it can be forward-projected to all the other frames in S. This way, the contributions from all the N points of view can be retrieved and, as detailed in the following, used to estimate the optimal label of each point P^{xy}. Figure 2.3.1 shows that the optimal label of $P^{xy}_{N/2}$ can be selected after considering also the contributions from forward-projected points FP^{xy}_i in the frames I_1 and I_N while Figure 2.3.2 shows that not always a forward projection exists so the contribution from some frames can be missing.

Anyway, due to single frame distortions and SLAM imprecisions like double walls or chairs, we cannot be sure that each point $P^{xy} \in I_{\text{ref}}$ truly coincides with the 3D points corresponding to each forward projection $\{FP^{xy}_i\}$. Hence, we introduced a geometrical validation step, each FP^{xy}_i is transformed to the reference

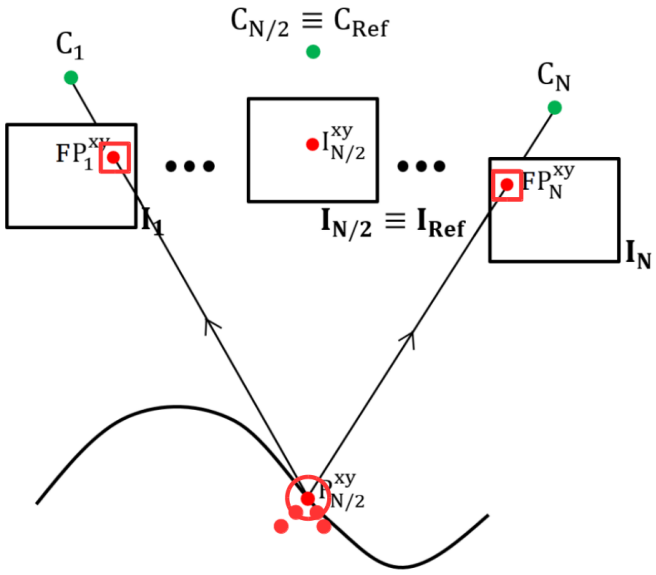


Figure 2.3.1: Forward projection from 3D to I_i, $i \neq$ ref. The red boxes around FP_1^{xy} and FP_N^{xy} denote the Moore neighbourhood. The red circle around $P_{N/2}^{xy}$ the geometric validation step: only the points side it can contribute.

coordinate system and can contribute only if:

$$\left| FP_i^{xy}.z - P_{\text{ref}}^{xy}.z \right| < \varepsilon \tag{2.1}$$

A good ε proved to be 0.05 m. Instead, a quadratic depth dependent threshold, typically used when modelling depth noise with a triangulation-based device like the Kinect V1, did not improve the results. Indeed, here, just the contributions of truly coinciding 3D points are of interest. An example of points that are discarded by this check is reported in Figure 2.3.1, in particular the red points outside the red circle.

To consider the contributions from the other frames, a method based on the Bayesian fusion at the pixel level is considered. Not only this method operates on labels but it takes in input also the classifier confidences. Given a point $P_{\text{ref}}^{xy} \in I_{\text{ref}}$ and the respective forward projected points $\{FP_i^{xy}\}$ with $i \in \{1, \ldots, N\} \bigwedge i \neq$ ref,

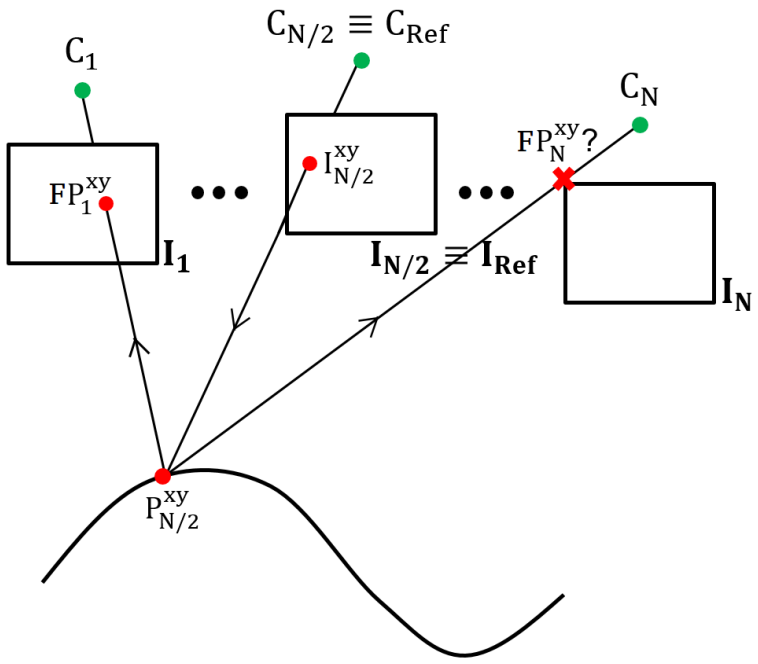


Figure 2.3.2: Example of missing forward projection.

let j be a semantic label and $z^{\text{ref}} = \{z_1, \ldots, z_{\text{ref}}, \ldots, z_N\}$ its measurements in each frame I_i, i.e. the labels assigned to the point $P^{xy}_{\text{ref}}(z_{\text{ref}})$ and its forward-projections FP^{xy}_i (z_i, with $i \neq$ ref). According to Bayes' rule:

$$p(j|z^{ref}) = \frac{p(z_{\text{ref}}|j, z^{\overline{\text{ref}}})p(j|z^{\overline{\text{ref}}})}{p(z_{\text{ref}}|z^{\overline{\text{ref}}})},$$

where $z^{\overline{\text{ref}}} = z^{\text{ref}} \setminus \{z_{\text{ref}}\}$, i.e. the labels assigned to the forward-projections only. Under the assumptions of i.i.d. condition (independent and identically distributed condition) and equal a-priori probability for each class, it can be simplified to:

$$p(j|z^{\text{ref}}) = \tau_j \prod_i p(z_i|j),$$

where τ_j is a normalization factor such that:

$$\sum_{j=1\ldots N} \tau_j p(j|z^{\text{ref}}) = 1 \,.$$

In particular τ_j is calculated as:

$$\tau_j = \frac{1}{\sum_{k=1\ldots N} p(k|z^{\text{ref}})} \,.$$

Parity cases are important and must be addressed appropriately. In the event of parity, the label from the reference frame is kept.

Finally, in contrast with previous works, here, the forward projection is improved by means of a smoothing step. This step takes into account the pixel context so as to improve robustness with respect to imprecisions in the forward projection process, which can be due to noise or locally imprecise registration. Each forward-projected point FP_i^{xy} does not contribute with its label only but with the most frequent label in its Moore neighbourhood, which comprehends the pixel itself and the eight Neighbour Points, NP_{ik}^{xy} with $1 \leq k \leq 8$, see the red boxes enclosing them in Figure 2.3.1. Formally, let $d_{FP^{xy},j}$ denote whether the classifier selects the label j on point FP_{ref}^{xy} or not, and let $d_{NP_{ik}^{xy},j}$ denote whether the classifier selects the label j on point NP_i^{xy} or not. The majority label combination leads to the class J receiving the largest total vote:

$$d_{FP_{\text{ref}}^{xy},J} + \sum_{k\in 1\ldots 8 \bigwedge i\neq \text{ref}} d_{NP_{ik}^{xy},J} =$$
$$\max_{j=1,\ldots,c} \left(d_{FP_{\text{ref}}^{xy},j} + \sum_{k\in 1\ldots 8 \bigwedge i\neq \text{ref}} d_{NP_i^{xy}k,j} \right) \,.$$

In addition, each forward-projected point does not contribute with its label confidences but with those of the neighbour pixel with the most frequent label J in the Moore neighbourhood. Nevertheless, without any geometrical verification step, this method could introduce noise in the labelling results. To be sure that each

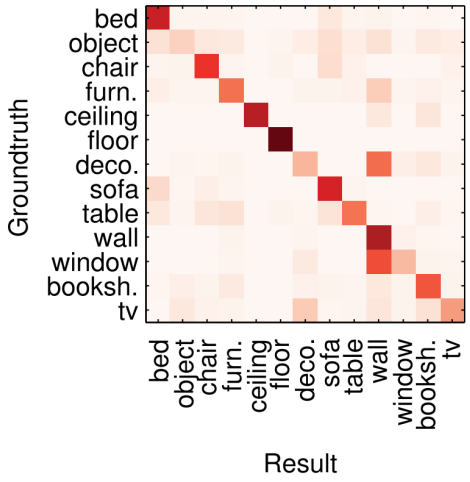


Figure 2.4.1: Confusion matrix of 3DEF on the NYUv2 dataset. Two challenging classes are the labels *Object* and *Furniture*, which comprehend many different objects of different sizes and shapes. The main confusion values appear between *Wall*/*Wall Decoration*, *Wall*/*Wall Window* and *Wall Decoration*/*TV*.

point in the 2D Moore neighbourhood is a real neighbour in 3D, only the points passing the geometrical verification step previously introduced in Equation 2.1 can contribute, in this case:

$$\left| NP_{ij}^{xy}.z - P_{\text{ref}}^{xy}.z \right| < \varepsilon \, .$$

This smoothing step proved to be very useful in practice.

2.4 Joint Semantic Segmentation and Object Detection

From the analysis of the confusion matrix reported in Figure 2.4.1, the 3DEF classifier shows to suffer from a number of shortcomings, which can only be mitigated by the availability of multiple points of view. Two challenging classes are the labels *Object* and *Furniture*, which comprehend many different objects of different sizes and shapes making it very hard for a classifier to capture any distinct properties.

The classes *Wall Decoration*, *Window* and *TV* are also challenging since they all are objects located/mounted on walls so their segmentation can rely mainly on color cues. Given that a multi-view method can only slightly improve over these underlying issues, we further studied how to combine the strengths of 3DEF, which can accurately segment many scene structures and relatively big objects like *Floor*, *Ceiling*, *Wall*, *Bed*, *Sofa*, *Chair* or *Bookshelves* with those of a state-of-the-art object detector trained to be fast and robust when detecting a variety of objects but lacking the capability of segmenting them. This can be achieved with a smart combination of two-state-of-the-art algorithms, respectively aiming at detecting and segmenting objects.

In particular, we selected a real-time object detector known as You Only Look Once (YOLO) [65], more precisely the second version YOLOv2 [66]. This method applies a single neural network to a full RGB frame so that its predictions can be informed by the global frame context. It divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. The network architecture of the first version YOLOv1, see Figure 2.4.2, is inspired by the GoogLeNet model [107] for image classification. The network has 24 convolutional layers followed by 2 fully connected layers. Instead of the inception modules used by GoogLeNet, it uses 1×1 reduction layers followed by 3×3 convolutional layers, similar to Lin *et al.* [108]. The detection framework of YOLOv2 improves in speed and accuracy thanks to various design choices. For a full description, we refer to [66]. A number of models trained on different datasets are available. For our purposes, we selected a model trained on the COCO detection dataset [109], containing over 200 000 images with 80 different object classes, which can be easily mapped to our 13 classes: most of the YOLO classes simply falls in the *Object* class. One of the advantages of YOLO with respect to other methods is that it can be jointly trained on the COCO detection dataset and the ImageNet classification dataset [110], which, containing 14 197 122 images, is orders of magnitude larger, allowing to predict 9000 classes even in the absence of labelled detection data. Nevertheless, on

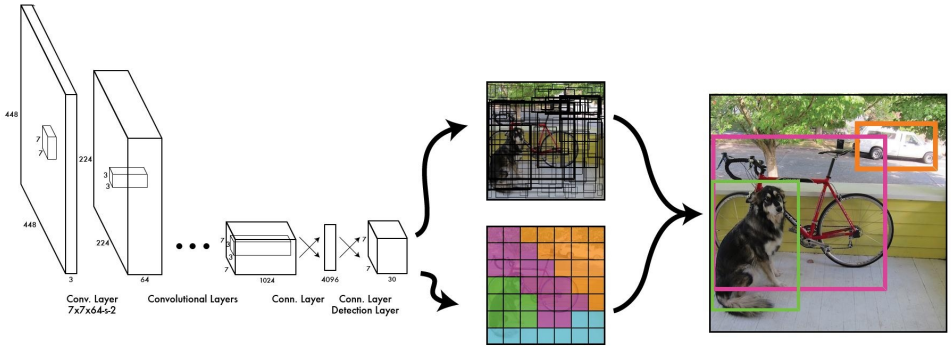


Figure 2.4.2: The YOLOv1 model is implemented as a convolutional neural network. The initial convolutional layers of the network extract features from the image while the fully connected layers predict the output probabilities and bounding box coordinates. These bounding boxes are weighted by the predicted probabilities.

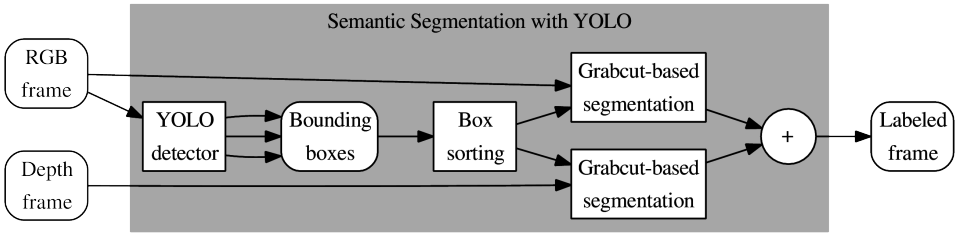


Figure 2.4.3: Overview of the algorithm to perform semantic segmentation with YOLO. In this scheme, for ease of visualization, YOLO detector generates only three bounding boxes.

our classes, this powerful model, YOLO9000, has a lower recall and precision than YOLOv2.

In order to integrate YOLO into our pipeline, the object inside the bounding box has to be segmented alongside its contours, leading to a semantic segmentation. This can be achieved by means of the pipeline illustrated in Figure 2.4.3. For each RGB frame, YOLO finds a set of bounding boxes associated with a label

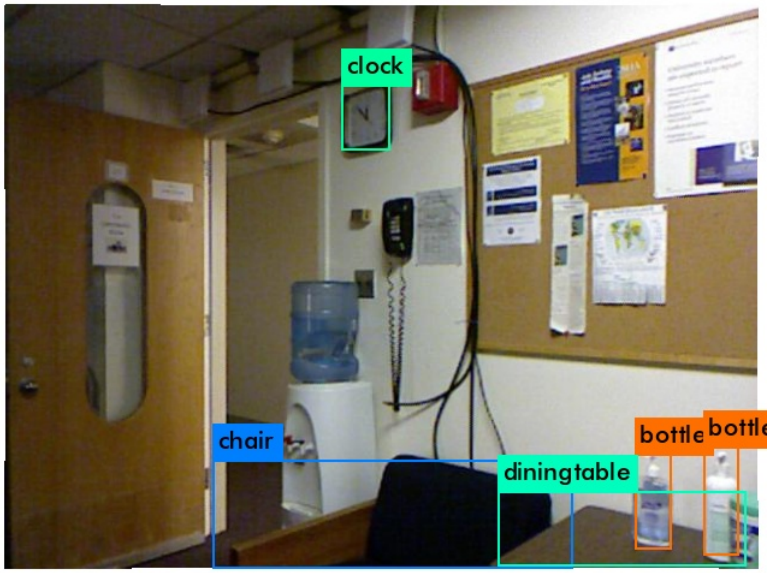


Figure 2.4.4: The YOLO object detector finds a set of bounding boxes, for each of which a label and a confidence are associated. In a straightforward implementation, results may be negatively influenced by the order with which bounding boxes are considered. Here, the bigger boxes, e.g. the box of a table, are segmented before the smaller ones, e.g. the boxes of small objects lying on it. This pipeline leads to a novel 2D semantic segmentation.

and a confidence, as in the example in Figure 2.4.4. Given each bounding box, the detected object is segmented with a method based on Grabcut, a state-of-the-art segmentation algorithm [53], briefly introduced in the following. It requires to be initialized at least with a user-specified bounding box around the foreground region and, if available, some hints on background and foreground pixels inside of it. Then it creates the background/foreground segmentation combining hard segmentation by iterative graph-cut optimization with border matting to deal with blur and mixed pixels on object boundaries. Here, for each detection, Grabcut is initialized with the YOLO bounding box and, for robustness, given that not always a segmentation can be found, Grabcut is run on both RGB and depth frames. This way, the segmentations obtained from RGB and depth frames can be fused. If Grabcut cannot return any segmentation, two alternatives have been considered:

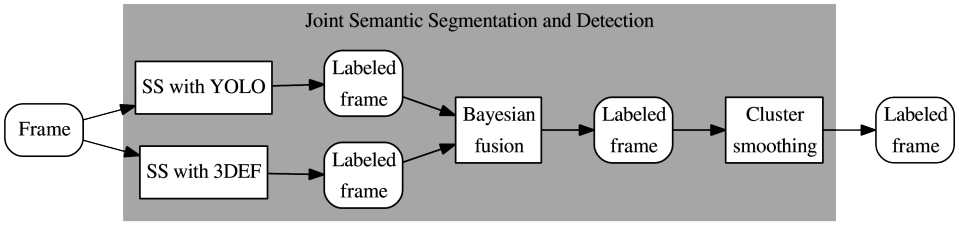


Figure 2.4.5: Overview of the 3DEF+YOLO, our algorithm to combine 3DEF and YOLO. The Bayesian fusion allows to leverage on the strenghts of both methods. The cluster smoothing is a final refinement.

skipping the detection or considering the entire bounding box as foreground. Experimentally, we found out that the second choice is better performing. Indeed, it does not penalize labels like *Object* and *Book*, which can be characterized by tight bounding boxes. After this step, for each segmented pixel, both YOLO label and confidences can be easily memorized. Unfortunately, bounding boxes can overlap so the order with which the bounding boxes are processed may negatively impact the results: for instance, depending on the order with which each bounding box is processed by Grabcut, an object on a table (see Figure 2.4.4) may be segmented before the table itself so the subsequent table segmentation may override the previous object segmentation. Because of this, a straightforward method running Grabcut on each bounding box would lead to a loss of semantic information and fail. Here, as a countermeasure, YOLO bounding boxes are sorted in decreasing order of size. This way, first of all, bigger boxes, which are more likely to be supporting entities like tables, are segmented and, only after, small boxes, which may contain objects lying on supporting surfaces, are segmented. This pipeline leads to a novel 2D semantic segmentation.

Given that YOLO does not support the detection of all the 13 classes, e.g. it cannot detect scene structures like floor, walls and ceiling, the semantic segmentation obtainable from YOLO is incomplete. Hence, as a further contribution, for each frame pixel, the predictions of 3DEF and YOLO are retrieved and fused in

a bayesian way. The two contributions can be easily retrieved in 2D by iterating over the two semantic images, output of 3DEF and our semantic segmentation method based on YOLO. An overview of the fusion process is provided in Figure 2.4.5. Given a frame I and a frame pixel $P^{xy} \in I$, let j be its semantic label, $z_{3\text{DEF}}$ the semantic label returned by 3DEF and z_{YOLO} the semantic label returned by YOLO. According to Bayes' rule and under the assumption of i.i.d. condition, confidences can be accumulated as follows:

$$p(j|z_{3\text{DEF}} \wedge z_{\text{YOLO}}) = \tau_j p(z_{3\text{DEF}}|j) \times p(z_{\text{YOLO}}|j)\,,$$

where $p(z_{3\text{DEF}})$ is the confidence returned by 3DEF, $p(z_{\text{YOLO}})$ is the confidence returned by YOLO and τ_j is a normalization factor such that:

$$\sum_{j=1\ldots N} \tau_j p(j|z_{3\text{DEF}} \wedge z_{\text{YOLO}}) = 1\,.$$

The selected label J is the one with the highest probability:

$$J = \arg\max_j p(j|z_{3\text{DEF}} \wedge z_{\text{YOLO}}).$$

Nevertheless, as shown in Figure 2.4.6, imprecisions in YOLO's bounding box locations or in the Grabcut-based segmentation may lead to the assignment of wrong labels and confidences to the pixels close to the object borders. To alleviate this, a subsequent cluster smoothing step is performed. In contrast with previous steps, this one exploits 3D data, in particular the 3D preliminary segmentation based on the the Voxel Cloud Connectivity Segmentation (VCCS) [106] and the subsequent region growing, see Section 2.2. Given each unlabeled cluster C, which is the output of the preliminary segmentation phase in the 3DEF approach, the most frequent label of the points in C is considered. Each point in C is labelled consistently with the most voted label in the cluster. In the same way, the respective confidences are propagated inside the cluster to all the other points.

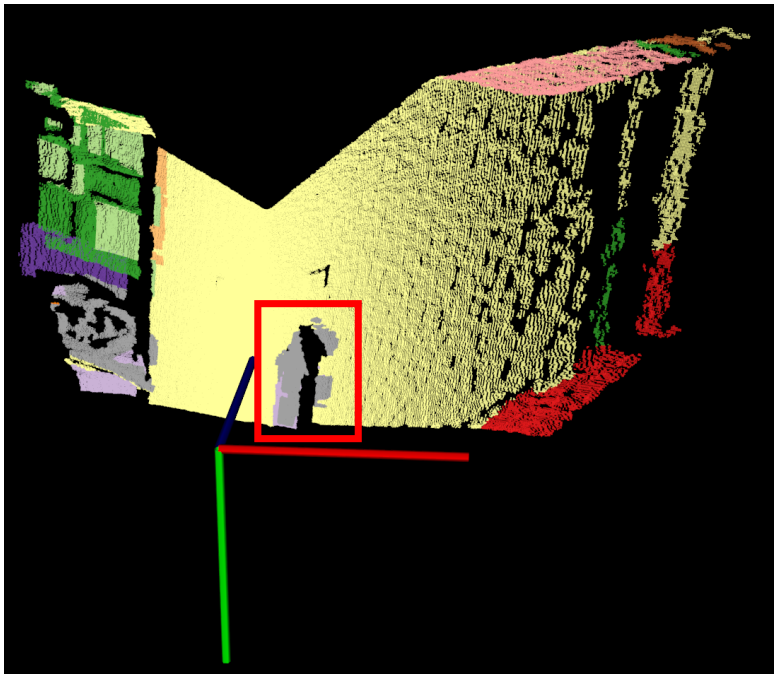


Figure 2.4.6: Without the Bayesian fusion and the final cluster smoothing, a few pixels on the wall in the red box would have been annotated as part of the gray object (a fire extinguisher).

The improvements due to the presented methods will be extensively discussed in the following sections.

2.5 Datasets

For benchmarking, we evaluate our methods for the 13-class semantic segmentation problem on three public datasets: the two NYU Depth datasets, NYUv1 [30] and NYUv2 [31], and the S3DIS dataset [105]. Additionally, we evaluate the 3DEF classifier on our own dataset, the COROMA dataset, with our own 6 classes with the aim of studying its potential also in a real industrial scenario, in which semantic segmentation can aid autonomous robot navigation. These datasets as well as the evaluation pipeline are described as follows.

Table 2.5.1: Training and test splits on the S3DIS dataset.

Fold #	Training (Area #)	Test (Area #)
1	1, 2, 3, 4, 6	5
2	1, 3, 5, 6	2, 4
3	2, 4, 5	1, 3, 6

The NYU Depth datasets contain images from indoor scenes recorded with a Kinect V1 sensor while the S3DIS dataset was collected with a Matterport Camera, which, by means of 3 Kinect-like structured-light sensors, offers a mature technology for scanning large-scale spaces. The NYUv1 dataset provides 2284 pixel-wise labelled RGB-D frames, for which missing depth values in the depth maps have been filled. It includes 10 splits for training and test. The NYUv2 dataset contains 1449 pixel-wise labelled RGB-D frames which is commonly split into a subset of 795 frames for training/validation and 654 for testing. The number of frames in the S3DIS dataset is an order up: it contains 70 496 RGB and depth images. Furthermore, it contains 3D reconstructions of the whole buildings of mainly educational and office use, originated from 6 large-scale indoor areas, leading to a total of 271 3D models of rooms. For each area, they are provided as textured meshes, as well as with the corresponding 3D semantic meshes. Each areas are subdivided in different sets following a 3-cross fold validation scheme in order to avoid overfitting and asymmetric sampling, typical of the subdivision of the dataset in two parts. The splitting scheme is reported in Table 2.5.1. With respect to the others, our COROMA dataset was recorded with a Kinect One V2 mounted on a mobile robot, the one presented in Appendix A: it includes 6548 pixel-wise labelled RGB-D frames, split following the common 70-30 subdivision for training and test. Despite the high number of frames, this dataset is simpler since all the frames were acquired in the same facility.

We evaluated our methods in the following way. As pointed out in [33], the NYUv1 dataset has more labelled images per scene while the NYUv2 dataset has a higher annotation quality making the first more suitable for reconstruction ex-

periments and the second more useful for validating the classifier performances and comparing with many recent single-view approaches. We reconstructed all the scenes in both datasets feeding the RGB-D modelling technique in [1] or RGB-D SLAM [46] with the available raw frames, which are over 25K. Similarly to previous works, when no reconstruction is present, we fall back to the predictions of the baseline single frame. In addition, we did not semantically segment all the raw frames but just those with an available ground-truth annotation. Given the higher variability in scenes and objects, our methods combining 3DEF and YOLO are evaluated on NYUv2. In contrast to the NYU datasets, the S3DIS dataset has far more scenes, which are also more precisely annotated, so our batch approach working on whole scene reconstructions is tested on it. Finally, we delved deeper into an industrial application. Our experiments on the COROMA dataset showcases the power of the 3DEF classifier on a real industrial scenario, which differs from a typical one for a number of reasons: for instance, spaces are wider and include different objects of different materials in different arrangements. Without any shrewdness, acquiring and annotating a dataset composed by thousands of frames is a very difficult task. Hence, our approach is outlined in the next subsection.

2.5.1 COROMA Dataset Acquisition and Annotation

One of the aims of the EU Project COROMA[2] (COgnitively enhanced RObot for flexible MAnufacturing of metal and composites) is the robotic automation of the boat hull sanding procedure, which is heavy and commonly performed by humans. The localization of the boat hull to be sanded can be aided by semantic segmentation for a coarse localization of the boat hull and the main scene elements, e.g. boat hull supports, scene structures, machines, robots and other objects. In order to speed up the acquisition and annotation process, the SceneNN annotation tool in [111] is adopted. In particular, we annotated the whole reconstruction, instead of each single-view frame, whose annotation would have required whole weeks

[2] https://www.coroma-project.eu/

instead of hours. This way, the pixel-wise annotation of each single-view frame can be retrieved by re-projecting the annotation from the 3D model. The entire pipeline consists of three steps, described in the following:

1. scene reconstruction from single-view RGB-D data;

2. manual scene annotation;

3. forward-projection from 3D model to single-view frames.

The SceneNN annotation tool takes in input 3D reconstructions in the form of triangular 3D meshes, i.e. collections of vertices, edges and faces that define the model shapes, and the respective list of frame-by-frame transformations, necessary to re-project the annotation from the scene to each single-view frame. In [111], the input data is obtained by means of Elastic Reconstruction [112], an approach to reconstruction of detailed scene geometry from RGB-D frames. It deals with both sources of error by reconstructing locally smooth scene fragments and letting these fragments deform in order to align to each other. Nonetheless, despite a volumetric registration formulation that leverages the smoothness of the deformation to make optimization practical for large scenes, the approach still requires several hours of processing. Instead, the scene reconstruction task could be performed with one of the faster techniques already mentioned, the RGB-D modelling technique in [1] or RGB-D SLAM [46]. Since real-time performances on a CPU are a minor concern when annotating a dataset, we adopted the popular Elastic Fusion [113], which is a better trade-off between precision and speed. This method works on the GPU, requires some minutes of processing instead of hours and is robust with loopy trajectories. In contrast to state-of-the-art methods, it performs an optimization at the level of the 3D model instead of optimizing the camera trajectory. This way, problems like double walls or double objects are alleviated. Unfortunately, the output of Elastic Fusion is a point cloud, which needs to be converted to a 3D triangular mesh in order to be fed to SceneNN. This was implemented by means of four steps based on tools freely available in MeshLab [114], an open source system for processing and editing 3D meshes:

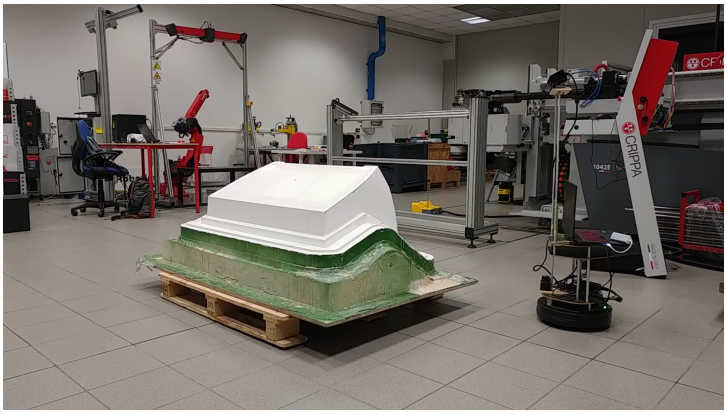
1. cleaning: well-visible outliers, typically due to the sensor noise, can be selected and removed manually thanks to the handy graphical interface of MeshLab;

2. Poisson disk sampling [115]: this step is necessary to reduce the size of the problem and make the subsequent steps faster;

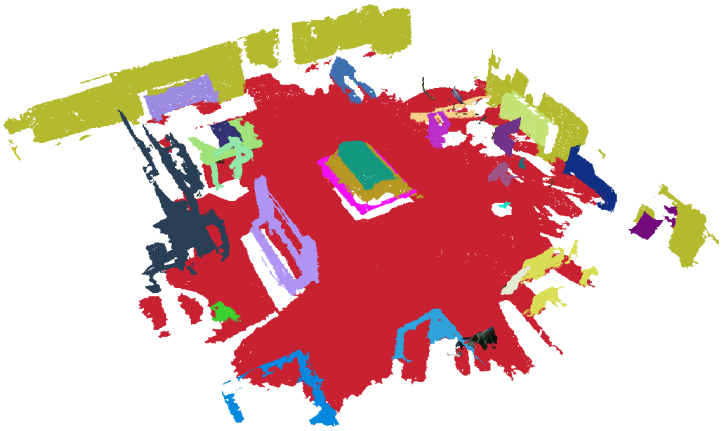
3. normal reconstruction: this step is necessary to calculate the normals required by the subsequent meshing step;

4. surface reconstruction: there exist many remeshing algorithms, the Ball pivoting [116] fits to our purposes since it is light and allows to create a detailed mesh in a few minutes.

By means of the SceneNN annotation tool, both the subsequent steps, manual scene annotation and forward-projection from model to single-view frames, can be performed. The system aids the user in the labeling task thanks to two main functions: automatic 3D segmentation and interactive refinement with merge, extract, split and template matching operations. In Figure 2.5.1, the whole pipeline is recapped. The input RGB-D frames were acquired with our mobile robot in an industrial facility, see Figure 2.5.1(a). Annotating the scene in Figure 2.5.1(b) required only two hours. The final annotation, see Figure 2.5.1(c)(d), is very precise. The boat hull is annotated with three different colors in order to differentiate among the boat hull itself, the technical areas under it and the supporting frame. For our first tests, we considered one frame every three to reduce the amount of data and defined the common 70-30 subdivision for training/validation and test. We plan to extend this dataset with other scenes and adopt a k-fold validation scheme.

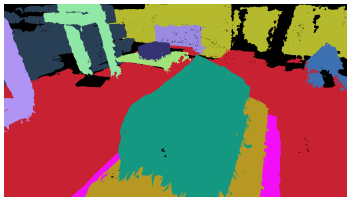


(a)



(b)

(c)



(d)

Figure 2.5.1: Overview of the COROMA dataset: (a) data acquisition in an industrial environment; (b) manually annotated scene reconstruction; (c) view of the boat hull to be localized and sanded; (d) manually annotated view: different colors stay for different semantic classes.

2.6 Experimental Results

2.6.1 Experiments on NYUv1

We compared the three methods we presented, 3DEF-WRSM, 3DEF-FFSS and 3DEF-FFSM, with the single-view approach 3DEF and other state-of-the-art methods. We used three performance indicators: pixelwise recall (in the following: Global Accuracy - GA), classwise recall (in the following: Class Accuracy - CA) and classwise precision (in the following: Class Precision - CP). The first is calculated as the overall portion of correctly labeled points, the second is defined as the mean diagonal of the confusion matrix and the third is the average precision class per class. The last two indicators are less biased towards highly frequent classes.

We tested how much the Bayesian fusion and our fusion scheme considering pixel context can improve over the single-view semantic segmentation for a varying number of forward-projected frames K, with $K = \{2, 4, 10\}$, where $K = N-1$, i.e. the number of frames in the sequence, except of the reference one. This analysis is graphically reported in Figure 2.6.1. The performance indicator has been calculated on one of the 10 default splits. The best K is 10. The single-view is always outperformed by the Bayesian fusion (BF) and the Bayesian fusion with pixel content (CBF).

Furthermore, if K is set to 10 and performances are averaged over all the splits, the Bayesian fusion leads to an improvement in pixelwise recall GA of 1.34%, in classwise recall CA of 0.47%, and in classwise precision CP of 1.92% while our novel Bayesian fusion with pixel context leads to an improvement in GA of 1.58%, in CA of 0.51%, and in CP of 2.36%, proving to be the best online method. If all the frames available in each scene are considered, performances keep increasing: 3DEF-FFSM improves in GA of 1.78%, in CA of 0.62%, and in CP of 2.71%, leading to accurate semantic maps.

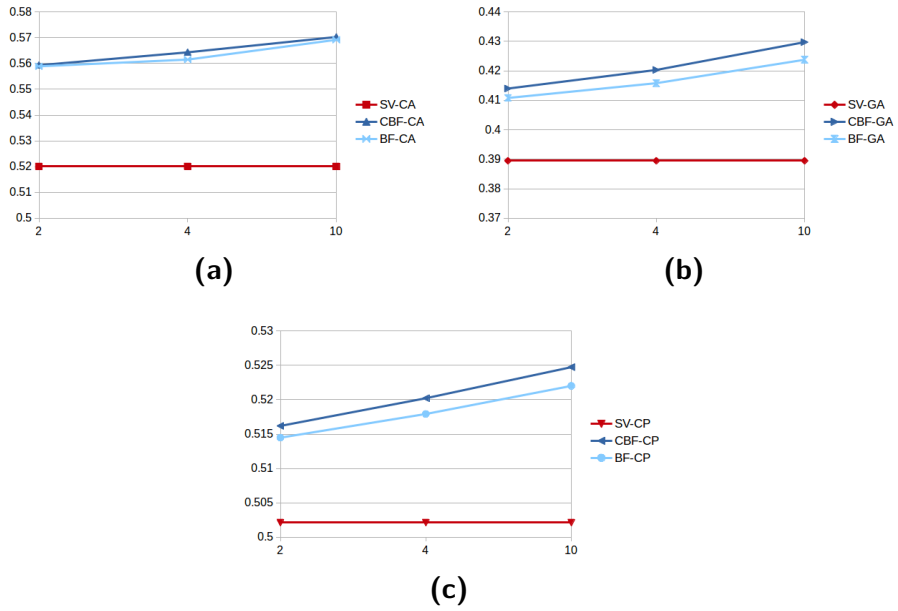


Figure 2.6.1: Comparison in classwise accuracy (CA), global accuracy (GA) and classwise precision (CP) between Bayesian fusion with pixel context (CBF) and Bayesian fusion (BF) for a varying number of forward-projected frames K on one of the default splits. The single-view (SV) is outperformed by the Bayesian fusion (BF) and the Bayesian fusion with pixel context (CBF) for all coefficients and for all K. Furthermore, performance are higher when considering the contributions from more frames.

In Table 2.6.1, we compared all the methods based on 3D Entangled Forests: 3DEF, 3DEF-WRSM, 3DEF-FFSS, with K set to 10, and 3DEF-FFSM, with K set to the maximum number of frames available in each scene. The performance indicators are averaged on the 10 default splits. Finally, we reported also two other methods: Hermans *et al.* [33] and Ren *et al.* [117], a classic CNN approach, which, anyway, takes more than one minute per image. Our single-view method needs one second only. We can see that 3DEF-FFSS outperforms the single-view method. As already discussed, 3DEF-WRSM performs poorly ($-21,3\%$ in CA) indicating that retraining and feature selection on a more adequate dataset is necessary, see Section 2.6.3. Nevertheless, more accurate semantic maps can be ob-

Table 2.6.1: Performance comparison on the NYUv1. The methods are reported in increasing order of pixelwise accuracy GA. The performance values improved by our multi-view methods are enclosed in boxes. The best results are in bold.

Method	GA	CA	CP
3DEF-WSSM	40.0	30.7	27.1
Hermans *et al.* [33]	44.4	59.5	-
3DEF [44]	50.2	52.0	38.9
3DEF-FFSS	51.7	52.5	41.3
3DEF-FFSM	**52.0**	52.6	**41.7**
Ren *et al.* [117]	-	**76.1**	-

Table 2.6.2: Class performance comparison on the NYUv1. The class performance improvements with respect the single-view are enclosed in boxes. The best results are in bold.

Method	Bed	Blind	Books	Cabinet	Ceiling	Floor	Picture	Sofa	Table	TV	Wall	Window	Background
Hermans *et al.* [33]	50.7	57.6	59.8	57.8	92.8	89.4	55.8	70.9	48.4	81.7	75.9	18.9	13.5
3DEF [44]	54.2	13.5	62.9	48.8	88.7	**95.3**	29.1	59.9	62.3	46.3	89.1	11.9	14.1
3DEF-FFSS	56.0	12.5	64.1	51.8	88.4	94.8	25.3	60.7	**65.5**	44.3	**91.0**	12.5	15.9
3DEF-FFSM	55.9	12.2	64.5	52.4	88.2	94.7	26.0	60.7	**65.9**	44.1	**91.2**	12.4	16.1
Ren *et al.* [117]	**85**	**80**	**89**	**66**	**93**	93	**82**	**81**	60	**86**	82	**59**	**35**

tained by 3DEF-FFSM, which is the best performing method. With respect to state-of-the-art methods, we improve in pixelwise recall GA thanks to the good performance on classes like *Bed, Bookshelf, Cabinet, Sofa, Table, Wall, Window* and *Background* but not in classwise recall CA. Indeed, 3DEF is already outperformed by Hermans *et al.* [33] on classes like *Blind, Cabinet, Ceiling, Picture, Sofa, TV* and *Window*. Thus, the improvement of the multi-view on 8 out of the 13 classes does not allow to outperform them on average. Anyway, our fusion scheme could be applied to improve them too. The class performances are all reported in Table 2.6.2.

From a qualitative point of view, our multi-view method improves over the

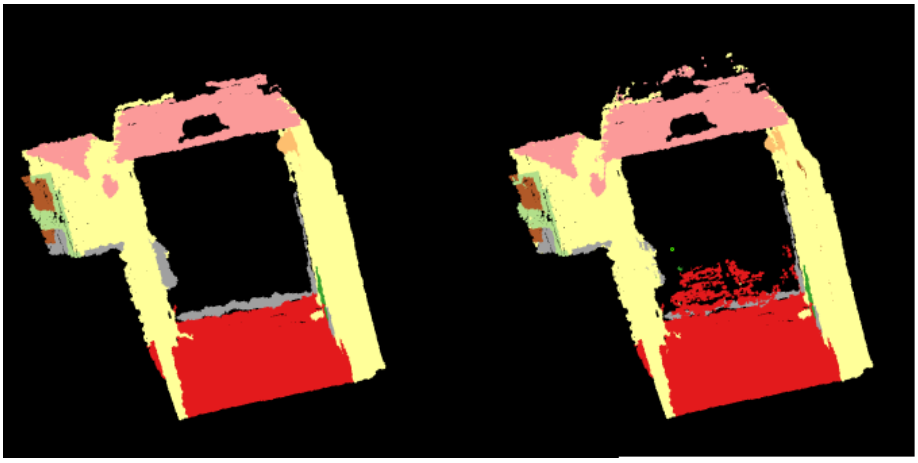


Figure 2.6.2: From a qualitative point of view, our multi-view method improves over the single-view also because it can label a higher number of points. This is clearly visible on the most distant points of the floor (in red) and ceiling (in pink).

single-view also because it can label a higher number of points. In Figure 2.6.2, the single-view semantic segmentation of a corridor scene is compared with the multi-view semantic segmentation by means of our 3DEF-FFSS method. Points, which were filtered out by the single-view approach because they were too far from the camera, are now labelled, this is clearly visible on the most distant points of the floor (in red) and ceiling (in pink).

2.6.2 Experiments on NYUv2

In Table 2.6.3 and Table 2.6.4, we compare the following methods: Couprie *et al.* [32], Hermans *et al.* [33], Eigen *et al.* [41], SEGCloud [50], Eigen-SF [34], 3DEF [44] and our best methods: 3DEF-FFSS, 3DEF-FFSM and their variants combining the 3DEF classifier with the object detector YOLO, 3DEF+YOLO-FFSS and 3DEF+YOLO-FFSM. Couprie *et al.* is a classic CNN approach while

Table 2.6.3: Performance comparison on the NYUv2. The methods are reported in increasing order of pixelwise accuracy GA. The performance improvements with respect to the single-view are enclosed in boxes. The best result are in bold.

Method	GA	CA	CP
Couprie *et al.* [32]	36.2	52.4	-
Hermans *et al.* [33]	48.0	54.3	-
3DEF [44]	65.0	55.7	53.3
3DEF-FFSS/FFSM	65.3	56.1	53.7
Eigen [41]	66.5	59.9	-
SEGCloud [50]	66.8	56.4	-
3DEF+YOLO (rgb only)	67.4	60.9	56.0
3DEF+YOLO	67.6	61.3	56.3
3DEF+YOLO-FFSS/FFSM	67.7	61.5	**56.4**
Eigen-SF [34]	**69.3**	**63.2**	-

Table 2.6.4: Class performance comparison on the NYUv2. The class performance improvements with respect to the single-view are enclosed in boxes. The best result are in bold.

Method	Bed	Object	Chair	Furniture	Ceiling	Floor	Picture	Sofa	Table	Wall	Window	Books	TV
Couprie *et al.* [32]	38.1	8.7	34.1	42.4	62.6	87.3	40.4	24.6	10.2	86.1	15.9	13.7	6.05
Hermans *et al.* [33]	68.4	8.6	41.9	37.1	83.4	91.5	35.8	28.5	27.7	71.8	46.1	45.4	38.4
3DEF [44]	74.2	17.2	63.4	48.1	80.3	98.7	26.5	71.0	46.5	84.0	25.4	55.1	34.1
3DEF-FFSS/FFSM	73.2	17.5	64.5	48.8	80.2	**98.7**	27.2	**74.5**	50.4	84.2	29.5	56.0	42.7
Eigen [41]	42.3	46.5	72.4	60.8	73.1	85.7	57.3	38.9	42.1	85.5	**55.8**	49.1	68.5
SEGCloud [50]	75.1	39.3	62.9	61.8	69.1	95.2	34.4	62.8	45.8	78.9	26.4	53.5	28.5
3DEF+YOLO	86.9	17.7	**82.4**	55.0	79.2	96.8	24.1	71.6	51.4	82.7	25.0	66.3	57.5
3DEF+YOLO-FFSS/FFSM	**87.8**	17.7	82.3	54.8	**81.3**	96.6	23.0	71.6	**51.2**	82.7	25.8	**66.7**	57.3
Eigen-SF [34]	47.8	**46.7**	73.3	**62.8**	79.0	90.5	**64.5**	45.8	46.0	**88.5**	55.2	50.8	**70.7**

Eigen *et al.* [41] and SEGCloud [50] are two more recent ones. Eigen-SF is Semantic Fusion, which combines Eigen *et al.* [41] with Bayesian fusion. Apart from Hermans *et al.*, they all require an high-end GPU.

On this dataset, the average number of labelled frames per scene is only 2.74. On NYUv1, they were 28.91. This penalizes the performance benefit of our multi-

Table 2.6.5: Class performance differences between the two best methods on the NYUv2. 3DEF+YOLO-FFSS/FFSM outperforms Eigen-SF in 7 out of 13 classes. Improvements are in bold.

Methods	Bed	Object	Chair	Furniture	Ceiling	Floor	Picture	Sofa	Table	Wall	Window	Books	TV
3DEF+YOLO-FFSS/FFSM vs Eigen-SF [34]	**+40.0**	-29.0	**+9.0**	-8.0	**+2.3**	**+6.1**	-41.5	**+25.8**	**+5.2**	-5.8	-29.4	**+15.9**	-13.4

view method, which we demonstrated to improve when increasing the number of forward-projected frames. Indeed, 3DEF-FFSS still improves over the single-view 3DEF but less than before: the pixelwise recall increases of 0.3 % and the classwise recall and precision increase both of 0.4 %. Our method is the best one working on the CPU only. As far as the comparison with deep learning methods is concerned, there is still a gap of 1.2 % in GA and 3.8 % in CA with respect to Eigen *et al.* and 4.0 % in GA and 7.1 % in CA with respect to the best method, Eigen-SF. On the one hand, our approach could be possibly improved by considering all the available frames, not only the annotated ones, as done by the competing methods. On the other hand, we could adopt a common post-processing phase like a CRF, which we would like to explore soon. It would be interesting to apply our fusion scheme to Eigen *et al.* in order to see if it can improve over Eigen-SF, which is based on a standard Bayesian fusion.

As reported in Table 2.6.3, a significant boost in performances is obtained by combining the 3DEF classifier and the YOLO detector. The single-view 3DEF+YOLO outperforms Eigen *et al.* of 1.1 % in GA and 1.4 % in CA, and SEG-Cloud of 0.8 % in GA and 4.5 % in CA. As shown in Table 2.6.3, these improvements are slightly reduced if the bounding box segmentation is performed on the RGB image only. The multi-view 3DEF+YOLO-FFSS and 3DEF+YOLO-FFSM keeps improving over 3DEF+YOLO of 0.1 % in GA, 0.2 % in CA and 0.1 % in CP leading to a total improvement with respect to the starting 3DEF of 2.7 % in GA, 5.8 % in CA and 3.1 % in CP. Anyway, with respect to the best method, Eigen-SF,

there is still a gap of 1.6 % in GA and 1.7 % in CA. Interestingly, as reported in Table 2.6.4, this performance gap is due to the bad results on the classes *Object*, *Furniture*, *Picture*, *Wall*, *Window* and *TV*. Nonetheless, our methods are the best performing on a number of classes, in particular 7 out of 13: *Bed*, *Chair*, *Ceiling*, *Floor*, *Sofa*, *Table* and *Bookshelf*. For clarity, in Table 2.6.5 the class performance differences between our best methods 3DEF+YOLO-FFSS/FFSM and Eigen-SF are reported.

2.6.3 Experiments on S3DIS

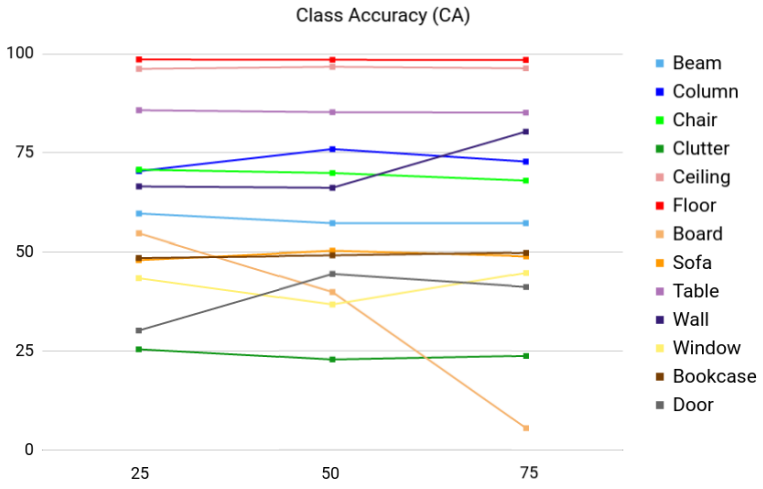
In Subsection 2.6.1, 3DEF-WRSM performed poorly since the NYUDv1 dataset does not provide an adequate amount of annotated data for retraining. In this Subsection, we give another chance to this method, retraining the 3DEF classifier on the whole pre-reconstructed scenes available in the S3DIS dataset. In a first round of tests, we worked on the first fold in Table 2.5.1 and performed a number of experiments in order to find the best parameters for both the segmentation and classification phases. In order to speed up the validation process, we deactivated the 3D entangled features, which are computationally expensive[3], and trained the 3DEF classifier with the unary features only. Thus, we tested the default hyperparameters, activating and deactivating augmentation to check if it is as crucial as it was for the original 3DEF. Then, we analyzed performances with varying N_{ms}, the maximum bound on the number of merged supervoxels introduced in Section 2.2. We also varied one of the main hyperparameters of the random forest classifier: the number of trees in the forest. Finally, in a second round of tests, we followed the 3-fold validation scheme in Table 2.5.1. We activated the 3D entangled features in order to understand the importance of contextual cues when working on whole pre-reconstructed scenes. We also compared 3DEF-WRSM with two recent state-of-the-art methods, PointNet [50] and SEGCloud [51], which are based on deep neural networks and work on pointclouds.

First of all, we trained 3DEF-WRSM with the same hyperparameters proposed

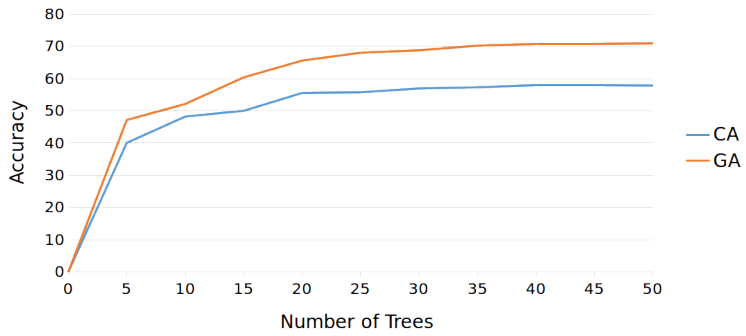
[3] Training can last about 30 min instead of about 1 week on a high-end CPU.

in the original work [44], i.e. we trained a forest composed by 40 trees of depth 20. The bagging rate, i.e. the fraction of training data to be used for each tree, was kept to 0.3. In contrast to [44], we found out that augmenting the training set, i.e. artificially extending the set of available data, is a bad choice, which leads to a performance detriment of 27.5% in CA and 29.9% in GA. Indeed, the augmentation procedure in [44] is performed by mirroring and applying small random rotations about each axis before segmentation. This procedure aims at making the 3DEF classifier more robust in the event of wrong estimations of the floor plane, on which most of the features depends. This is useful when working on the NYU datasets since, in many frames, the floor may fall out of the field of view and an estimation is necessary, e.g. by means of the previous frames on which it was still visible. Nevertheless, in our case, given that the floor is always present and well recognizable in the S3DIS dataset, augmentation may simply confuse the classifier. We also analysed performances at varying values of N_{ms}, the maximum bound on the number of merged supervoxels, and varying values of the number of trees in the forest. The two trends are reported in Figure 2.6.3. Interestingly, see Figure 2.6.3(a), the bound on the number of merged supervoxels proves to be very useful for the class *Board*, which improves of 49.2 % when N_{ms} equals to 25 and 34.4 % when N_{ms} equals to 50. We selected the latter value since, with it, most of the classes improve in accuracy, in particular 8 out of 13. In Figure 2.6.3(b), we report both accuracy coefficients CA and GA for various numbers of trees: as with the original 3DEF, both coefficients hit a plateau when the number of trees is approximately 40, thus confirming the starting choice.

Finally, in Table 2.6.6, we compare two recent competitors, PointNet [50] and SEGCloud [51], with two versions of 3DEF-WRSM: 3DEF-WRSM (unary only), on which 3D entangled features are disabled, and the full 3DEF-WRSM with unary and 3D entangled features. 3DEF-WRSM (unary) outperforms the other methods. In particular, the class accuracy CA improves of 8.9 % with respect to PNET [50] and 0.5 % with respect to SEGCloud [50], showing that handcrafted unary features are competitive. We did not report other performance coef-



(a)



(b)

Figure 2.6.3: Sensitivity analysis on the main parameters: (a) the class accuracy is reported for different N_{ms}, the maximum bound on the number of merged supervoxels. The bound on the number of merged supervoxels proves to be very useful for the class *Board*; (b) the number of trees in the forest is varied. Both performance coefficients hit a plateau when the number of trees is approximately 40.

ficients because not available. 3DEF-WRSM (full) is not as good as 3DEF-WRSM (unary) in CA, showing that further study is necessary to assess the power of 3D entangled features when handling reconstructed scenes instead of single frames.

Table 2.6.6: Performance comparison on S3DIS. The methods are reported in increasing order of pixelwise accuracy GA. The best result are in bold.

Method	GA	CA
PNET [50]	-	49.0
SEGCloud [51]	-	57.4
3DEF-WRSM (unary)	70.9	**57.9**
3DEF-WRSM (full)	**71.8**	54.3

Table 2.6.7: Class performance comparison on the S3DIS dataset. The best result are in bold.

Method	Beam	Column	Chair	Clutter	Ceiling	Floor	Board	Sofa	Table	Wall	Window	Books	Door
3DEF-WRSM (unary)	**48.66**	47.29	71.19	**24.34**	95.38	**98.32**	**38.33**	**45.96**	77.21	73.66	33.24	46.51	52.20
3DEF-WRSM (full)	42.03	**50.37**	**77.76**	21.49	**95.44**	98.21	10.47	23.51	**79.59**	**76.39**	**36.73**	**53.32**	**57.15**

Anyway, the class accuracy CA still improves of 5.3 % with respect to PNET [50]. Instead, CA is lower with respect to SEGCloud: it decreases of 3.1 %. To further study the behaviour of 3DEF-WRSM when disabling and enabling entangled features, class per class performances are reported in Table 2.6.7. Even if 3DEF-WRSM (full) outperforms 3DEF-WRSM (unary) on the majority of classes, 8 out of 13, it is highly penalized by the bad performances on the classes *Board* and *Sofa*, whose greatest confusions are with the classes *Wall* and *Chair*. We could possibly improve these results by designing new unary or entangled features.

To further examine the role of each feature, we calculated a separate histogram of the selected feature types for each depth level, normalized across a forest with 40 trees. Figure 2.6.4 shows a visualization as a heatmap, displaying which features have been selected the most at which depth level and are therefore the most suitable ones to separate the classes. As with the single-view 3DEF [44], the most distinctive unary features are the minimum and maximum height of a segment.

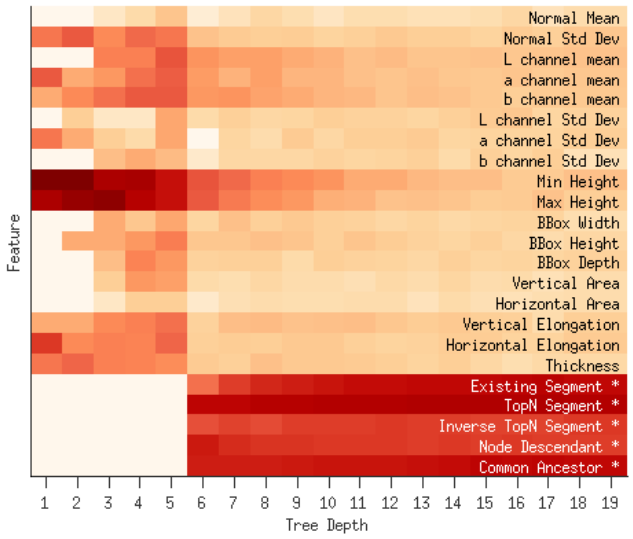


Figure 2.6.4: Distribution of learned split features, depending on the depth level of the trees (3D entangled features marked with *). The darker the cell, the more often the feature has been selected at a particular depth (data accumulated over 40 trees.

Vertical elongation and the thickness of a segment still show to carry important information. As soon as the 3-D entanglement features are activated (level 6), they are clearly the most frequently selected features of the whole set, with the *TopN Segment Feature* being on top. Similarly to the single-view 3DEF, the least selected features are the *Inverse TopN Segment* and the *Node Descendant*. There are also a few differences: the normal mean is not as important as before; instead, the normal standard deviation and the horizontal elongation are more significant. Nevertheless, in general, this analysis agrees with the previous work on single-views, demonstrating the feasibility of the feature set even if calculated on whole 3D reconstructions.

A qualitative analysis, see Figure 2.6.5 for a few examples, leads to very positive results. For 10 scenes, we report: the input point cloud, the ground truth, the output of 3DEF-WRSM after disabling entangled features, the output of the full

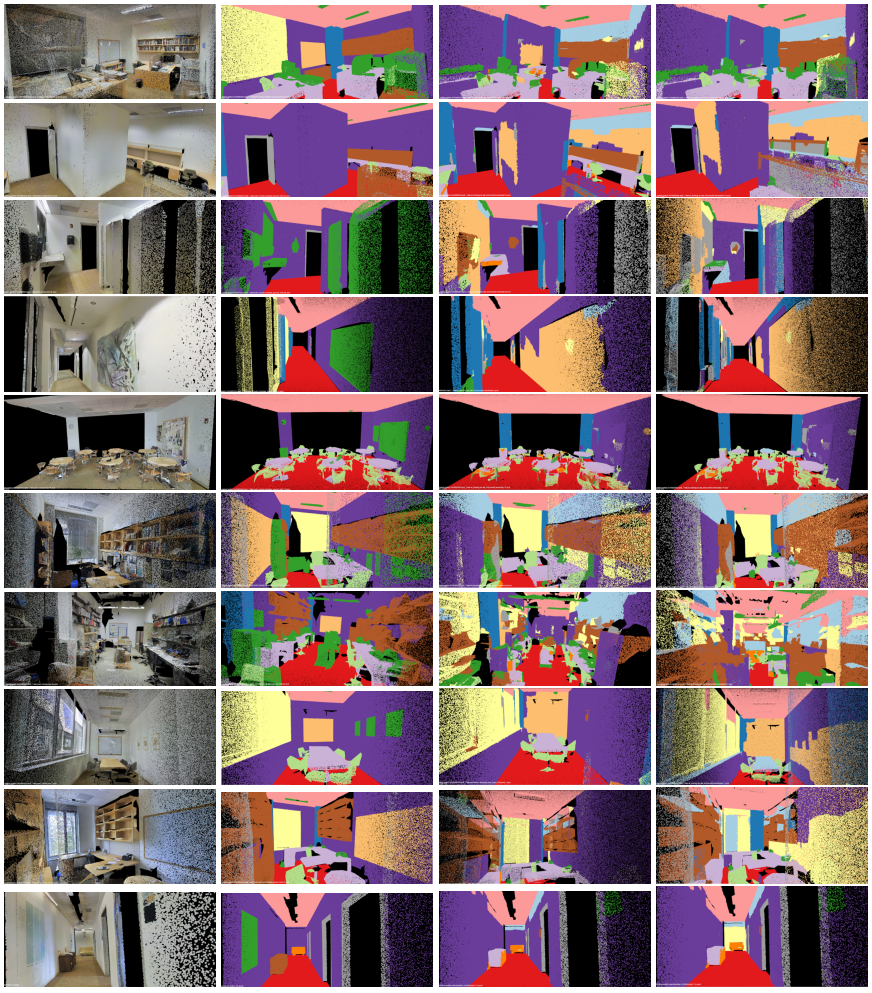
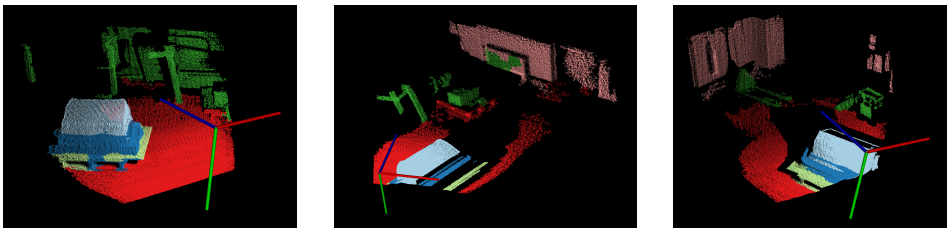


Figure 2.6.5: Qualitative results of the 3DEF-WRSM on S3DIS, in particular input point cloud (first column); ground truth (second column); 3DEF-WRSM without entangled features (third column); full 3DEF (last column). In contrast to previous works, classes which tend to be merged in larger class segments like the *Board* class are successfully recognized.

3DEF-WRSM. The benefit of the bound on the number of supervoxels is clear, e.g. by looking at the *Board* class segmentation results. Classes which tend to be merged in larger class segments are well-recognized.

Table 2.6.8: Class and global performances on the COROMA dataset.

Method	Boat Hull	Technical Area	Frame	Clutter	Wall	Floor	GA	CA	CP
3DEF [44]	95.7	77.3	76.5	94.5	93.7	98.6	95.3	89.2	90.7



(a) (b) (c)

Figure 2.6.6: Qualitative results of 3DEF on the COROMA dataset, in particular different views of the main object of interest: the boat hull to be sanded automatically.

2.6.4 Experiments on COROMA Dataset

On this dataset, we tested only the original 3DEF classifier, leaving the evaluation of the other methods as a future work. These results are interesting to evaluate the power of this tool on a real industrial scenario, which can be simpler when the requested degree of variability is low, as well as the number of classes of interest. Indeed, in this use case, both the boat hull to be recognized and the environment might not change much over time. The average performances are very good: 95.3 % in GA, 89.2 % in CA and 90.7 % in CP. The most challenging classes are the *Technical Area* and *Frame* classes, mainly because of undersegmentation. Indeed, their preliminary segmentation by means of supervoxels is harder given that they are thin and adjacent structures.

Qualitative results are also very good. In Figure 2.6.6, three examples depicting the main object of interest, i.e. the boat hull, from three different perspectives are

reported.

2.6.5 Runtime Analysis

We tested our system on the laptop Dell Inspiron 15 7000. It runs Linux Mint 17.30 and is equipped with an Intel Core i7-6700HQ CPU with 4 cores clocked at 2.60 GHz, the graphic card NVIDIA GeForce GTX 960M and 16 GB of DDR3 RAM. Our batch approach can classify a whole reconstructed scene in about 4.6 min. As with the single-view approach, the bottleneck stays in the preliminary segmentation. Our online system based on the frame fusion scheme makes use of a fast technique for semantic segmentation, which requires only approximately 0.7 s per frame on the CPU and a fast object detector, which works at approximately 4.2 fps on the GPU. Our fusion scheme can work online with full resolution images (640×480 pixels). In particular, it can work at an average speed of 2.2 fps when the number of re-projected frames is 2, 1.1 fps if they are 6 and 0.9 fps if they are 10. Of course, its time complexity depends on the number of forward-projected frames. Our semantic segmentation combining YOLO and Grabcut works at an average speed of 0.9 fps, depending on the number of bounding boxes detected in each frame. As done by other methods, we could work at lower resolution. In this case, real-time performances could be easily reached. Runtimes are important on mobile robots for fast reaction times and human interaction. In comparison with the state-of-the-art, our accuracies may be lower in some cases but our results are outstanding if considered with respect to computational resources and processing time.

2.7 Summary

In this work, we proposed a batch method and a multi-view frame fusion technique, useful online and offline, for enhancing the semantic labelling results with 3D Entangled Forests. All the approaches proved to be valuable choices for batch and online semantic segmentation and mapping on various datasets, NYUv1, NYUv2 and S3DIS. The two methods based on the multi-view frame fusion tech-

nique improves over the single-view in recall and precision as well on the popular datasets NYUv1 and NYUv2. Furthermore, the proposed Bayesian fusion exploiting pixel context could be applied also to other single-view methods in order to improve them and build semantic maps on their basis. As a proof, we extended our single-view approach 3DEF combining it with our novel semantic segmentation method based on the YOLO object detector and the Grabcut segmentation showing that state-of-the-art performances can be reached. In the future, we would like to further work on the most recent and accurate dataset presented in [118], as well as our own COROMA dataset. We think that the potential of the online method has not been fully demonstrated. The availability of a larger dataset with greater variability in depth and angulations will help us finding more effective ways to fuse the single-view results, for example other priors or a distance/angulation-based weight system could be considered to improve the bayesian fusion. We plan to install it on the robots presented in the Appendices A and B in order to foster human-robot interaction. Finally, it would be interesting to extend it so as to handle many other class labels. The next chapter goes in this direction by applying techniques similar to those presented in this chapter to the problem of detecting people also in unconventional poses, in particular fallen people lying on the floor for elderly assistance.

3

Semantic Segmentation and Mapping for Detecting Fallen People

Semantic segmentation and mapping techniques can handle very challenging scenarios with objects characterized by varying appearance, populating cluttered scenes and potentially occluded. In this chapter, taking inspiration from these techniques, we study how to address the problem of detecting fallen people lying on the floor in a novel and effective way. This is an important application for home robots, which are making their way into our homes. New products like Softbank's Pepper have been introduced into the market and many research platforms, e.g. the healthcare robots Pearl [119], ASTRO [120], Max [121], Hobbit [122] or our prototype O-Robot [103] presented in Appendix A, have been proposed. Not only such robots aim at fostering research to keep the house safe by monitor-

ing and detecting anomalies, but also at being friendly companions able to enhance the elderly people's social lives without invading their privacy. One of the possible applications they should be prepared to is the detection of fallen people. Indeed, in the richest countries, the population is getting older [123] and, among all the sources of harm, falls are known to be the major one in elderly people [124].

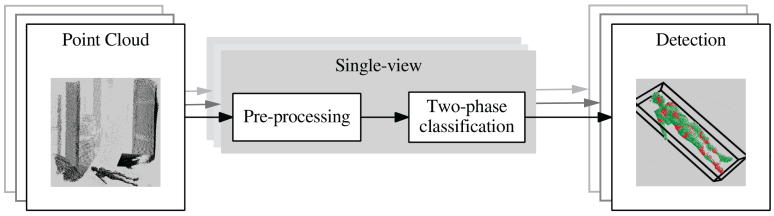
In this chapter, given that it is unlikely for a robot to capture the act of falling while patrolling, the focus is on detecting people already lying on the floor. Our main contributions are:

- a real-time pure-3D approach to detect fallen people suitable for real cluttered scenes;
- its integration with two basic robot functionalities, 2D mapping and navigation, in order to suppress false positives thanks to the a-priori knowledge of the environment and the availability of multiple view points;
- our RGB-D dataset of fallen people[1] consisting of several static and dynamic sequences with 15 different people acquired in 2 different environments.

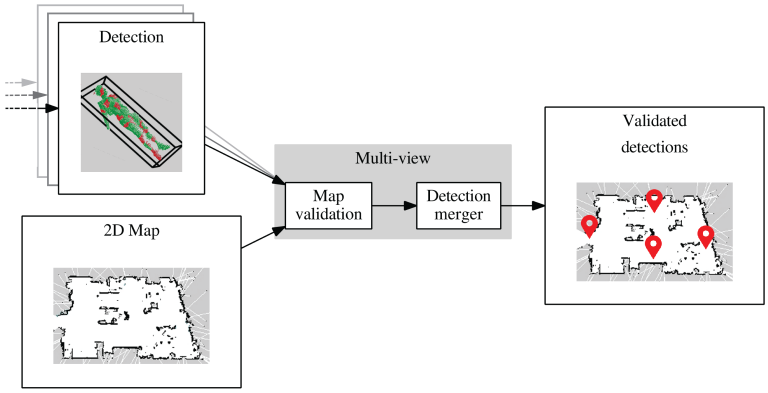
These methods are kindred to those presented in the previous chapter because, on the one hand, the single-view approach is inspired by semantic segmentation techniques in order to handle cluttered scenes not addressed by previous works. On the other hand we show that the existence of a scene model, in this case in form of 2D map, and the availability of multiple points of view are crucial to be robust and avoid annoying false alarms.

The remainder of the chapter is organized as follows. Section 3.1 describes our novel approach, first giving a picture of the entire workflow, then focusing on both the single-view approach (Section 3.2) and its integration with mapping (Section 3.3) and robot navigation (Section 3.4). In Section 3.5, our dataset is described and our methods thoroughly evaluated. Finally, in Section 3.6, our main achievements are recapped and future directions of research identified.

[1] http://robotics.dei.unipd.it/117-fall



(a) The *single-view detector*.



(b) The *multi-view analyzer*.

Figure 3.1.1: The proposed approach is split into two separately running processes. The *single-view detector* detects fallen people on the single frames in a way which proves to be fast and robust to clutter. The *multi-view analyzer* fuses the single-view results exploiting the availability of the 2D map and the multiple points of view explorable during the robot navigation. The final map includes also the semantic information about the location of the fallen people, see the red placeholders.

3.1 Approach

An overview of the proposed approach for detecting people lying on the floor is given in Figure 3.1.1(a)(b). It is decoupled into two separately running processes, the *single-view detector* (Figure 3.1.1(a)) and the *multi-view analyzer* (Figure 3.1.1(b)). The former process, the *single-view detector*, operates on pure-3D Point Clouds generated by a RGB-D sensor such as the Kinect One V2, which, in our experiments, is mounted on a mobile robot 1.16 m off the floor and parallel

to it. Major details on the robot setup are reported in Appendix A. First, the input cloud is preprocessed in order to restrict the subsequent phases to work on a region of interest comprehending all the objects above the floor and below a maximum height. Then, the pre-processed cloud is over-segmented into small patches of voxels with similar appearance. In a two-phase classification step, the patches are classified as part of person or not and gathered together. The use of the Euclidean clustering on the cloud including only the person patches makes it possible to handle also cluttered scenes. Finally, to further improve performances, the latter process, the *multi-view analyzer*, rejects all the detections not belonging to the free space of the 2D map and accumulates the detections from several frames by taking into account their 2D map positions and timestamps. Each phase is deeply discussed in the next sections: Section 3.2 deals with the description of the *single-view detector* while Section 3.3 and Section 3.4 describe the *multi-view analyzer*.

3.2 Patch-based Detection of Fallen People

Each point cloud is pre-processed to restrict the analysis to a region of interest and reduce the data noise. First of all, the point cloud is truncated to a 3D region containing the floor and the points between it and a maximum height of about 0.7 m. Then, the floor is removed with an approach based on the RANSAC segmentation [92]. To improve its robustness to the robot motion, two floor planes are estimated, on a first half of the cloud close to the robot and on a second half far from the robot. In particular, a good split distance proved to be 3 m. Finally, to reduce the data noise without affecting the running time, a soft statistical outlier removal is applied with the number of neighbours set to 50 and the standard deviation set to 0.3.

The core of the algorithm draws upon two recent works about the semantic segmentation of objects and scene structures [43, 44], which anyway do not face with the problem of segmenting people. It comprehends the following 4 phases:

1. supervoxel over-segmentation in 3D patches;

2. classification of each 3D patch as positive, i.e. part of a fallen person or negative, i.e. not part of a fallen person;

3. clustering of positive patches;

4. classification of each cluster as positive, i.e. a fallen person, or negative, i.e. not a fallen person.

In contrast to previous works on the detection of fallen people [68], they allow to segment and classify correctly also the people lying close to other objects or structural elements since they do not rely on fine distance thresholds. Moreover, with respect to [43, 44], humans are considered and the last two phases are introduced. They replace the region growing step and the 3DEF classifier in [44], already presented in the previous chapter. This is necessary because the region growing is risky: even if it may lead to more descriptive planar patches, it may cluster fallen person patches to bigger non-planar patches, on which geometric features are less descriptive. In the following, each phase is described.

The pre-processed point cloud is over-segmented into homogeneous 3D patches by means of the Voxel Cloud Connectivity Segmentation (VCCS) [106]. An example of over-segmented cloud is reported in Figure 3.2.1. This solution preserves the edges by finding patches not crossing object boundaries and, at the same time, it reduces the noise and the amount of data. The set of parameters used here is: voxel resolution 0.06 m, seed resolution 0.12 m, color importance 0.0, spatial importance 1.0 and normal importance 4.0. The voxel resolution is a good trade off between speed and having a sufficient number of points per patch. The seed resolution is a good trade off between having big patches and over-segmenting also the thinner body elements, e.g. arms and legs. The others are suggested in [125]. As the proposed approach does not rely on RGB data, color is not considered at all by setting the color importance to 0.0.

For each patch generated by the over-segmentation, a feature vector $\mathbf{x}_i$ of length 16 is calculated. The choice of the features is based on the semantic segmentation works [43, 44], whose presented features proved to be as fast as ef-

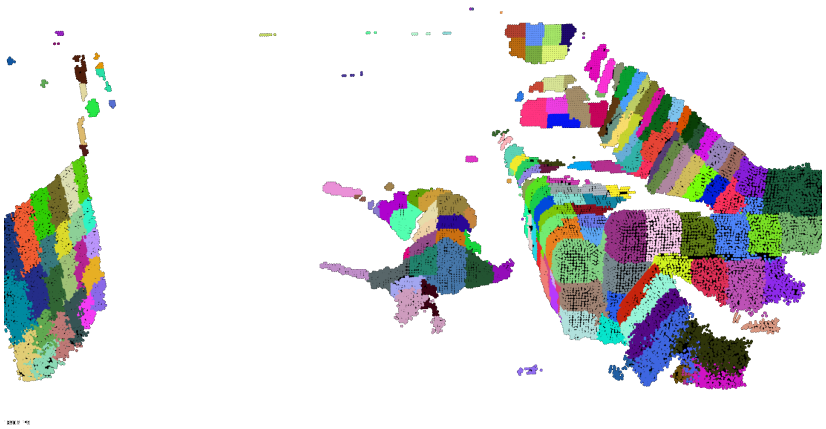
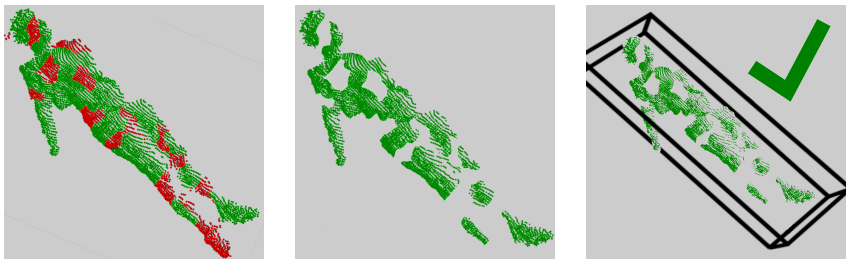


Figure 3.2.1: An example of pre-filtered and over-segmented cloud. A random color is assigned to each patch. The person is lying in the center.

fective. Here, the color features are left out and only the geometric features are taken into account. Some of them are calculated from the eigenvalues of the scatter matrix of the patch, which represent the variance magnitudes in the main directions of the spread of the patch points. In increasing order of magnitude, they are: $\lambda_0 \leq \lambda_1 \leq \lambda_2$. Others are calculated from the Oriented Bounding Box (OBB) including all the patch points. The complete list is given in Table 3.2.1. To calculate the predicted label (part or not part of a fallen person) for each patch, this feature vector is then passed to a binary SVM classifier. After grid search and k-fold validation, a Radial Basis Function (RBF) kernel with the misclassification cost C equal to 62.5 and the bandwidth γ equal to 0.51 turned out to be the best performing solution. Of course, having each patch classified as part of a person (positive) or not (negative) does not suffice to detect a fallen person. Indeed, as shown in Figure 3.2.2(a), given that this classifier analyses just small patches, there can be false positives and false negatives. Because of this, two further steps, explained in the next lines and sketched in Figure 3.2.2(b)(c), have been developed in order to find 3D regions with a high density of positive patches and whose size is comparable to that of a person.

Table 3.2.1: List of features calculated for each 3D patch and their dimensionality.

Features	Dimensionality
Compactness (λ_0)	1
Planarity ($\lambda_1 - \lambda_0$)	1
Linearity ($\lambda_2 - \lambda_1$)	1
Angle with floor plane (mean and std. dev.)	2
Height (top, centroid, and bottom point)	2
OBB dimensions (width, height and depth)	3
OBB face areas (frontal, lateral and upper)	3
OBB elongations ($\frac{height}{width}$, $\frac{depth}{width}$, $\frac{height}{depth}$)	3
Total number of features	16



(a) **(b)** **(c)**

Figure 3.2.2: The last three steps of the algorithm core: a) The first SVM classifies each patch as a person part (green color) or not (red color); b) Euclidean clustering of the positive patches; c) Calculation of the cluster OBB. The second SVM classifies each cluster as a person or not. Here, the response is positive.

In contrast to the methods in [68], having two sets of patches respectively with the positive and negative ones opens up the possibility to apply the Euclidean cluster extraction without the risk of segmenting a fallen person together with the adjacent scene elements. First of all, some false positive patches can be easily recognized, e.g. all the patches with less than 5 neighbouring positive patches in a radius of 0.5 m can be filtered out. Then, the negative patches are pushed aside, and

Table 3.2.2: List of features calculated for each 3D cluster and their dimensionality.

Features	Dimensionality
OBB dimensions (width, height and depth)	3
Number of positive patches	1
Percentage of positive patches	1
4-bin histogram of positive patch confidences	4
Total number of features	9

the Euclidean clusters are extracted from the point cloud of the remaining positive patch centroids using a large distance threshold of 1.0 m.

For each cluster, its OBB is calculated. Thus, depending on the OBB dimensions and the number of positive and negative patches in it, each cluster may be a fallen person or not. For each cluster, a feature vector $\mathbf{x}_2$ of size 9 has been devised. The complete list of features is given in Table 3.2.2. In particular, the sample distances to the separating hyperplane returned by the former SVM turned out to be really useful. They have been exploited by means of an histogram with 4 bins for the distance intervals $[0, 0.25)$, $[0.25, 0.5)$, $[0.5, 1)$ and $[1, \infty)$. For each cluster, each histogram bin is filled with the positive patches whose distance to the hyperplane falls in the respective interval. Thus, the number of positive patches in each bin/interval gives 4 additional features. Doing so also for the negative patches turned out to be counterproductive. The whole feature vector is passed to a binary SVM classifier. After grid search and k-fold validation, a RBF kernel with the misclassification cost C equal to 312.5 and the bandwidth γ equal to 2.25×10^{-3} turned out to be the best performing solution.

3.3 Map Verification

A mobile robot navigates through the environment thanks to the information of two maps: a static one necessary to compute a collision-free plan with static ob-

jects, e.g. walls or furniture items, and a dynamic one necessary to avoid moving obstacles, e.g. people. In this work, the static map, which is usually acquired only once and for all with a 2D SLAM algorithm like [126, 127], is exploited to implement a false positive rejection phase. If necessary, movable elements like chairs or other small furniture elements can be easily left out so that just walls and large furniture elements can be present. This way, true positives cannot be located on dynamic elements erroneously present in the static map and will not be rejected. Let the static map be defined as a set of cells $S = \{Cell_i, 0 \leq i \leq N\}$, where:

$$Cell_i = \begin{cases} -1 & \text{unknown content} \\ 0 & \text{free space} \\ 0 < n \leq 1 & \text{probability to be occupied} \end{cases}$$

Thanks to the transformations computable with a localization algorithm like the Adaptive Monte Carlo Localization (AMCL) [128], each single-view detection can be transformed from the camera coordinate system to the map coordinate system and projected to a cell map $Cell_i$. If the $Cell_i$ value is unknown (-1), i.e. it is out of the room map perimeter, or occupied by a static obstacle $(K \leq Cell_i \leq 1$ with $K = 0.30)$, then the detection can be easily rejected. An example of successful false positive rejection is shown in Figure 3.3.1, in which a single-view detections falls on the static furniture, in this case a tree trunk. Indeed, given its geometric similarity to a lying person, the single-view algorithm may detect it as a person. The map verification allows to reject it, enhancing the final detection performances. This step handles also other challenging situations, like glass pieces of furniture which can be really noisy.

(a) (b)

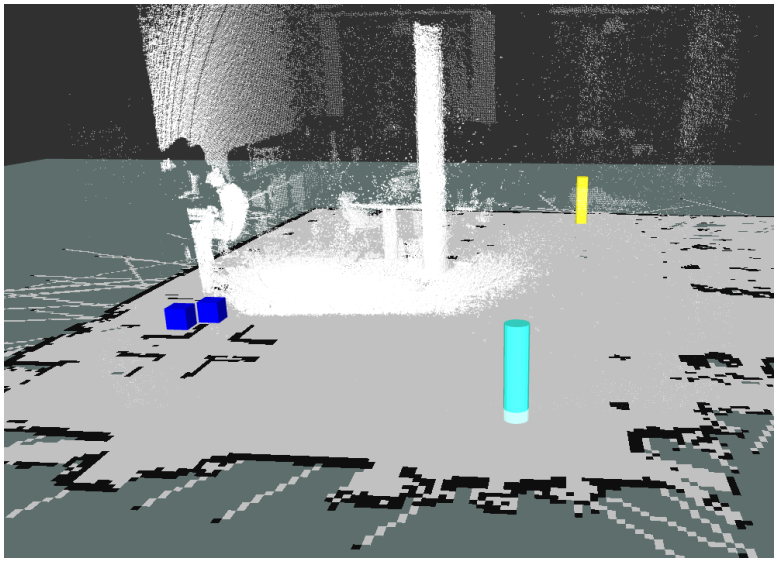
Figure 3.3.1: An example of successful false positive rejection performed by the map verification step: a) shows the furniture item raising some false positives, a tree trunk (in green) b) shows that these detections (the blue squares on the right) are located onto the map occupied space.

Algorithm 1 Cluster validation for detecting fallen people exploiting multiple vantage points

procedure VALIDATE_CLUSTERS($\mathfrak{C}, \mathfrak{P}, \widehat{t}, \widehat{f}, \widehat{n}$)
  for each $C_i \in \mathfrak{C}$ **do**
    for $j \in [0, k-2]$ **do**
      for $o \in [j+1, k-1]$ **do**
        if $|d_o.t - d_j.t| > \widehat{t}$ **then**
          $index \leftarrow$ ARG_MIN$(d_o.t, d_j.t)$
          $C_i \leftarrow C_i \setminus d_{index}$
    $t_m \leftarrow \min_{d \in C_i}\{d.t\}$
    $t_M \leftarrow \max_{d \in C_i}\{d.t\}$
    $f_i \leftarrow \frac{||\mathfrak{C}||}{t_M - t_m}$
    if $f_i \geq \widehat{f}$ and $||C_i|| \geq \widehat{n}$ **then**
      $loc_i \leftarrow \sum_{d \in C_i} \frac{d.loc}{||C_i||}$
      $\mathfrak{P} \leftarrow \mathfrak{P} \cup loc_i$
      $\mathfrak{C} \leftarrow \mathfrak{C} \setminus C_i$
  return $\mathfrak{C}, \mathfrak{P}$

3.4 Merging Detections from Multiple Vantage Points

Mapping is not the only robot capability that can enhance the detection performances. Indeed, in a typical scenario, the robot is patrolling a known environment.



(a)

Figure 3.4.1: The single-view detections projected on the 2D map are analysed by the *multi-view analyzer*. If they meet both the distance and time criteria, they are clustered. The white points compose the input point cloud, the blue cubes are the projected detections, here rejected false positives, and the randomly coloured cylinders are the validated detection.

Thus, given that the location of each fallen person is mostly static, all the single-view detections available from the multiple points of view can be easily tracked. A detection may be a false positive from a certain view, while a true negative from many others. Moreover, the false positive detection rate is very low compared to the true positive one. Given these two facts, another contribution of this work is the exploitation of the detections available from the different vantage points.

After the map verification, the single-view detections are already expressed in the map reference system. In this section, an algorithm able to cluster or reject each of them is devised. Its output is a set $\mathfrak{P}$ of validated lying person locations p_i in the map, formally $\mathfrak{P} = \{p_i, 0 \leq i \leq g\}$, where g is the total number of people. Given each new detection $d = (loc, t)$, where $d.loc$ is its location in the map coordinate system and $d.t$ its timestamp, the set of clusters, formally $\mathfrak{C} = \{C_i : 0 \leq i \leq n\}$,

is updated with the following rule:

$$C_i = \{d_j : ||d_j.loc - d_m.loc|| < \widehat{th}, \forall j, m \in [0, k-1]\},$$

in which $\widehat{th}$ is a user-defined threshold which indicates if a detection is close enough to be considered in the cluster or not, and k the number of detections in the cluster.

The set $\mathfrak{P}$ of fallen people is computed by a fixed-time periodic thread which analyses the set $\mathfrak{C}$. It updates the set $\mathfrak{P}$ by deleting the old detections and analysing the new ones in $\mathfrak{C}$. Indeed, in order to maintain a lightweight representation of $\mathfrak{C}$ and reject the false positives, whose frame rate is typically low, the old detections are discarded and a further check on the timestamp is performed. The pseudo-code of the whole procedure is reported in Algorithm 1, in which $\widehat{f}$ is the minimum detection frequency, $\widehat{t}$ is the maximum detection age and $\widehat{n}$ is the minimum number of detections in a cluster. Lines 3-7 handle the time-based rejection on the basis of the maximum allowed age, while Lines 8-14 reject the clusters whose detections have a low frame rate or are less than the minimum allowed.

In our implementation, we used $\overline{th}$ equal to 1 m, $\widehat{t}$ equal to 60 s, $\widehat{f}$ equal to 1 Hz and $\widehat{n}$ equal to 5. The use of the frame rate allows to set a low $\widehat{n}$, thus preventing over-fitting. The procedure is invoked by the periodic thread every 10 seconds. In Figure 3.4.1, the algorithm is shown in action.

3.5 Experimental Results

The detection of fallen people is a challenging problem also because of the lack of public datasets. For this reason, another contribution of this work is the release of the IASLAB-RGBD Fallen Person Dataset[2]. On it, 4 common metrics, the detection *accuracy*, *precision*, *recall* and $F_{0.5}$ score, are evaluated for each presented method. If *TP*, *TN*, *FP* and *FN* are the true positives, true negatives, false positives

[2] http://robotics.dei.unipd.it/117-fall

and false negatives, then these metrics are defined as in the following:

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F_{0.5} = \frac{(1 + 0.5^2) * precision * recall}{0.5^2 * precision + recall},$$

where the $F_{0.5}$ score, already proposed in [68], is an harmonic average of precision and recall promoting an high precision, i.e. a low number of false positives. In addition, given the impossibility to compare with other existent and similar approaches, the baseline to which our algorithms are compared is a simple approach based on the Euclidean cluster extraction. This way, it will be clear how important the use of patches is in order to handle cluttered scenes. Finally, a detailed analysis of the running times is provided.

3.5.1 IASLAB-RGBD Fallen Person Dataset

This dataset consists of several RGB-D frame sequences containing 15 different people. It has been acquired in two different laboratory environments, the *Lab A* and the *Lab B*, by means of a Microsoft Kinect One V2, placed on a pedestal or on our mobile robot. The *Lab A* is bigger and useful to test whether the algorithm can find people in the full distance range of the sensor (up to 5 m). The *Lab B* is smaller and more similar to a real domestic scenario. It is more cluttered and contains a sofa. It comprehends also glass surfaces which can be very noisy with time-of-flight sensors like the Kinect One V2. For the sake of explanation, the dataset can be divided into three parts:

1. Part 1 includes 360 RGB-D frames acquired from 3 static pedestals. It is composed of several views of 10 people, which have been asked to lie in 12

different poses, 6 from the back and 6 from the front. Each person has been manually segmented in 3D;

2. Part 2 includes 4 sequences of RGB-D frames, for a total of 15932 frames, acquired from a mobile robot during its patrolling task in the *Lab A*. People lie in 4 different fixed locations;

3. Part 3 includes 4 sequences of RGB-D frames, for a total of 9391 frames, acquired from a mobile robot during its patrolling task in the *Lab B*. People lie in 4 different fixed locations.

Training and test splits are also available. Some images of the dataset will be reported when discussing the results even if our approach does not exploit the RGB info.

The first classifier of the *single-view detector* has been trained/validated on thousand of patches extracted from the frames in Part 1 and Part 2 and tested on patches extracted from the frames in Part 1 and 3. This way, the samples composing training and test sets belong to different people or scenes. All the positive samples have been taken from Part 1. The 70-30 train-test split of the segmented fallen people in Part 1 is also available. Negative samples have been taken from the *Lab A* (just 24 frames out of 15932, for training), the *Lab B* (just 32 frames out of 9391, for testing) and the NYU Depth Dataset V2 [31] (just 35 out of 1449, enough to extend the training set size), which contains thousands of indoor scenes for scene understanding. Only some of the negative samples have been used for balancing the number of positive and negative samples. Finally, given the relatively limited size of the dataset, another split has been considered to evaluate performances: the classifier has been trained on Part 1 and 3, and tested on Part 2. Analogously, the second classifier of the *single-view detector* has been trained/validated on positive clusters extracted from the frames in Part 1 and negative clusters extracted from the frames in Part 2 (*Lab A*) and tested on clusters extracted from the frames in Part 3 (*Lab B*). Finally, another split has been considered: it has been trained on Part 1 and 3, and tested on Part 2.

Not only the *single-view detector* but also the *multi-view analyzer* has been tested on Part 3. Indeed, both Part 2 and 3 comprehend the entire robot transformation tree. Given that the position of the fallen people in the 2D map is known, this allows to calculate the performance indices automatically by checking if the location of the detected cluster centroid is close (at a distance less or equal to 1 m) to the ground truth centroid of a person position in the 2D map.

3.5.2 Validation

The presented methods have been quantitatively evaluated on the IASLAB-RGBD Fallen Person Dataset. First, the separated evaluation of each classifier is presented. Then, the entire pipeline has been evaluated on both rooms, the *Lab A* and the *Lab B*. In particular, we present the results for each of the 3 contributions, the *single-view detector* and the two modules of the *multi-view analyzer*: the map validation and the detection merging from multiple vantage points. Furthermore, given the impossibility to compare directly with [68], the comparison baseline (B) is a simple approach not exploiting patches. It finds putative clusters by means of the Euclidean cluster extraction with a distance threshold of 0.10 m, which is really low considering a voxel resolution of 0.06 m and far less than the one required by our approach (1 m). The baseline predicts a label for each cluster by checking the height of the centroid position and by feeding its OBB size to a Random Forest classifier. We chose this approach for two reasons. First, thanks to the low distance threshold, it can find a large number of clusters. Second, since each cluster is classified from its size only, this method can lead to a large number of true positives.

For both classifiers, grid search with K-fold cross-validation has been performed on the training/validation set in order to find the optimal misclassification cost C and bandwidth γ values of the RBF kernel. The parameter C has been varied in the interval $[0.1, 500]$ with logarithmic step 5. The parameter γ has been varied in the interval $[10^{-5}, 0.6]$ with logarithmic step 15. The parameter K has been set to 10. As a preliminary evaluation, the SVM performances on the respective test sets

are reported in Table 3.5.1. They are the averaged on the two available splits.

Table 3.5.1: Performances of the two classifiers on their test sets. They are averaged on the two splits. The standard deviation on $F_{0.5}$ is reported.

Method	Accuracy	Precision	Recall	$F_{0.5}$
Classifier 1 (C1)	0.90	0.87	0.86	0.87 ± 0.06
Classifier 2 (C2)	0.96	0.97	0.96	0.97 ± 0.02

In the following, the first split is considered. The results of the quantitative comparison between all the methods are shown in Table 3.5.2 and 3.5.3. Thanks to the patches, our methods outperform the baseline, not only in precision but also in recall. Furthermore, the map validation can further improve performances by rejecting some false positives. It also causes a small detriment in recall since the map validation may erroneously discard some true positives.

Table 3.5.2: Performance comparison on the *Lab A*.

Method	Accuracy	Precision	Recall	$F_{0.5}$
Baseline (B)	0.88	0.65	0.33	0.54
Single-view (SV)	0.87	0.73	0.85	0.75
SV+Map verification (MV)	**0.92**	**0.85**	**0.85**	**0.85**

Table 3.5.3: Performance comparison on the *Lab B*, never seen before by both classifiers.

Method	Accuracy	Precision	Recall	$F_{0.5}$
Baseline (B)	0.84	0.64	0.26	0.50
Single-view (SV)	0.88	0.80	**0.86**	0.81
SV+Map verification (MV)	**0.90**	**0.87**	0.81	**0.86**

As shown in Table 3.5.4, also the detection merging from multiple vantage points proved to be useful. It has been tested on each one of the eight frame sequences acquired in the two environments. Each time, even if the environment is the same, the navigation path can differ slightly due to dynamic obstacles like people and the different positions of the lying people on the floor. After the 4 patrolling tasks of the *Lab A*, each person is always detected and only once, a false positive is still present due to the trunk shown in Figure 3.3.1 while, after the 4 patrolling tasks of the *Lab B* (never seen before by both classifiers), each person is always detected and all the false positives are successfully rejected.

Table 3.5.4: Performances of the *multi-view analyzer* on both environments. Each time, even if the environment is almost the same, the robot path can differ because of dynamic obstacles and different positions of the lying people on the floor.

Environment	TP/P	FP
Lab A (sequence 1)	4/4	0
Lab A (sequence 2)	4/4	1
Lab A (sequence 3)	4/4	0
Lab A (sequence 4)	4/4	0
Lab B (sequence 1)	4/4	0
Lab B (sequence 2)	4/4	0
Lab B (sequence 3)	4/4	0
Lab B (sequence 4)	4/4	0

Finally, in Figure 3.5.1, some qualitative results are reported. They show the ability of the *single-view detector* to find people in cluttered environments, see Figure 3.5.1(a)(b)(c)(d). Two difficult cases due to close objects or noisy regions, like glass surfaces, are also reported, see 3.5.1(e)(f). Anyway, they are easily handled by the *multi-view detector*, 3.5.1(g)(h). A video is attached to demonstrate the quality of the system[3].

[3] https://www.youtube.com/watch?v=-lnqXTvH69I

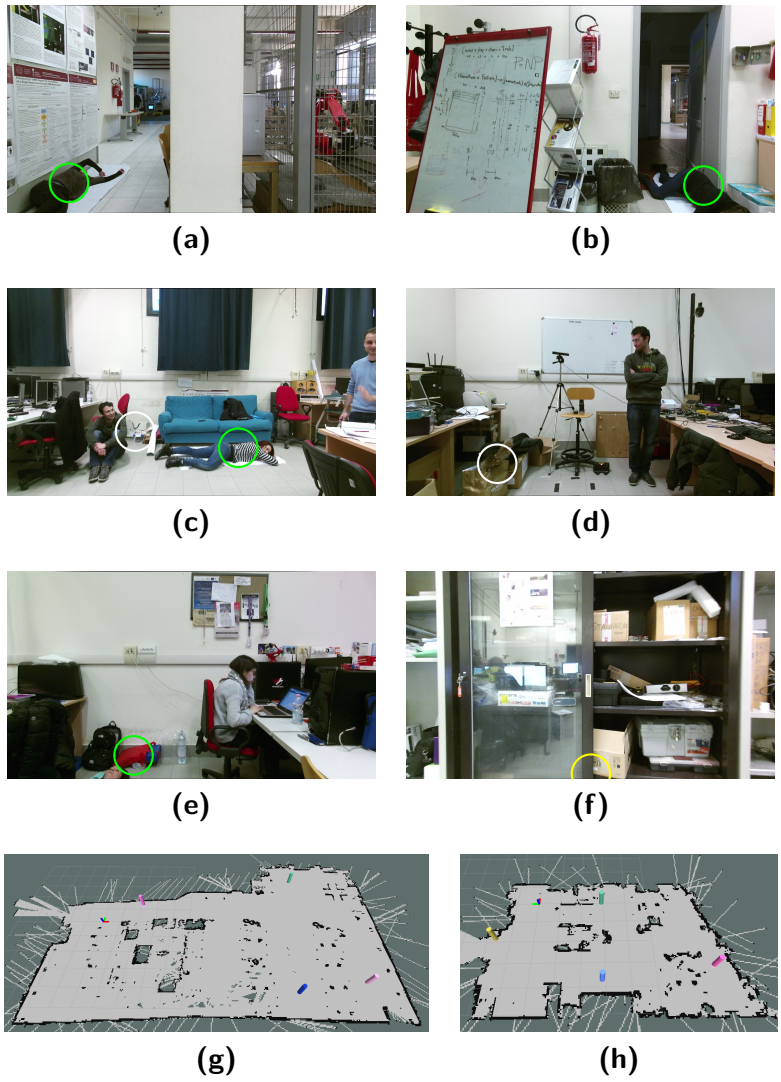


Figure 3.5.1: Qualitative results on the IASLAB-RGBD Fallen Person Dataset: (a)(b) even if the lying people can be very close to the wall or other scene elements, the *single-view detector* can find them at a high detection rate; (c)(d) the *single-view detector* can discard fake lying people, see the white circles; (e)(f) the *single-view detector* may find some false positives in the presence of clutter (several close objects) or high noise (glass surfaces); (g)(e) the *multi-view analyzer* can reject both FP like in (e) thanks to the low frame rate or in (f) thanks to the map validation (yellow circle).

3.5.3 Runtime Analysis

In Table 3.5.5, the running times of *single-view detector* are reported. The algorithm is very efficient in terms of computing time proving to be an optimal choice for a mobile robot. Even if it is not yet fully parallelized, it can work in real-time at an average speed of 7.72 fps. The test machine is a Dell Inspiron 15 7000 with an Intel Core i7-6700HQ CPU with 4 cores clocked at 2.60GHz, 16 GB of DDR3 RAM and Linux Mint 17.3. Given that the *multi-view analyzer* is a daemon running in the background, its running times are of no interest and thereby not reported.

Table 3.5.5: Average runtimes of the main steps of the proposed algorithm on our test machine (Intel Core i7-6700HQ CPU, 2.60GHz $\times$ 4).

Processing Stage	Runtime
Pre-processing and Oversegmentation	10.27 fps
Patch Feature Extraction	105.98 fps
SVM Classification 1 (per patch)	0.84 μs
Cluster Feature Extraction	2639.56 fps
SVM Classification 2 (per cluster)	0.04 μs
Total runtime	7.72 fps

3.6 Summary

This chapter presented a real-time and robust approach to detect fallen people lying on the floor in various positions and from different distances. A single-view algorithm, which draws upon recent developments in the semantic segmentation field and does not need restrictive distance thresholds to segment putative clusters, was fully integrated on a mobile robot. The map of the environment and the availability of many different vantage points allowed to reduce the number of false posierencetives, further improving the final performances. The algorithms presented here were thoroughly validated on the IASLAB-RGBD Fallen Person

Dataset, which was published online for the benefit of the research community. They clearly outperform a simple method based on a finer distance threshold. In the near future, we would like to validate not only the ability of the algorithm to detect, but also to semantically segment fallen people. We also plan to extend the test bed with sequences taken from real apartments along with different navigation paths. It would be interesting to merge close similar patches before their classification and to improve the multi-view method by making it independent from the running time of the single-view method.

Up to this point, cutting-edge machine learning techniques were applied to the problem of segmenting and classifying objects, scene structures and humans in domestic applications. In the following chapters, two industrial applications are presented. They deal with the problem of inspecting Carbon Fiber Reinforced Polymers in order to build semantic models out of them, respectively with the carbon fiber arrangement and defects. In both cases, two specific approaches are proposed in order to handle the specific challenges. The differences but also commonalities between domestic and industrial applications will be discussed too. Indeed, the construction of semantic models of scenes and objects has several common points, e.g. the fusion of multiple views by means of the framework introduced in Section 1.2 and the importance of contextual cues.

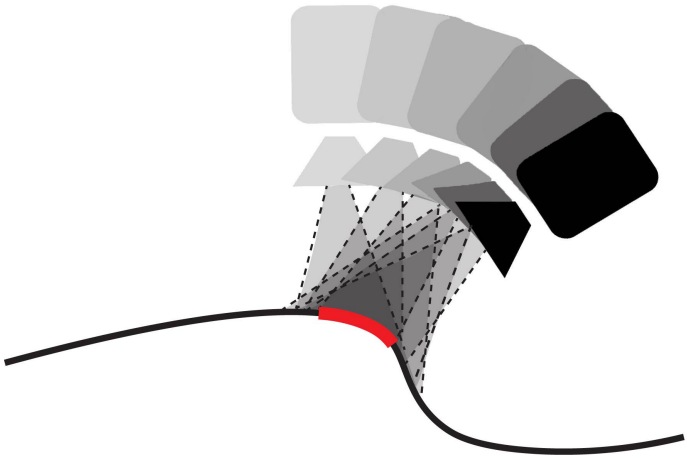
4

Continuous Segmentation and Mapping of Carbon Fibers to 3D Models

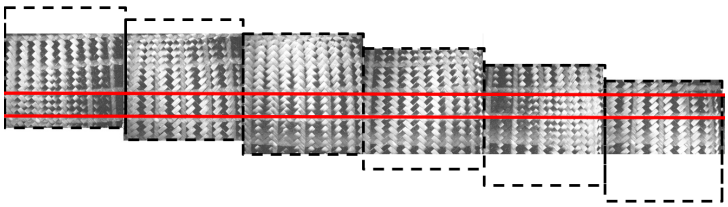
In previous chapters, we have presented our first studies to build semantic models of scenes, in particular how to segment objects, scene structures and humans showing that context and the multiple points of view play important roles to improve over the single-view performances. Again, in this chapter, context and the different points of view will be exploited to improve performances and fully inspect the part. Nevertheless, we will focus on robots exploring object surfaces instead of their surrounding. We will build semantic models of objects enriching their model with surface properties, further working on the back-projection and forward-projection systems introduced in Section 1.2 and developed in Section 2.3.

Over the last few years, machine vision has become a fundamental tool for automating quality inspection and enhancing industrial manufacturing processes. Even if most of these techniques rely on the individual analysis of single shots, sometimes multiple shots are needed for observing the features of interest or to simply obtain a better observation/measurement. As an example, photometric stereo [129], instead, can measure shape at every pixel by combining images of the same object taken with illuminations from different directions. Also texturing applications [130] can benefit from multiple images acquired with different illuminations, since images can be combined for performing shadow and reflection removal. One drawback of multi-image measurement is that a longer time is necessary for acquiring all the needed images, thus before being able to produce the measurement. If the sensor can acquire just a small portion of the object for every frame, the sensor must be moved all over the part, e.g., with a robotic manipulator. However, as sketched in Figure 4.0.1, since each set of multiple images should refer to exactly the same portion of the scene, a stop-and-go scanning process that stops the sensor every time it has to take a set of photos is usually required, but the consequent long scanning times can make impractical the use of these technologies in industrial context.

This chapter provides methods for enabling the continuous motion of sensors in multi-image measurements, thus guaranteeing shorter scanning times and a measurement quality similar to that obtained with static measurements. In particular, it describes solutions to the problem of image registration and mapping of the measurements to a 3D model of the scanned object. These contributions already led to a ground-breaking application for fast mapping of carbon fiber orientation with high accuracy. This is done by means of an inspection robot equipped with a sensor that exploits a technique similar to photometric stereo for generating 3D carbon fiber orientation at every pixel [88]. This chapter will focus on this application for better explanation, but the approach and the methods of image registration and 3D mapping are general and can be applied also to other sensors and other application scenarios once the object model and position are known.



(a) A sensor moving over a 3D shape while capturing images. Only the part marked in red is visible from all the points of view.



(b) Example of six consecutive images of a carbon fiber part acquired with different light sources with a sensor in continuous motion. Only the part in the red box is visible from all the points of view.

Figure 4.0.1: Sketch of the problem addressed in this chapter.

Increasing use of Carbon Fiber Reinforced Polymers (CFRPs) is required whenever high strength-to-weight ratio and rigidity are necessary, as it happens in some application fields like automotive, aerospace and civil engineering, sports goods and military applications. Carbon fiber is made of long strings of carbon atoms that can be woven together to form sheets, which can be layered onto each other. The mechanical strength of these parts depends on how fiber angles are arranged all over the product. For this reason, during the design of carbon fiber

preforms, much time is spent optimizing the process and mechanical parameters according to the expected loads that will later act on the part ([131], [132]). An ideal process would then require to check the quality of the produced preform and provide direct feedback to the simulation for correcting the production process. However, this kind of loop is not implemented in current production systems, because:

- there was no method of acquiring dense fiber angle measurements for complex 3D parts. Measurements are often done manually, where accurate coordinate systems and reference lines are hard to establish and it is not possible to obtain a dense mapping of the part;
- this lack of accurate fiber angle data is also hindering further development and improvement of draping simulation, that would require to tune simulation models by comparing predictions to reality.

The lack of technologies for accurate fiber angle measurement and prediction thus led to unnecessarily high safety factors in the amount of used material (and thus weight), which could be reduced if the fiber angles on the surface could be accurately predicted and measured. Last, but not the least, design parts also need to be checked for not containing visible defects.

The methods and results presented in this chapter led to the development of an automatic quality control and feedback mechanism to improve draping of carbon fibers on complex parts. The main contributions are:

- a novel image registration method that takes into account the motion of the sensor in its 6 Degrees of Freedom (6DoFs), the object 3D surface and position. It allows to perform fiber segmentation and multi-image measurements with a moving sensor while guaranteeing a measurement quality close to that of static measurements and shorter scanning times;
- a novel system for high accuracy mapping of carbon fiber orientation with an inspection robot, tested on several carbon fiber preforms of increasing complexity;

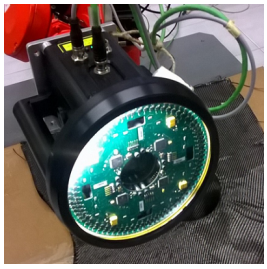
- a highly optimized rasterizer exploiting the SSE2 vectorization for real-time mapping of 2D measurements to 3D models of the scanned part;
- a fully automatic hand-eye calibration procedure, which does not require the knowledge of the calibration pattern 3D location and can be easily exploited on other robot platforms. The source-code is online[1].

The remainder of the chapter is organized as follows. Section 4.1 describes the developed quality inspection robot for carbon fiber parts. Section 4.3, 4.4 and 4.5 describe the proposed registration and mapping methods for multi-image reprojection and measurement in motion, while Section 4.6 reports the main experiments performed for assessing the accuracy of the proposed algorithms for mapping carbon fiber orientation and a frame-rate analysis comparing different methods. Finally, in Section 4.7, our main achievements are recapped and some final remarks reported.

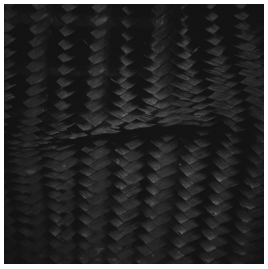
4.1 Quality Inspection Robot for Carbon Fiber Parts

The technology that has been developed in this work includes a new sensor system for fast and robust estimation of fiber orientation and a robotic system to move the sensor to scan large parts. The sensor, displayed in Figure 4.0.2(a), is produced by Profactor GmbH, Austria. It includes the Genie TS-M2500 vision camera, the Schneider XENOPLAN-M 17/1,4 lens and a light source setup for photometric stereo. The industrial robotic arm is the Comau Smart5 SiX, displayed in Figure 4.0.2(d). It is controlled by the C5G control unit, which comprehends 1 B&R APC 820 and 4 B&R ACOPOS. After capturing a sequence of images of a small patch of the part surface to be inspected, fiber angles are calculated and mapped back onto the surface of the virtual CAD model of the object, whose position is known after calibration. A sample grey-scale image acquired by the vision camera is reported in Figure 4.0.2(b). The fiber orientations estimated from an image

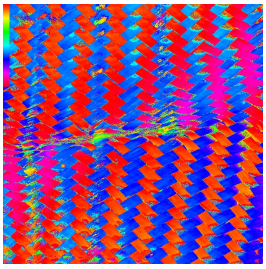
[1] http://robotics.dei.unipd.it/129-auto-hand-eye



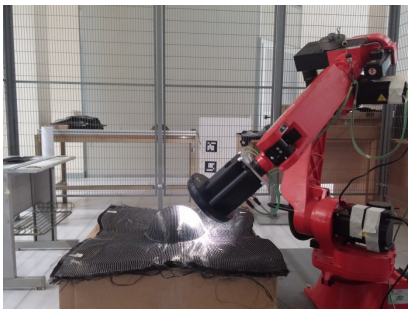
(a) Profactor's sensor.



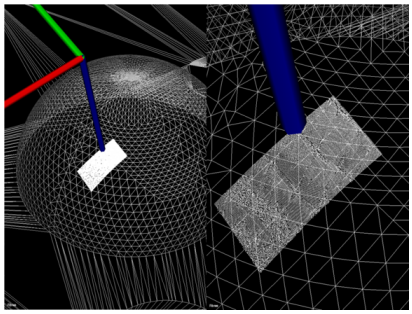
(b) One of the single grey-scale images of the fibers acquired with it.



(c) The estimated fiber orientations in 2D coded with colors.



(d) The robot while scanning a complex carbon fiber part.



(e) The online mapping of the calculated fiber orientations in 3D on the part CAD model.

Figure 4.0.2: This figure illustrates our quality inspection system.

sequence can be represented with colors as in Figure 4.0.2(d) for ease of visualization or with 3D vectors as in Figure 4.0.2(e). All these steps can be performed online thanks to the efficiency of the proposed methods. Then, in a final step, the single scans are fused leading to a 3D CAD model of the part augmented by the 3D fiber orientations.

In the aforementioned pipeline, a number of modules is involved. They are displayed in the flowchart in Figure 4.1.1 and briefly described in the following. As a preliminary step, the work-cell is calibrated with standard procedures, like the hand-eye calibration [133], in order to retrieve the geometrical relations be-

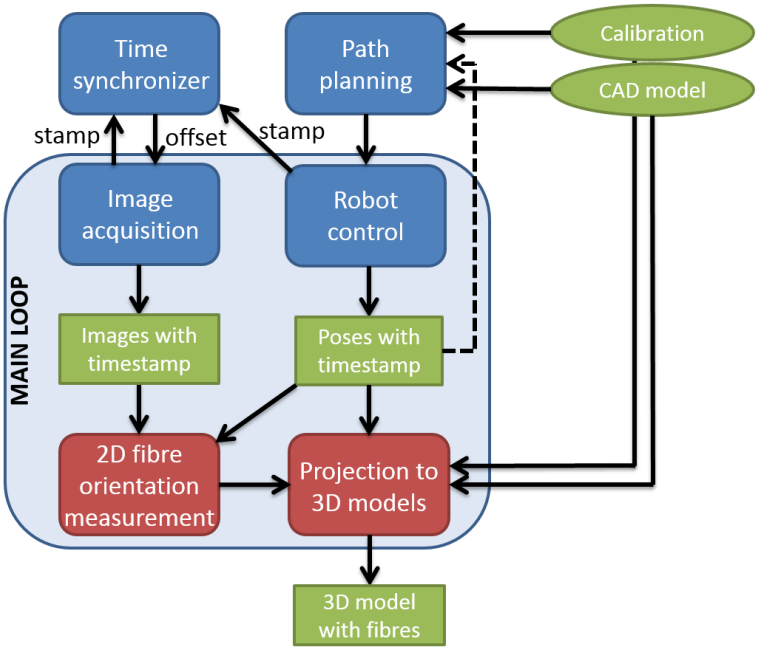


Figure 4.1.1: Workflow of the fiber mapping software. In red, the main contributions of this work.

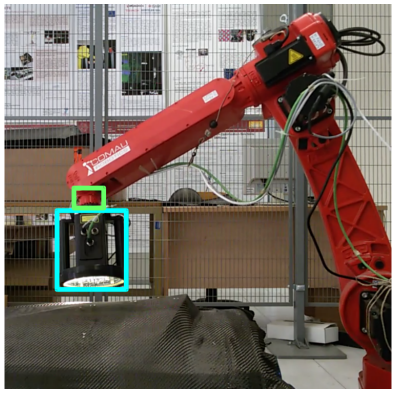
tween the elements in the work-cell: the robot, the sensor and the part. These procedures, which provide input to the coverage planner and projection modules, are mentioned in Section 4.2. As a side contribution, we automatized one of these techniques with the aim of eliminating human intervention. Given a calibrated work-cell, the required robot path to fully cover the part, i.e. gather a suitable set of images such that all the part details of interest are observed, can be calculated. The offline programming software for 3D simulation and motion planning developed by IT+Robotics Srl, Italy [134] is exploited to fully cover also complex shapes avoiding collisions and dealing with robot reachability issues. Not only IT+Robotics solution optimizes cycle time, but it also improves the quality of the scan since it guarantees the sensor perpendicularity with respect to the surface in the field of view. Then, the part inspection can start. Five modules working in parallel play a crucial role: the image acquisition or sensor interface, the robot interface, the synchronizer, the projection to the 3D model and the 2D fiber orien-

tation measurement. The first two are self-explanatory. The synchronizer guarantees that each image is associated with the right robot position. The projection (or mapping) to the 3D model, in Figure 4.0.2(d), and the 2D fiber orientation measurement modules, displayed in red, define the main contributions of this work. In our first work [10], the 2D fiber orientation measurement is sequentially followed by a basic registration algorithm, presented in Subsection 4.3.1, and the projection to the model. In our second work based on the registration algorithm presented in Subsection 4.3.2, these modules are intertwined processes, thus providing increased efficiency. Indeed, as detailed in Section 4.3, the projection module is the basis of a novel image registration algorithm which now accounts for 6DoF sensor motion and 3D surfaces. Given that the projection to the 3D model is part of the registration algorithm, there is no need to perform it again for mapping the 3D model. As we will discuss in Section 4.3, feature-based approaches are not applicable in scenarios where images are collected with very different illuminations and in particular when dealing with carbon fibers.

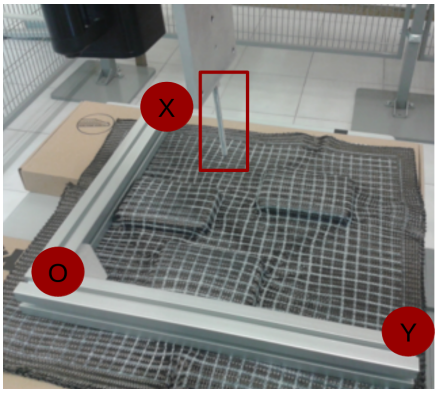
4.2 Work-cell Calibration

The mapping and coverage planner modules require to calibrate the work-cell so as to retrieve the mutual relations between the elements in it, i.e. the robot, the sensor mounted on its end-effector and the part under inspection. This way, the fiber orientations can be mapped to the right points in the model and, similarly, the coverage planner can guarantee completeness, safety and constraints such as the sensor perpendicularity to the part. Specifically, two transformations are to be estimated: the former between the robot end-effector and the sensor, and the latter between the robot base and the part under inspection. The two procedures for getting them are briefly described in the following.

The hand-eye calibration [133, 135] estimates the transformation between end-effector and sensor, see Figure 4.2.1(a). The standard calibration process requires to view a calibration pattern, e.g. a checkerboard, from several different perspectives. As detailed in Appendix C, we extended the standard approach per-



(a) Hand-eye configuration in FibreMap: the "hand", i.e. the flange of the last link, is in the green box and the "eye" is in the cyan box.



(b) View of the three points defining the object reference system. They are to be touched with the tip on the robot tool center point in the red box. Here, they are defined on a rig aligned to the object for precision and repeatability[3].

Figure 4.2.1: Work-cell calibration.

forming calibration pattern localization and hand-eye calibration in a fully automatic way [136]. A two phase procedure has been developed and tested in both simulated and real scenarios, demonstrating that the automatic calibration reaches the same performance level of a standard procedure, while avoiding any human intervention. As a final contribution, the source code has been released[2]. Of course, standard camera calibration, i.e. the calculation of camera intrinsics and distortion coefficients, can be performed by means of the same set of images. To further mitigate distortion, which is visible in the full resolution images in Figure 4.0.1 (b), we worked on cropped images of size 800×200 px or 800×400 px. Indeed, distortion effects are minimal close to the image center.

In the industry production plants, a manual technique is usually exploited to

[2] http://robotics.dei.unipd.it/129-auto-hand-eye

[3] During the project, this rig was replaced by chalk signs left by an human operator just after the preform production.

define a reference frame on an object of interest, and thus its transformation with respect to the robot base. As shown in Figure 4.2.1(b), it consists in touching three reference points on the object with a calibrated tip mounted on the robot. These points can be hard to determine in case of objects with complex shapes, i.e. without well-defined edges nor vertices, and, once and for all, must be correctly identified also in the respective CAD models. This procedure is applied to three type of parts with increasing complexity, respectively named *Three-Hills*, *Spherical Bump* and *Complex Part*. More details about them will be reported in Section 4.6.

4.3 Image Registration for Accurate Fiber Orientation Measurement in Motion

The technique proposed in [82] for segmenting fibers and estimating fiber orientation is based on photometric stereo and requires to image a carbon fiber point under different angles of illumination, that are generated by LEDs arranged on an illumination ring around the camera. Experimentally, eight different angles of illuminations proved to be enough [82]. Thus, in this work, eight images are sequentially acquired and then fiber orientation is computed at each pixel. If the sensor is still, the corresponding pixels in the eight images represent the same point in space seen with different illuminations and fiber orientation can be estimated at every pixel. If the sensor moves during the acquisition of the eight images, the correspondence between pixel locations and 3D points is lost, thus the technique described in [82] cannot be directly applied.

In the following, two novel approaches for retrieving this correspondence are described. Both of them exploit the sensor positions provided by the robotic arm. The former approach works if the sensor is moving parallel to a flat surface while the latter works also for a generic sensor motion and a generic surface.

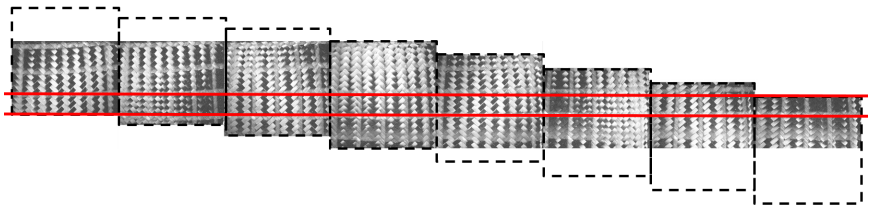


Figure 4.3.1: Eight images of a carbon fiber part acquired with different light sources and aligned to the reference frame of the forth image.

4.3.1 Image Registration for Sensor Translations and Planar Surfaces

If the object surface is planar and the sensor motion can be approximated with a translation $\mathbf{T} = (T_x, T_y, 0)$ parallel to this surface, the pixel displacement between the eight images corresponds to a 2D translation $\mathbf{t} = (t_x, t_y)$ in the image plane. The inverse of this translation can be used to refer all the images to a common reference frame, select the overlapping part and compute fiber orientation for all the points in the overlap. This overlap varies with the robot speed, the distance of the sensor to the part and the sensor field of view. In Figure 4.3.1, an example of image alignment and overlap selection is shown for eight images of a carbon fiber part acquired while the sensor was moving along the image vertical axis.

For estimating the transformation between two images of a planar scene, algorithms that extract and match features in the images are usually exploited. However, in the scenario considered in this work, these algorithms are unreliable because of the repetitive and regular texture of the carbon fiber pattern and the strong changes in illumination between images due to light source changes and high reflectivity of carbon fibers. In specific experiments, when using SIFT [91] detector and descriptor on carbon fiber images, more than 60% of wrong feature matches have often been measured, thus making the alignment with RANSAC [92] too prone to errors. This can be seen in Figure 4.3.2, which reports two consecutive frames taken with the sensor perpendicular to the part and a pure translational motion. Despite the simple transformation between the two frames, most of the

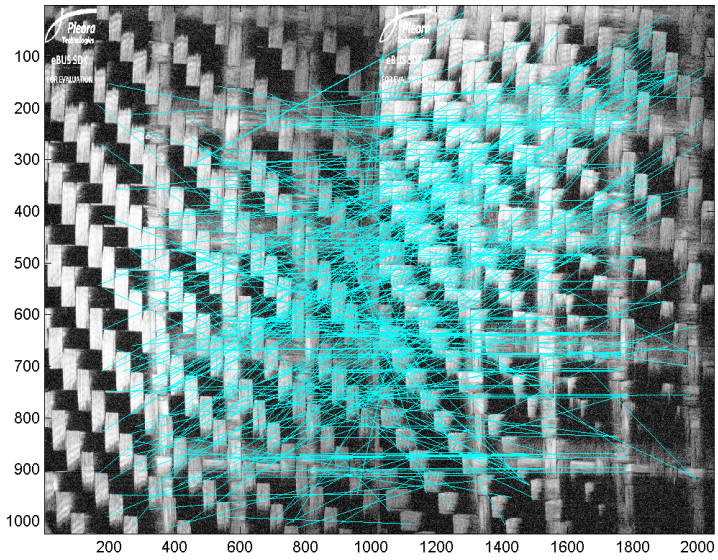


Figure 4.3.2: Example of feature matching for a pair of consecutive images using SIFT. Most of the matches are wrong so an alignment with RANSAC would not work.

matches are wrong due to the pattern and the changes in illumination between them.

Given the shortcomings of the standard approach, an image alignment procedure has been proposed, that takes into consideration the pose of the sensor associated to each image and provided by the robotic system in use and computes the 2D pixel translation $\mathbf{t}$ between images from the 3D translation $\mathbf{T}$ between sensor poses. In particular, it is assumed to have eight images taken under different angles of illuminations, $I_i, i = 1...8$, and the corresponding sensor poses, $P_i, i = 1...8$, derived from the calibration procedures mentioned in Section 4.1 and relative to the Cartesian reference system of the part. These images must be registered with respect to a common reference pose, that can be one of the eight camera poses, e.g. P_4. The relative poses, $RT_i, i = 1...8$, describing the transformations between the

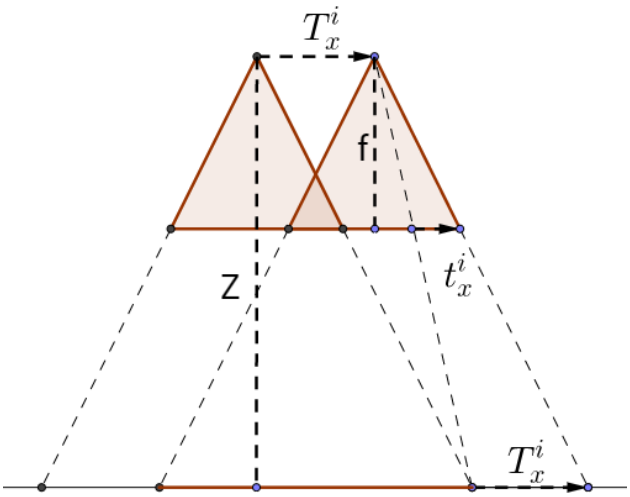


Figure 4.3.3: A sample translation along the x axis in 3D (T_x^i) and pixel (t_x^i) coordinates. Also the sensor height Z and focal length f are shown.

poses $P_i, i \neq 4$ and the reference P_4 can be computed. For each pair of poses (P_i, P_4), the translation vector in 3D coordinates (T_x^i, T_y^i, T_z^i) is the translational component of the roto-translation matrix RT_i. This vector is then converted to image coordinates (t_x^i, t_y^i), i.e. pixels, so as to apply it to each image $I_i, i \neq 4$ to perform the registration. The relationship between these translation vectors can be derived from the equations of the pinhole camera model [137], describing the mapping from 3D points to 2D points in the image plane of the sensor:

$$x_i = f_x \frac{X}{Z} + c_x,$$
$$y_i = f_y \frac{Y}{Z} + c_y,$$

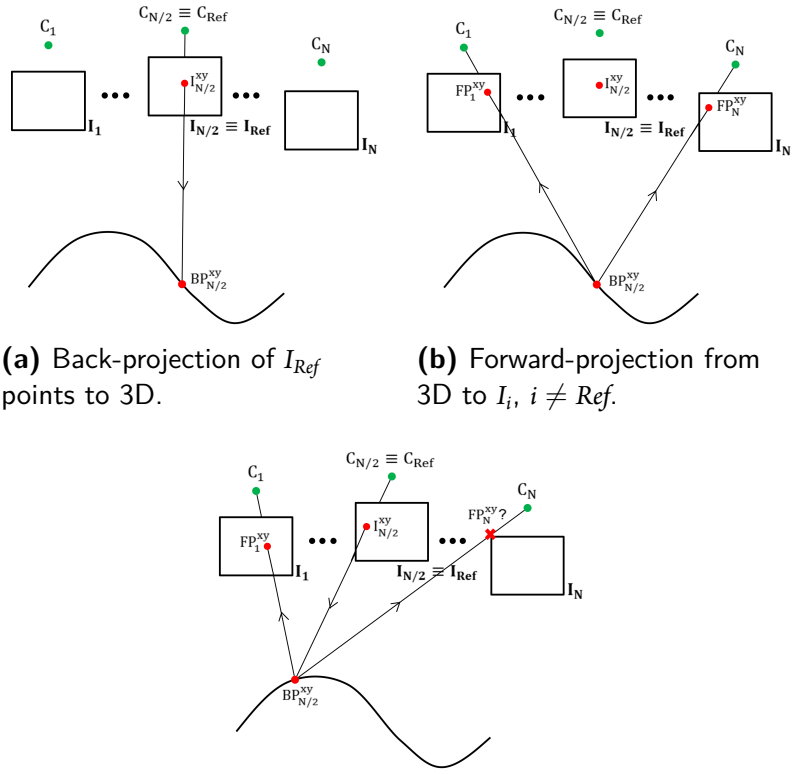
where f_x and f_y are the sensor focal lengths, c_x and c_y the optical center coordinates and Z the distance of the sensor to the part. The resulting proportions are then:

$$t_x^i : T_x^i = f_x : Z,$$
$$t_y^i : T_y^i = f_y : Z,$$

which allow to calculate the translation vectors in the image plane, t_x^i and t_y^i, from the 3D translation vectors, T_x^i and T_y^i. An illustration of the relation between 3D and image translations is reported in Figure 4.3.3.

4.3.2 Image Registration for 6DoF Sensor Motion and 3D Surfaces

For computing fiber orientation with maximum accuracy, the algorithm in [82] requires the sensor optical axis to be normal to the part surface. When scanning 3D objects, this implies to move the sensor with a six degrees-of-freedom motion in order to adapt the pose of the camera to the geometry of the part, thus the algorithm described in Section 4.3.1 cannot be used. For this reason, a method for registering points in different images without any constraint on the sensor motion or the part geometry is proposed in this work. In particular, the developed method consists in referring all the acquired images to a common (reference) image plane and select those pixels of the reference image that receive contributions from all the images. As reported in Section 4.3.1, the reference image can be any of the images corresponding to the eight illuminations. In the remainder of the chapter, we choose $N/2$ as the reference image, without lack of generality. Indeed, the extent of the overlapping region does not depend on the choice of the reference. For avoiding holes in the mapping of all the images to the reference image, every image is searched for points corresponding to those of the reference image. Indeed, as depicted in Figure 4.3.4, the proposed algorithm first back-projects [138] all the pixels of the reference image, in symbols I_{Ref}^{xy} where x and y identify the pixel coordinates, to the surface of the part, finding BP_{Ref}^{xy} (Figure 4.3.4(a)). Then, it forward-projects [138] these 3D points to the other images I_i, $i \neq Ref$, thus defining the forward-projected points FP_i^{xy} (Figure 4.3.4(b)). Fiber orientation can be computed only for those points of the reference image for which all forward-projections exist. As illustrated in Figure 4.3.4(c), if the forward-projection does not exist for one image, it means that the 3D point is not visible from that sensor pose, thus it is not possible to compute fiber orientation for that point. Then, a virtual image V is defined, with N channels that contains forward-projections from all



(a) Back-projection of I_{Ref} points to 3D.

(b) Forward-projection from 3D to I_i, $i \neq Ref$.

(c) Example of successful and missing forward-projection.

Figure 4.3.4: Image registration process for 6DoF sensor motion and 3D surfaces.

the N images and a binary mask M that keeps track of all the valid points in V, that are the points with forward-projections available from all the N images. Algorithm 2 describes this in pseudo-code.

The set of points in V for which M equals 1 are then used with the software in [82] for computing fiber orientation. This method is efficient since it projects only one of eight images to the mesh. Moreover, in comparison to the previous approach, there is no need to project the image after registration since it has been already projected during it. The projection process is detailed in the next section.

Algorithm 2 Image Registration Algorithm for 6DoF Sensor Motion and 3D Surfaces

procedure REGISTER(I) ▷ Array of N images I
  $N \leftarrow$SIZE(I)
  $Ref \leftarrow N/2$
  for all I^{xy}_{Ref} **do**
    $BP^{xy}_{Ref} \leftarrow$BACK_PROJECT(I^{xy}_{Ref})
    $M^{xy} \leftarrow true$
    for $0 < i < N \wedge i \neq ref$ **do**
      $FP^{xy}_i, M^{xy} \leftarrow$FORWARD_PROJECT($BP^{xy}_{Ref}$)
      if $M^{xy} = true$ **then**
        $V^{xy}_i \leftarrow FP^{xy}_i$
      else
        break
  return V, M ▷ Binary mask M and virtual image V

4.4 Efficient Rasterization for Real-time Arrow Mapping to 3D Models

The mapping process that projects the 2D fiber images to the 3D models is based on an highly optimized implementation of the *ray-casting* method [138], depicted in Figure 4.4.1. According to the base method, each pixel I^{xy} of a fiber image I is associated to a ray r^{xy} originated in the camera optical center C and passing through the pixel coordinates I^{xy} in the image plane. In particular,

$$r^{xy} = \{r^{xy}_c; r^{xy}_d\}$$

where $r^{xy}_c = C$ is the ray center and $r^{xy}_d = |I^{xy} - C|$ is the ray direction. As shown in Figure 4.4.1, if a ray r^{xy} intersects a triangle T_i of the model mesh, the intersection point P^{xy}_i is the projection of I^{xy} to the 3D model; if r^{xy} intersects the mesh in more than one point (i.e. triangle), the *z-buffering* technique [138] is applied and only the closest to C is considered. The ray-triangle intersections are computed effi-

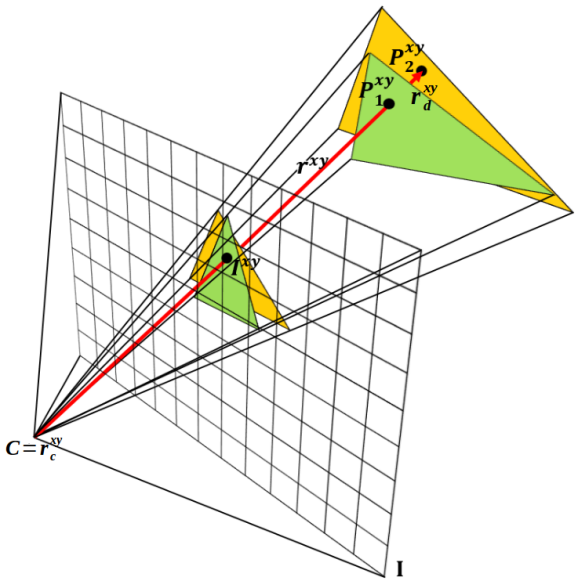


Figure 4.4.1: Ray-casting method for projecting pixel I^{xy} to point P_1^{xy} and P_2^{xy} in 3D space. According to the Z-buffering technique, if a ray r^{xy} intersects the mesh in more than one point P_i^{xy}, the closest to C, here P_1^{xy}, is considered.

ciently by specifying the triangle vertices in barycentric coordinates and by means of the Mőller-Trumbore algorithm [139]. In order to optimize both the ray-casting and the z-buffering techniques, the mapping process first checks which triangles can be hit by at least one ray. Each triangle T_i is forward-projected to the image plane; if the projection of T_i does not intersect the image area, T_i is discarded. The remaining triangles are further filtered by applying the back-face culling technique [140] which consists in the elimination of those triangles that lie in the back of the model surface, with respect to the observing camera. Next, for each of the remaining 3D triangles, the list of pixels belonging to the respective 2D forward-projected triangle are to be derived so as to project them. This process is known as triangle rasterization. As in [10], it can be implemented straightforwardly by approximating 2D triangles with bounding rectangles. Nevertheless, a triangle can be rasterized more efficiently by tracing out the pixels of its edges and rasterizing it line by line. In particular, this standard technique has been implemented and fur-

Algorithm 3 Efficient rasterization for mapping to 3D models

procedure RASTERIZE(I, M) ▷ Image I,3D model M
  $\{T_i\}_{i=0,\dots,N-1} \leftarrow$M.*faces*
  for $0 \leq i < N$ **do**
    $T_i^I \leftarrow$FORWARD_PROJECT(T_i, I)
    if $(T_i^I \cap \mathrm{I}) = \emptyset$ **then**
      REMOVE(T_i)
    else
      $\mathbf{r}_{T_i} \leftarrow V^0_{T_i} - C$ ▷ $V^j_{T_i}$: vertices of T_i
      if $(\mathbf{r}_{T_i} \cdot \mathbf{n}_{T_i}) \geq 0$ **then** ▷ $\mathbf{n}_{T_i}$: normal of T_i
        REMOVE(T_i)
  for $0 \leq i < N$ **do**
    $\{(I^{xy}, P_i^{xy})\} \leftarrow$RASTERIZE_BY_LINES($T_i^I$)
  for all (x, y) in I **do**
    $P^{xy} = \arg\min_{P_i^{xy}} |P_i^{xy} - C|$
  return $\{P^{xy}\}$ ▷ Projection of points in I to M

ther optimized so as to raster whole pixel lines at once instead of single pixels. By organizing rays, projected points and intermediate entities into vectors, the SSE2 vectorization offered by the *Eigen* algebraic C++ library [25] can be fully exploited for every arithmetical computation, thus making the total runtime decrease by a factor of four. Hence, this implicates that bigger frames can be processed online, positively impacting the total scanning time. The pseudo-code of this method is reported in Algorithm 3.

4.5 Filtering and Multi-view Fusion

Fibers can be segmented and their orientations can be measured on the overlapping region defined by the registration algorithms in Section 4.3. Then, this set of fibers can be mapped to the 3D model of the scanned part according to the algorithm in Section 4.4. Anyway, these fibers can be noisy so three additional filtering strategies are necessary.

First of all, for each mapped fiber, we associated the inclination of the sensor z

Figure 4.6.1: Description of the robotic scanning system used for the tests and how the different tasks are distributed over the different computers.

axis with respect to the part surface. This way, since fiber estimation works better when this inclination is low, only the arrows for which it is below a given threshold can be retained. A tolerance of $\pm$ 10.00° proved to be enough to meet our accuracy goal.

Furthermore, given that the optimal scanning distance of the sensor is known ($\approx$ 15 *cm*), other parameters like the minimum and maximum distance can be set for leaving out invalid measurements.

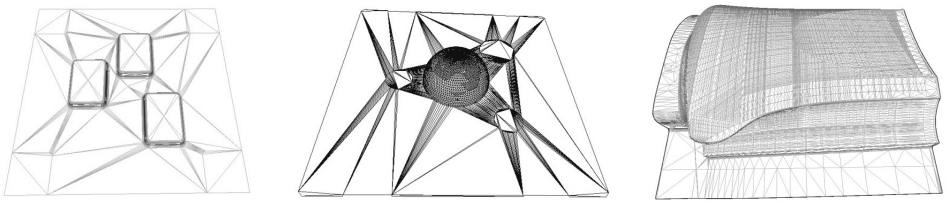
Finally, the sensor may pass twice or more on the same area so almost overlapping measurements estimated at different moments are to be fused. To solve this issue, when more valid measurements are present, our algorithm retains the best one in terms of orientation of the sensor with respect to the surface (maximum perpendicularity). This strategy has been implemented exploiting an octree data structure [141], which provides space-efficient methods for creating a hierarchical tree from point cloud data, enabling spatial partitioning in voxels, down-sampling and search operations on the point data set. Indeed, after a complete scan, the number of measurements can be high (up to $\approx 10^9$ with the most complex part) so this solution allows to quickly access to the point context, i.e. its neighbours, while saving memory space.

4.6 Experiments and Assessments

Several experiments have been performed for assessing the proposed techniques with the robotic scanning application described in Section 4.1. In particular, this work considers the system in Figure 4.0.2a, that features a Comau Smart5-SiX robotic manipulator and a C5G controller communicating via Ethernet Powerlink. The C5G controller includes an industrial PC with Linux Mint and RT Preempt that communicates with the robot in real-time. As depicted in Figure 4.6.1, that PC is used to control the robot according to the output of a path planner and to continuously stream robot positions as UDP messages. Another computer within the same network, a desktop PC running Windows 7, performs the computation of the robotic path and runs all the image acquisition and processing software. Multiple operating systems were necessary to integrate a variety of libraries and third-party softwares with different requirements in the same working system, e.g. the sensor library and the path planner work only under Windows.

4.6.1 Experimental Results

The tests described in this section were performed on three different carbon fiber preforms of increasing complexity, that are shown in Figure 4.6.2. For each of these preforms, different types of material have been tested: woven or not, coated with EP-binder or not and with different filament counts of each bundle, e.g. 6 K or 24 K, and fiber orientation, e.g. +/- 45°or 0/90°. For these experiments, the meshes of these preforms have been derived from the CAD data of the press tools used for forming them. The *Three Hills* preform (Figure 4.6.2(a)) is a mostly flat sheet with three rectangular hills on top, while the *Spherical Bump* (Figure 4.6.2(b)) has an hemisphere at its center and is considerably larger. The *Complex Part* (Figure 4.6.2(c)) features a more complex shape, derived from a real automotive part. Tab. 4.6.1 reports the preform dimensions and the number of faces of the associated meshes. It can be noticed how the *Complex Part* has an order of magnitude more faces than the other meshes. Since the mesh of the scanned part is an input of the software for image registration and fiber projection, as it will



(a) Three Hills mesh **(b)** Spherical Bump mesh **(c)** Complex Part mesh

Figure 4.6.2: Pictures and mesh models of the carbon fiber parts used as test cases.

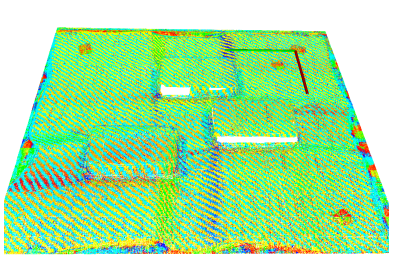
Table 4.6.1: Dimensions of the test cases and number of faces of the corresponding meshes.

	Three Hills	**Spherical Bump**	**Complex Part**
Dimensions (mm)	400x400x20	700x700x120	900x900x160
Number of faces	5974	6246	81690

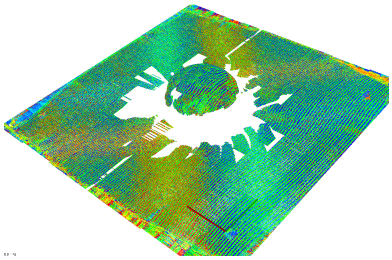
be pointed out later, the number of faces (triangles) of the mesh is a parameter that considerably affects the frame-rate of these algorithms.

Figure 4.6.3 reports three examples of complete scans, one for each test case, obtained with the pipeline reported in Figure 4.1.1 and the most general algorithm for image registration proposed in Section 4.3.2. These scanning results consist in 3D models of the parts mapped with fiber orientation measured at each point. For visualization, the azimuth value of the fiber orientation with respect to the part reference frame is represented with colors.

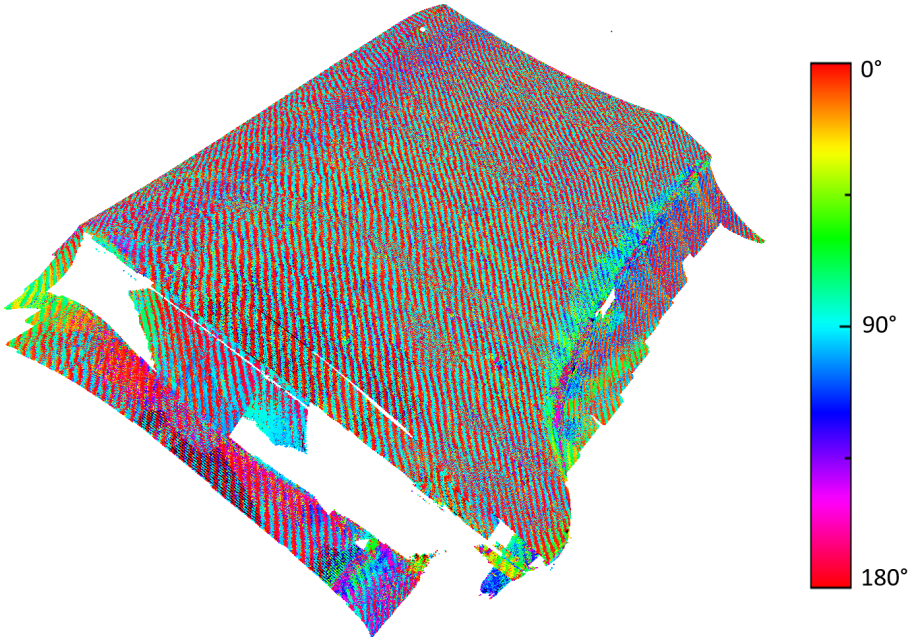
As displayed in Figure 4.6.4, such a system can also be used for continuous and real-time texturing of objects with a robotic system. In particular, the proposed setup has been used for optimally blending together the images taken with different illumination by a moving sensor, in order to obtain constant lighting all over the blended image. Then, these blended images have been projected to a 3D model of the scanned object.



(a) Three Hills

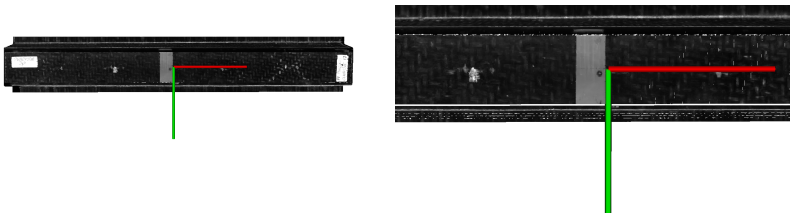


(b) Spherical Bump



(c) Complex Part

Figure 4.6.3: Examples of complete scans for the three test cases. Colors represent the azimuth value of the fiber orientation with respect to the part reference frame according to a cyclic color map. The error in azimuth is 0.48° and 1.55° on flat and 3D surfaces respectively. Because of the physical sensor size, the most concave and narrow regions cannot be analyzed without colliding or breaking the sensor-surface perpendicularity constraint. With a smaller sensor, they could be reached.



(a) Top view.

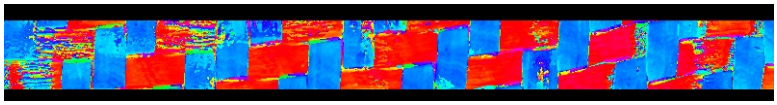
(b) Detailed view. The gray region is a tape we added for reference.

Figure 4.6.4: Two views of a 3D textured model of a glass fiber part. The texture was very dark. Our system can be used to optimally blend together the images taken with different illumination by the moving sensor.

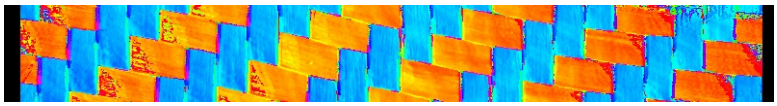
4.6.2 Assessment of Image Registration Accuracy

To qualitatively assess the algorithm proposed in Section 4.3.1 for image registration in presence of sensor translations and planar surfaces, Figure 4.6.5(a)-(b) shows some results of fiber computation when the sensor was moving along the y and x image axis, respectively. These images represent the fiber azimuth angles and the black parts refer to pixels that do not belong to the overlap between the eight raw images. Translations across both axes at the same time have been tested too, as reported in Figure 4.6.5(c). Fibers are badly estimated only along some chalk lines that were drawn as a reference, since the reflection properties of the material are altered by the chalk. In Figure 4.6.5(c), the azimuth images obtained by inverting on purpose the translation signs are reported in order to show an example of how a bad registration would look like.

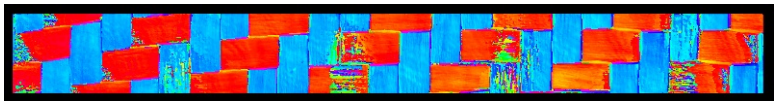
As previously explained, the algorithm described in Section 4.3.2 allows to correctly estimate fiber orientation even when the sensor is moving with a 6DoF motion and the surface is 3D. Figure 4.6.6 reports an example of azimuth image computed by the sensor when performing a 3D rotation. As it can be noticed, the colours representing the azimuth values are stable, thus reflecting a correct regis-



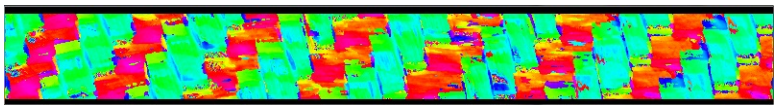
(a) Translation along y image axis only.



(b) Translation along x axis only.



(c) Translation along both axis.



(d) Example of bad registration, the translation signs have been deliberately inverted.

Figure 4.6.5: Azimuth images computed with the sensor performing translations over a planar surface.

tration of the eight images with different illumination acquired in motion.

In order to quantitatively measure the accuracy of the registration algorithm in Section 4.3.2, some images of a flat area and a 3D area of a *Spherical Bump* part (as shown in Figure 4.6.7) were acquired while the sensor was moving over the part with a generic (6 degrees of freedom) motion. As a reference, a checkerboard pattern was attached onto these areas. A checkerboard corner was then clicked on the first image of each series and the displacement (in pixels) of the corresponding scene point in the other images after the image registration operation was measured. Figure 4.6.8a reports eight images acquired on the flat area after having performed the image registration with respect to the 5th image, while, Figure 4.6.8b reports the alignment result for eight consecutive images acquired from the 3D

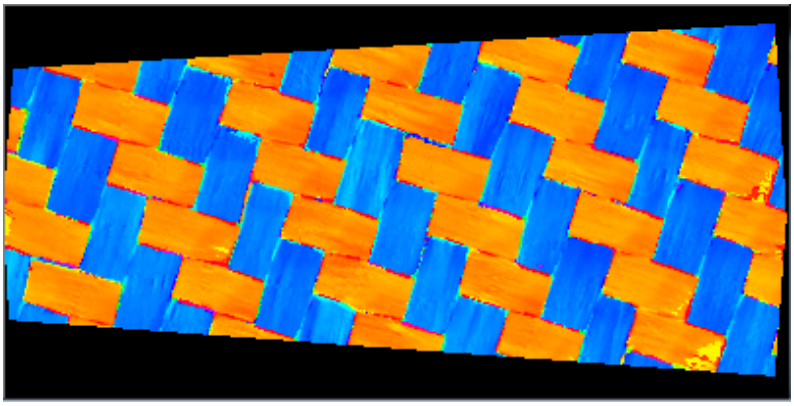


Figure 4.6.6: Azimuth image computed while rotating the robot end effector along the axis of the 6^{th} robot joint.

area. A red cross marks the same pixel position in all the images of a sequence as a reference to show the accuracy of the registration. From a visual inspection, it can be seen how this cross falls on the same scene point for all the images of the flat area, while it is slightly worse placed for some images of the 3D area. In particular, an average displacement of 3.1 pixels for the flat area and 7.8 pixels for the 3D area has been measured for images of 800×400 pixel resolution. These registration inaccuracies are quite small and could be due also to some differences between the CAD model of the part used for aligning the images and the real shape of the part that is scanned.

4.6.3 Assessment of Fiber Orientation Measurement Accuracy

The accuracy in estimating fiber orientation in motion on both flat and 3D surfaces has been evaluated by using the most general approach for image registration, proposed in Section 4.3.2, and the rasterization algorithm described in Section 4.4. We compare this approach with the baseline method presented in [88], which can estimate fiber angles with an error of less than 1.00°. Nevertheless, since it cannot work with a moving robot, it requires a stop and go motion, severely affecting the scanning time.

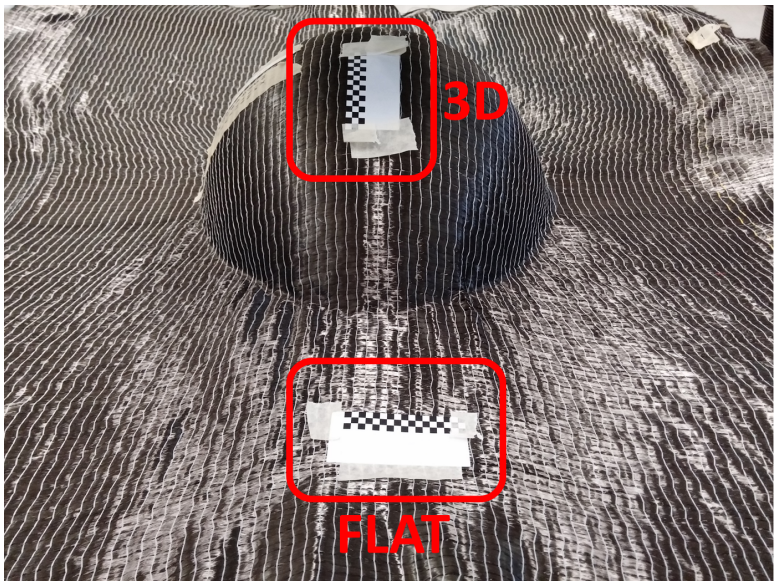
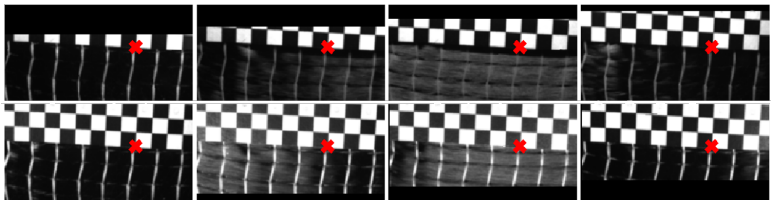
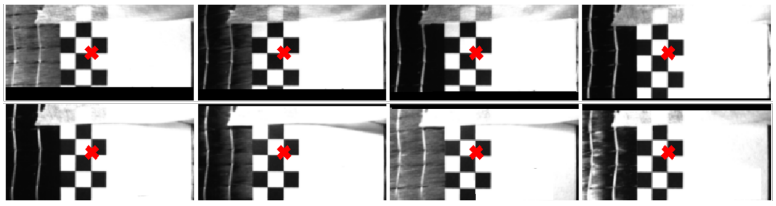


Figure 4.6.7: Experimental setup for evaluating the accuracy of the image alignment.

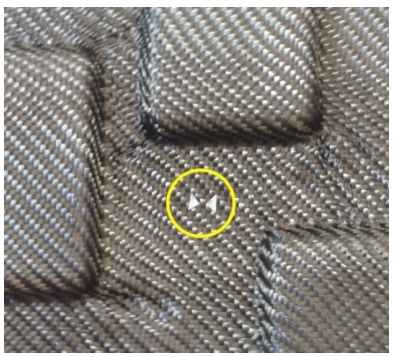


(a) Alignment evaluation on a flat part.

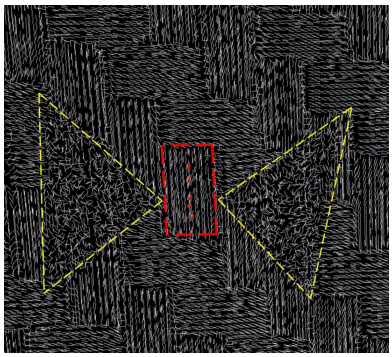


(b) Alignment evaluation on a 3D part.

Figure 4.6.8: Sample results of the image alignment evaluated on flat and 3D parts. The red cross highlights the same pixel in all the images of a sequence.



(a) Bundle of fibers marked on a *Three Hills* preform with some tape

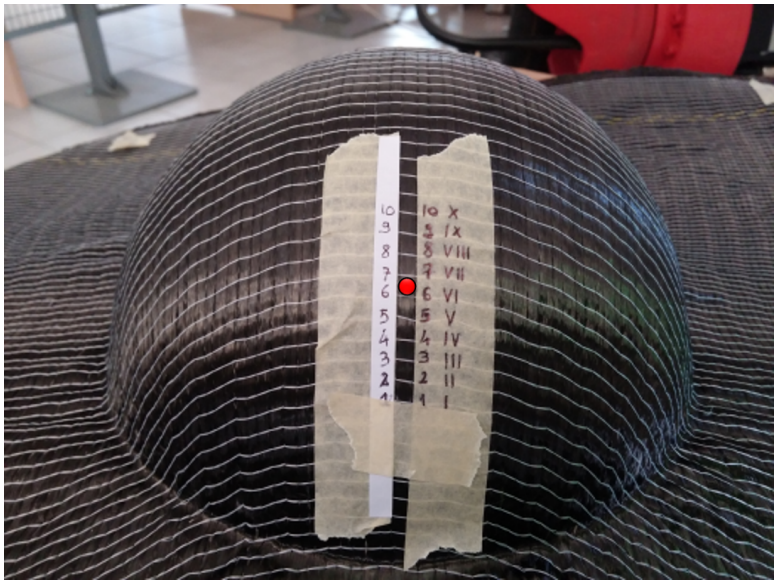


(b) Fiber arrows measured in the marked area with a sensor translation over the part. The tape is highlighted in yellow and the selected bundle of fibers is highlighted in red.

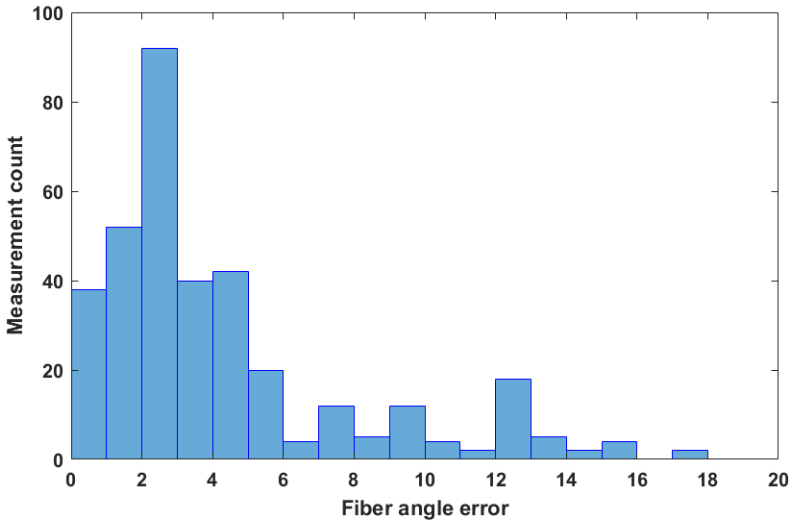
Figure 4.6.9: fiber orientation accuracy test on flat parts. The fiber orientations measured on the tape are random while they are locally the same on the rest of the image.

For the test on the flat area, a bundle of fibers on a flat part of a *Three Hills* preform was marked with two pieces of tape around it, as shown in Figure 4.6.9a, and a scan with the sensor translating over the part was performed. This way, as shown in Figure 4.6.9b, the orientations measured by the sensor system on the selected bundle of fibers could be easily retrieved and compared with the nominal value, that is the orientation measured by using a tip mounted on the robot for touching the two extremes of the fiber bundle so as to determine fiber direction. The mean error in fiber orientation within the selected bundle of fibers was 0.48°, thus below 1.00°, as in the static case [88].

To estimate the accuracy of fiber measurement in 3D, a *Spherical Bump* preform was marked with tape so as to select a particular bundle of fibers whose reference fiber orientation was also measured with the robot tip by touching several points on the selected curve, part of the hemisphere. The part was then scanned



(a) *Spherical Bump* preform marked with tape and, in red, the area selected for evaluation.



(b) Histogram of the error on fiber orientation for points within the selected area.

Figure 4.6.10: Fiber orientation accuracy test on 3D parts.

with a 6 degrees-of-freedom sensor motion for calculating the difference between the 3D orientations obtained with the proposed system and the curve measured with the robot tip within the region highlighted in red in Figure 4.6.10a. Figure 4.6.10b reports the histogram of this measurement error. It can be noticed that the peak 3D orientation has an error of 3.00° with respect to the reference and that the average error is of about 4.37°. This value is above the static error [88], but still within the tolerances allowed by the automotive industry. Another experiment was performed on a *Complex part*, this time enabling the filtering and multi-view fusion. Five pairs of markers like those in Figure 4.6.9(a)(b) were applied on it. Markers are almost white so they can be easily detected using the diffuse intensity available in the raw images and the points in between can be considered for evaluation. In particular, the ground-truth measurements were acquired keeping the sensor static. This way it can measure fibers with an error minor than 1.00°. After scanning them 12 times with different paths, we measured an average error of only 3.86° in 3D orientation and only 1.55° in azimuth.

Finally, positioning inaccuracies were also evaluated. Given that it is hard to manually define the ground-truth position of a fiber, we limited the analysis to the calculation of the average deviation w.r.t. the average positioning when scanning the same fiber several times. Again, the setup comprehending five pair of markers like those in Figure 4.6.9(a)(b) on a *Complex part* was considered. After scanning them 12 times with different paths, the position values on the 5 points were compared to the average positions finding an average difference of only 3.14 mm with a standard deviation of only 0.16 mm.

The estimated errors are below the accuracy that can be obtained with manual measurements, that is how fibers are currently measured in factories. Moreover, the proposed system allows to perform a dense inspection of the parts, that is a clear advantage with respect to manual inspection.

Table 4.6.2: Frame-rate (in frames per second) of the registration algorithm described in Section 4.3.2 for the test cases described in Section 4.6.1.

	Three Hills	**Spherical Bump**	**Complex Part**
Registration (fps)	45.9	44.5	28.0

Table 4.6.3: Frame-rate comparison (in frames per second) for different versions of the algorithm for projecting fiber arrows to the 3D meshes of Figure 4.6.2.

	Rect	**Rect+SSE2**	**Triangle**	**Triangle+SSE2**
Three Hills	3.9	44.8	47.6	243.8
Spherical Bump	14.8	127.4	47.3	202.0
Complex Part	14.5	52.1	28.5	57.7

4.6.4 Frame-rate Analysis

The frame-rates of the registration and projection algorithms described in Section 4.3 and 4.4 have been measured for all the test cases reported in Figure 4.6.2. All the tests have been performed with a desktop PC with an Intel i7-4770K CPU clocked at 3.5 GHz, 16 GB of DDR3 RAM, a 7200 rpm hard disk and Windows 7. The frame size was 400×200 pixels. As it can be noticed in Table 4.6.2, the frame-rate of the registration algorithm considerably decreases for the Complex Part test case, due to the higher number of faces that the mesh has with respect to the first two test cases. This is also true for the projection algorithm, whose frequency is reported in Table 4.6.3 with four different versions of the implemented algorithm. The Rect version exploits rectangular bounding boxes around a mesh face to determine ray intersections, while the Triangle version exploits triangular bounding boxes. The SSE2 versions exploit the rasterization of whole pixel lines at once thanks to SSE2 vectorization offered by the Eigen library, which makes the frame-rate increase by a factor of four.

The proposed method has proven to be efficient enough for real time scanning. These frame-rates allow to perform a complete scan of a *Three-Hills* part (400x400

mm) with an average speed of 0.02 m/s in about 5 minutes, that corresponds to the average time needed for manually inspecting it. The other two test cases can be completely scanned in about 15 minutes, that is half the time required to scan them in a stop and go fashion [88]. The sensor can support also high frame-rates, theoretically up to 1000 fps, making it possible to move the robot at higher speeds, theoretically up to 1 m/s. With the sensor capturing 800×400 pixels at 125 fps and the proposed algorithms running on downsampled frames of size 50×25 pixels, the robot could move at 0.10 m/s and inspect the most complex test case in only 150 s. A video is attached to demonstrate the quality of the system[4].

4.7 Summary

This chapter presented efficient algorithms for segmenting and mapping fibers to large surfaces with a moving sensor so as to obtain a geometric model augmented with the carbon fiber arrangement. Specifically, we addressed the complex scenario of sensors that need multiple shots to perform a measurement by proposing image registration and mapping algorithms enabling their use in continuous motion. The application of segmenting and mapping carbon fibers to 3D models with an autonomous robotic system has been considered for assessing the quality of this work in scenarios where the sensor moves with generic 6DoF motion over surfaces of complex shape. The reported experiments show how the proposed algorithms allow to segment fibers and densely estimate in real-time the fiber orientation of carbon fiber preforms. In particular, a frame-rate of about 30 frames per second is achieved for the most complex test case, that is derived from a real automotive component. The fiber mapped models produced with this system can be effectively used for either quality inspection or for improving draping simulation. Indeed, the system accuracy has been evaluated on flat and complex 3D surfaces, reaching an azimuth error of only 0.48° and 1.55°, respectively. These values, together with the possibility to perform a dense and complete check of the part, represent a clear advantage of the proposed system against a manual check,

[4] https://www.youtube.com/watch?v=a8vLiNIU870

that is usually operated in factories. Nevertheless, as discussed in the next chapter, several types of defect could affect the production process of Carbon Fiber Reinforced Polymers making of interest also the segmentation and mapping of other object properties besides fiber orientations.

5

Thermographic Segmentation of Defects in Upper Layers of Carbon Fiber Parts

In Carbon Fiber Reinforced Polymers (CFRPs), not only are the superficial fiber arrangements of interest, but also the inner properties invisible to the human eye, which are to be considered for a thorough quality inspection. Carbon fiber sheets can be layered onto each other and joined together by means of a glue layer, to be placed between every couple of adjacent carbon sheets; the whole structure is then filled with polymer to create CFRPs. Carbon fibers define the reinforcement, which provides the strength, while the thermosetting polymer, e.g. epoxy resin, is the matrix, which binds the reinforcements together [142]. CFRPs can be manufactured using a number of techniques, all of which aim to combine the fiber and resin into a well-consolidated product. Several types of defects can affect

the production process, including porosity, foreign bodies, incorrect fiber volume fraction due to excess or insufficient resin and bonding effects [143]. Defects can also occur in the placement of the glue between consecutive carbon fiber sheets. In this chapter, we will examine how these properties can be segmented and mapped to a 3D model by means of the back-projection from 2D to 3D introduced in Section 1.2. A complete model will be acquired by inspecting the part from different points of view.

Detecting defects in CFRPs is one of the goals of the European project Thermobot[1], which aims at replacing manual inspection in order to achieve better efficiency and performance in the production phase. The research leading to the results presented in this chapter is part of this project, and aims at inspecting the glue layer that bonds together carbon fiber sheets. Even though such layer is inside the part to be inspected, thermography, in particular the technique named Pulsed Phase Thermography (PPT) [144, 145] is capable of revealing the glue disposition, thus enabling the inspection. Three modules have been developed for the project: thermographic inspection, robot path planning and thermal image analysis.

PPT enables the nondestructive testing and evaluation (NDT&E) of CFRPs. Its data acquisition setup consists of a flash lamp, the Hensel EH Pro 6000 with its power supply Tria 6000-S, and an infrared camera, the ImageIR 8300 by InfraTec GmbH, Germany. The flash lamp is an external source of energy which applies a short thermal pulse inducing a temperature difference between areas with different distribution of glue in the specimen under examination. The thermal changes on the surface are captured by the infrared camera. An important parameter is the acquisition duration, which defines the frequency at which the PPT is evaluated and influences the capability to discriminate the property of interest. The flash, whose power can be set, overpowers the ambient light making the acquisition process controllable.

The flash lamp and infrared camera are mounted on an industrial robot, the

[1] http://thermobot.eu

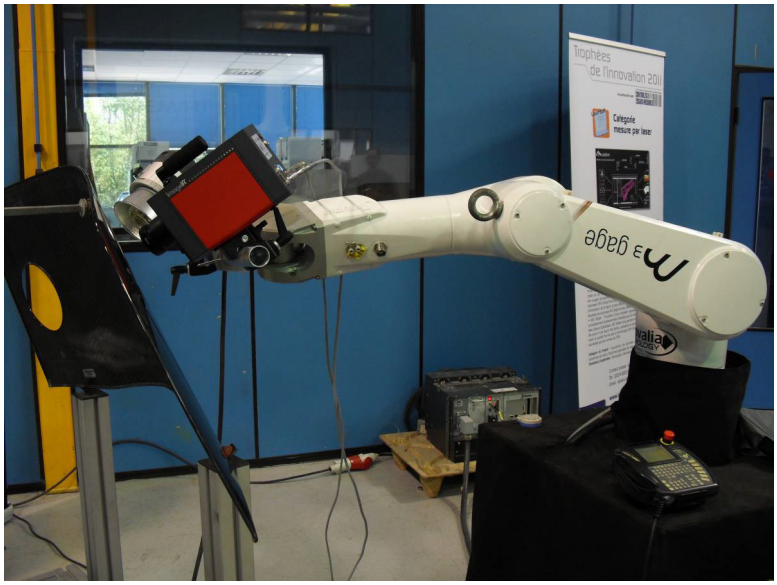


Figure 5.0.1: Complete experimental setup with robot, infrared camera, flash lamp and CFRP.

Stäubli TX90L with the CS8C controller, in order to inspect parts larger than the field of view of the camera. In particular, the surface of CFRP components are inspected in a series of static tests from a number of points of view such that the part can be covered. In particular, the perimeter is more interesting than the rest of the part because the glue is distributed along the edge. The system is depicted in Figure 5.0.1; the inspected component is a car side blade manufactured by one of the partners of the project, Benteler-SGL, Austria.

Images are acquired for each point of view and then processed for obtaining the PPT phase images. Their analysis for revealing defective areas is the main focus of this chapter. Unfortunately, given the lack of a big dataset, previous approaches based on machine learning cannot be applied. Thus, the problem is addressed by adapting an image sharpening technique well-known in photography, the UnSharp Masking (USM), which enhances contrast taking pixel context into account. Moreover, the segmentation is enhanced by means of a knowledge-based approach [102], i.e. by taking advantage of the high level of knowledge of the entire system gained thanks to the calibrations and applying the back-projection from

2D to 3D presented in Section 1.2. Since the bonding cannot be simulated, defects are revealed by comparing each PPT phase image with the respective defect-free reference. The imaging process is as controllable as the acquisition given the repeatability of the workcell conditions. In summary, the main contributions of this chapter are:

- an automatic robotic system for the detection of defects in the upper layers of CFRPs;
- a thermographic analysis of CFRPs based on PPT;
- the PPT phase image analysis exploiting USM and the high level of knowledge and control of the real workcell;
- the defect detection by comparison with a fault-free reference.

The detection of internal cracks cannot be performed since the required amount of heat generated by the flash lamp would be so high that it would damage the inspected object.

This chapter is structured as follows. The core of the visual inspection system that enables the detection of defects is detailed in Sections 5.1-5.4. The results obtained during the experiments are described in Section 5.5. Finally, in Section 5.6, the main achievements are recapped and some final remarks reported.

5.1 System Overview

This section focuses on the description of the comparison between two PPT phase images. They are gray-scale images, like the one in the top-left corner of Figure 5.1.1, in which darker shades of gray denote the presence of glue. This process is characterized by three main steps displayed from the top in Figure 5.1.2:

1. foreground extraction from PPT phase images;
2. detection of adhesive bondings in CFRP;

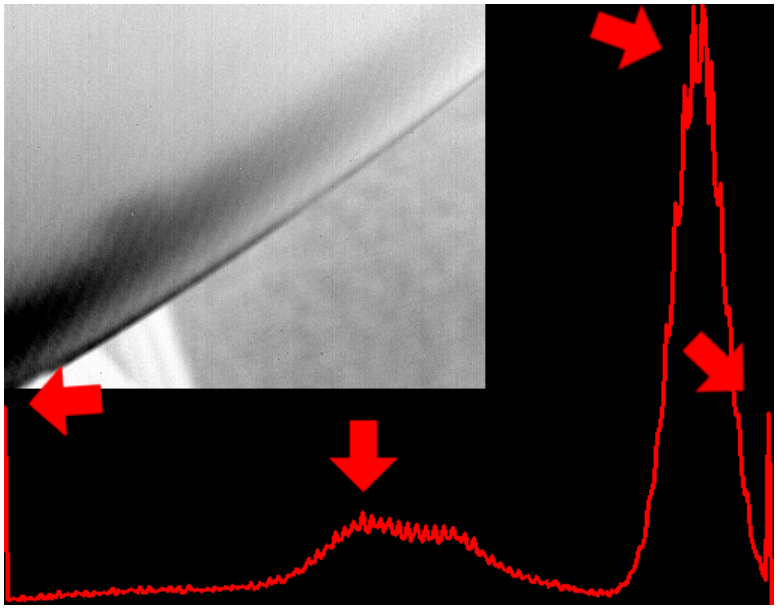


Figure 5.1.1: A PPT phase image (top-left) and its brightness histogram. There are four main peaks denoting that Otsu's binarization cannot be applied because the image is not bimodal.

3. pairwise comparison of adhesive bondings for difference detection.

The first two aim at restricting the comparison to a region of interest with glue in order to minimize false positives and false negatives. A simpler pixel-by-pixel comparison would be unreliable because of the differences in the carbon fiber texture.

5.2 Foreground Extraction from PPT Phase Images

Two complementary regions can be found in each image, one corresponding to the CFRP part in the foreground and the other to the background. As shown in the CFRP part in Figure 5.1.1, both regions might have non-uniform brightness distribution because the glue might not be uniformly distributed on the surface and the distance of each point in the background from the infrared camera might

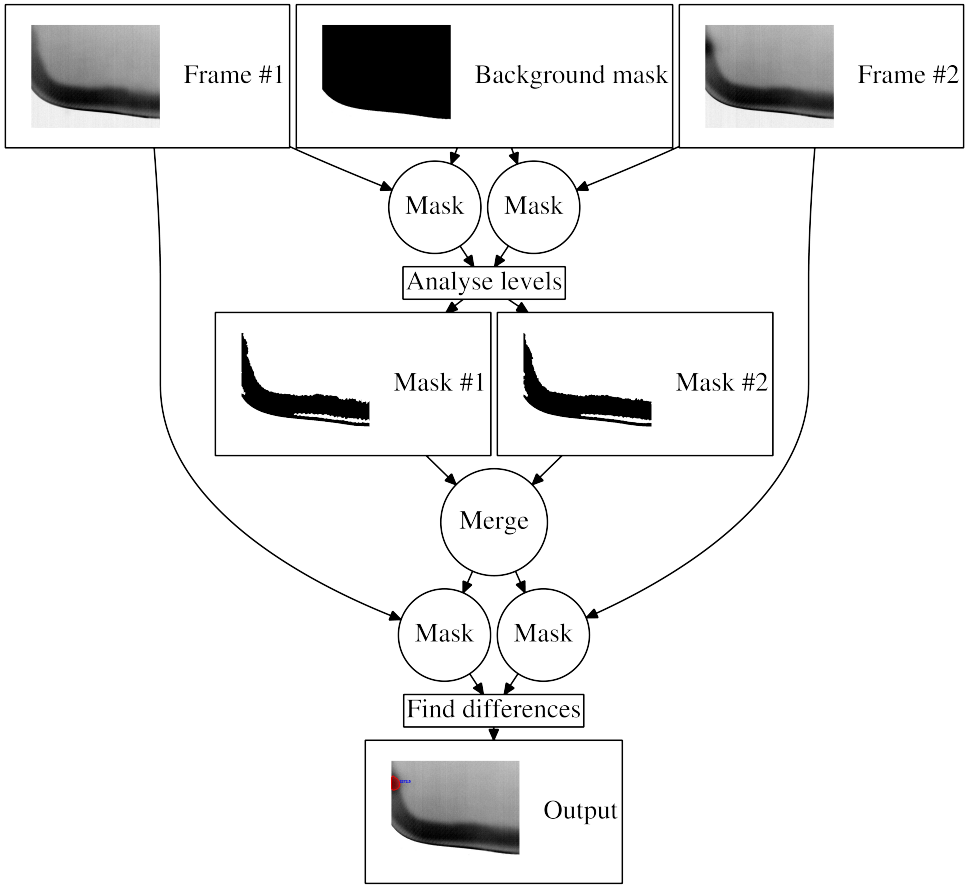
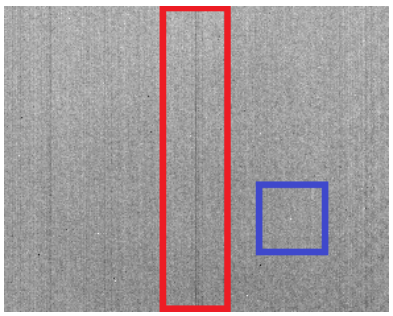


Figure 5.1.2: Flowchart illustrating the comparison process of two PPT phase images. This process is characterized by three main steps displayed from top to down: foreground extraction from PPT phase images, detection of adhesive bondings in foregrounds and pairwise comparison of adhesive bondings for difference detection.

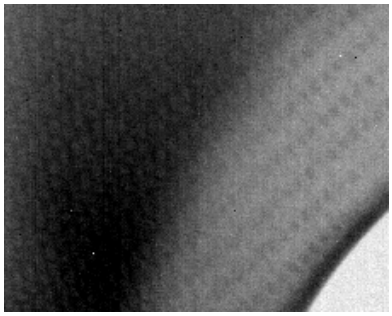
vary. Of course, a way to binarize images and extract foreground is of concern. As a consequence of non-uniform brightness distribution in both regions, not only is simple thresholding not accurate, but also no threshold can be automatically calculated from image histograms, e.g. by means of the Otsu's binarization [146]. Indeed, images are not bimodal as demonstrated by counting the number of main peaks in the brightness histogram shown in Figure 5.1.1.

Viable alternatives are to be evaluated. One of them is the knowledge-based approach [102] by means of the high level of knowledge of the experimental setup and the backward projection [147] of each image point onto the 3D polygon mesh model of the CFRP part. The model, view and projection transformation matrices, whose composition is the mapping from the part space to the image space, are all well-known. These pieces of information come from three preliminary calibrations that respectively lead to the knowledge of the camera parameters, the position of the camera with respect to the robot, and the position of the robot with respect to the part. In this way, the foreground extraction is performed by evaluating for each image point the intersection between the 3D model of the part and the ray passing through that image point and the focal point. If the ray intersects the 3D model, the image point falls on the part and belongs to the foreground, otherwise it does not fall on the part and belongs to the background.

As another option, if these matrices are not available, the foreground can be extracted with minimal user interaction running the GrabCut algorithm [53]. As introduced in Chapter 2, this algorithm starts with a user-specified bounding box around the foreground region and some hints on background and foreground pixels inside of it. Then it creates the background/foreground segmentation. The system combines hard segmentation by iterative graph-cut optimization with border matting to deal with blur and mixed pixels on object boundaries. Obviously, this latter approach, which requires interaction with the user, is more tedious.



(a) ISO noise or film grain in the blue box and vertical banding noise in the red box.



(b) Carbon fiber pattern.

Figure 5.3.1: Examples of noise.

5.3 Segmentation of Adhesive Bondings in CFRP

Dark shades of gray denote the layer of glue. Nevertheless, the definition of dark area is not straightforward and cannot rely on simple thresholding: first of all, the chosen value would not be the best for each image in the dataset, given the non-homogeneity of the glue layer in the CFRP part; second, illumination might not be uniform over the whole CFRP part. These remarks about brightness values hold mostly when comparing images taken from different points of view and secondarily when comparing different regions of an image. Hence, what is required for the detection of the glue layers is an approach that takes into account brightness gradients: this way, it is possible to detect dark regions compared to their surroundings, i.e. their context. For this kind of analysis, an approach based on local contrast adjustment has been studied and developed. This approach is performed in four steps:

1. noise filtering;
2. unsharp masking;
3. thresholding;

4. artifact filtering.

The core of the algorithm is the unsharp masking, which is not applied standalone because it could enhance unwanted details due to ISO noise or film grain, as in the blue box in Figure 5.3.1(a), and vertical banding noise, as in the red box in Figure 5.3.1(a). In addition, also artifacts like the carbon fiber pattern itself, Figure 5.3.1(b), could be strongly highlighted. This can be made up for by adding two steps, a pre-processing and a post-processing filtering.

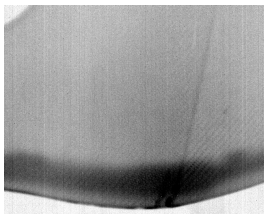
The pre-processing is a median filter which proved to eliminate vertical noise more effectively than a Gaussian filter. An aperture size of only 13 px is an appropriate trade-off between noise removal and level of detail.

UnSharp Masking (USM) is an image sharpening technique well-known in photography. Let O be the input image, depicted in Figure 5.3.2(a), in this case the softly blurred copy of the starting frame obtained after the pre-processing step, and B its blurred copy, depicted in Figure 5.3.2(b). The blur can be a 2D Gaussian blur with a certain aperture size, the radius r. The method consists in calculating an unsharp mask U, the difference between the original image O and its blurred copy B as shown in Eq. 5.1, and an high contrast version of the original image C, the sharpened difference between the original image O and its blurred copy B as shown in Eq. 5.2. The amount of overshoot, namely how much contrast is added, can be set via the parameter a.

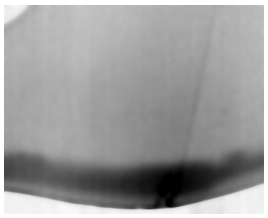
$$U = |O - B| \tag{5.1}$$

$$C = a * O - a * B \tag{5.2}$$

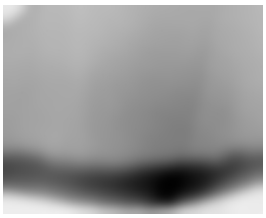
In Figure 5.3.2, the intermediate images are reported: O (b), B (c), U (d) and C (e). The unsharp mask U roughly points out the contours, which are sharpened in the high contrast image C. The sharpened image S is the sum of the original image O and the high contrast image C, optionally masked with the unsharp mask U by setting the threshold th, see Eq. 5.3. Given a 2D point P_U on the unsharp mask U,



(a) Initial frame. This is the softly blurred copy of the starting frame obtained after the preprocessing step.

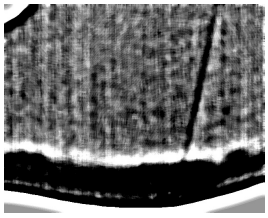


(b) Initial frame after soft Gaussian blurring O.



(c) O after heavy Gaussian blurring B.

(d) Unsharp mask U.



(e) High contrast image C

Figure 5.3.2: Unsharp masking is based on the simple detection of contours performed by subtracting blurriness B from the initial frame O.

let I_{P_U} be its gray level varying from 0 to 255:

$$S = \begin{cases} O & \text{if } I_{P_U} < th \\ O + C & \text{otherwise.} \end{cases} \tag{5.3}$$

The parameter *th* is the minimum difference in pixel values that indicates an edge to which some sharpen must be applied to protect areas of smooth tonal transition from sharpening. If it is set to 0, the unsharp mask U is not taken into account. The sharpened image S is depicted in Figure 5.3.3(a). The choice of the three parameters, the radius *r*, the amount *a* and the threshold *th*, is impor-

(a) The sharpened frame S obtained by unsharp masking.

(b) The sharpened frame S after thresholding.

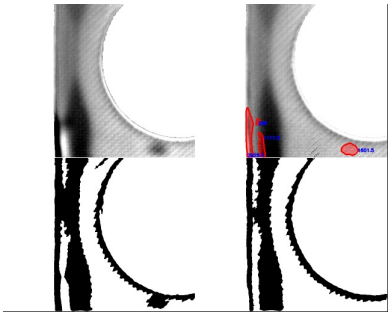
(c) The sharpened frame S after thresholding and the last post-processing step for removing artifacts characterized by small contours.

Figure 5.3.3: Postprocessing after unsharp masking.

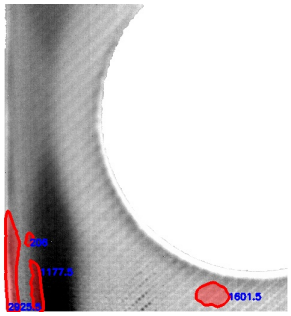
tant. This technique can increase either sharpness or local contrast. Indeed, they are both forms of differences, respectively small-scale (high-frequency) and large-scale (low-frequency), the latter of which are of concern. They can be increased using high radius and amount. Good values are 99 px for r and 25 for a. The third parameter, the threshold th, is not as crucial as the other two and can be set to 0.

Rather, the thresholding of S is of interest since only dark regions are to be kept, see Figure 5.3.3(b). A good value is 180.

The post-processing is a final check useful for removing possible remaining artifacts characterized by small contours. The function retrieves contours using the algorithm described in [148], already implemented in OpenCV. The final result looks like Figure 5.3.3(c) and provides the region of interest for the comparison in the next step.



(a) Comparison of a frame (upper right) with its reference (upper left) and their masks (bottom).



(b) Zoom on defective areas, which are displayed in red and labeled in blue with their area expressed in number of pixels.

Figure 5.4.1: Frame comparison to find defective areas.

5.4 Pairwise Comparison of Adhesive Bondings for Difference Detection

This step uses the masks built in the previous step in order to restrict the comparison to a region of interest with glue to minimize false positives and false negatives. Let A and B be the two images to be compared and M the union of the two masks obtained following the procedure explained in the previous step. First of all, a Gaussian or median filter is applied to obtain both C and D images. This blur should be soft (e.g. Gaussian with radius 29 px) otherwise differences in the shapes of the layers could be lost. Then, images C and D are masked with M obtaining images E and F, respectively; furthermore, a dissimilarity image is obtained by subtracting E and F. Finally, only differences in intensity greater than a threshold (e.g. 25, depending on the desired sensitivity) are pointed out. Their areas are also calculated: they are displayed in red in Figure 5.4.1, in which areas are also reported (in blue), expressed in number of pixels or in 3D world coordinates.

Areas can be expressed in 3D world coordinates by means of the knowledge-based approach, as done for the background/foreground extraction. In this case,

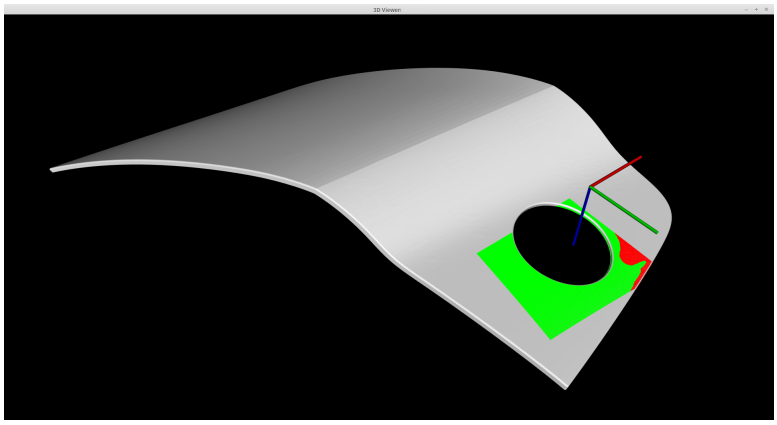
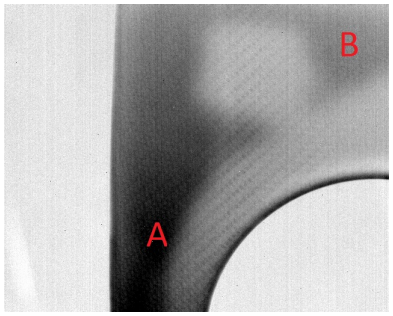


Figure 5.4.2: Backward projection of each pixel to the 3D model. The part area without defects is in green while the defective area is in red.

thanks to the knowledge of the position of the camera with respect to the robot and the position of the robot with respect to the part, the four vertices of each pixel belonging to the foreground are back-projected to the 3D model and the area of the back-projected pixel is approximated by the area of a parallelogram in 3D world coordinates. Thus, the area of the defective region can be calculated as the sum of many parallelograms, each of which approximates the area covered by a pixel. Figure 5.4.2 provides an example of frame back-projected to the 3D model: the part area without defects is in green while the defective area is in red.

5.5 Experimental Results

This section deals with the application of the methods proposed in the previous section on a dataset provided by one of the project partners, Trimek S.A., Spain. It contains 85 PPT phase images of 5 CFRP side blades from 17 points of view which cover the entire perimeter of the part. The frequency at which the PPT is evaluated depends on the acquisition duration and needs to be fine enough to enable discrimination of each signal of interest [145]. In our case, a good value was found to be 0.04 Hz. One of the 5 CFRP side blades is supposed not to have any



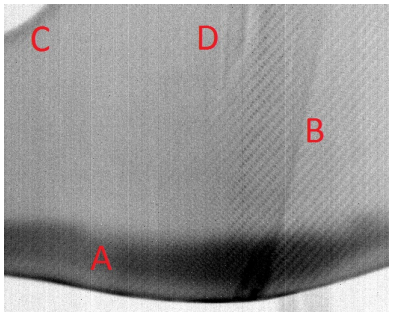
(a) Example of frame with darks regions of different brightness, shape and size labeled by the letters A and B.

(b) Corresponding mask: different kind of dark regions are detected.

Figure 5.5.1: Examples of different dark regions which are handled correctly.

defects and is taken as a reference for the comparisons so that each image can be analyzed and compared with the respective defect-free reference. In the following, the results are discussed: first of all, some remarks about the segmentation of the glue layer; second, the results of the 68 comparisons.

The detection of adhesive bondings in the extracted foreground must face with many types of dark regions with respect to the surrounding and light-dark transitions. There are some examples of dark regions of different brightness, shape and size labeled with the letters A and B in Figure 5.5.1(a) and transients with different brightness, shape, size and sharpness labeled with the letters A, B, C and D in Figure 5.5.2(a). As shown in Figure 5.5.1(b) and 5.5.2(b), the proposed method can successfully detect several combinations: i) large or tapered dark regions, ii) dark regions and brightness transitions along the contours of the part, iii) sharp transitions due to local dishomogeneities in the glue layer. If contrast were adjusted globally, many details, which can be captured by the proposed local method, would instead be lost. Figure 5.5.3 provides the comparison of the two methods: the first image (a) is the starting PPT phase image, the second (b) its global ad-



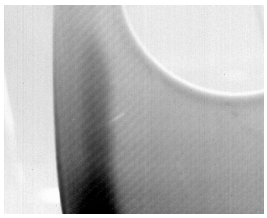
(a) Example of frame with transients of different brightness, shape, size and sharpness labeled by the letters A, B, C and D.

(b) Corresponding mask: different kind of transitions are detected.

Figure 5.5.2: Examples of transients which are handled correctly.

justment and the third (c) the mask obtained by USM.

To evaluate the performance of the whole system, the number of False Positives (FPs) and False Negatives (FNs) has been measured. Let A and B be the two images being compared, M the union of the two masks denoting the glue layers and D the detected defective areas: FPs may be caused by the noise or the carbon fibre pattern, both of which may differ in A and B. In addition, if the segmentation is not selective, M is large and areas without glue are compared potentially leading to FPs. Conversely, FNs occur if M does not completely cover the regions with glue. If those areas were different, they would not be revealed. In addition, given the experimental setup, the proposed method strongly relies on the assumption that the two parts are aligned. If this condition is not met in practice, there could be FPs or FNs. Two metrics are proposed here: $|M|$, which is the number of false positive or false negative pixels normalized by the number of pixels in M, and $|D|$, the same applied to D. The average values over the 68 comparisons and the worst/highest are reported in Table 5.5.1. The best/lowest percentages are zero. Regarding false positives, the algorithm proves to work well with the exception of some ambiguous situations as that shown in Figure 5.5.4, in which the reference



(a) This is the starting PPT phase image.

(b) This is its global adjustment.

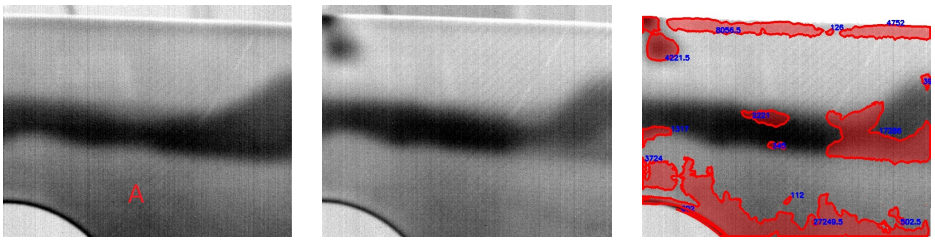
(c) This is the mask obtained by unsharp masking.

Figure 5.5.3: Local differences, captured by local contrast adjustment, may be lost by adjusting contrast only globally.

Table 5.5.1: System performance in terms of False Positives (FPs) and False Negatives (FNs). The number of FPs and FNs is normalized by the number of pixels in the union of the two masks $|M|$ or in the detected defective areas $|D|$. The average percentages over the 68 comparisons are reported in the first row, highest/worst percentages in the second.

$FP/\|M\|$	$FP/\|D\|$	$FN/\|M\|$	$FN/\|D\|$
0.026	0.108	0.004	0.036
0.210	0.659	0.060	0.585

frame (a), the second frame (b) and the output frame (c) are reported. In few comparisons, it can be hard to tell whether the ground truth or the algorithm is wrong. The region labeled with letter A in Figure 5.5.4(a) is classified as defective but the ground truth does not consider that region as part of the glue layer even if it might be considered darker with respect to the surroundings. That region is quite large and causes high scores with both metrics. False negatives are instead very rare given that the segmentation phase works well and the algorithm robustly takes into account the union of the two masks. In Figure 5.5.5, another comparison is reported. Given that the algorithm works well with the exception of few ambiguous situation, performances can be considered adequate for a real life application.

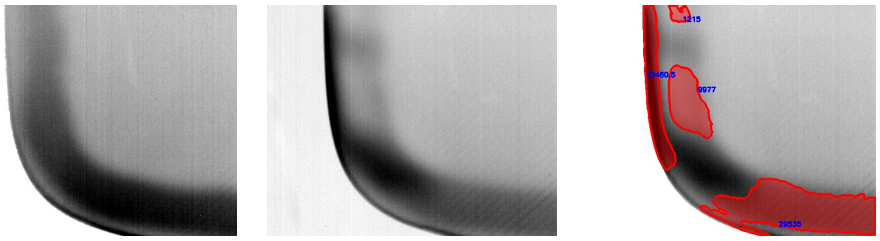


(a) Reference PPT phase image: region A is dark.

(b) Second frame.

(c) Result of difference detection.

Figure 5.5.4: Example of False Positives.



(a) Reference PPT phase image.

(b) Second frame.

(c) Result of difference detection.

Figure 5.5.5: Example of successful comparison.

The proposed method is very efficient in terms of computing time. The bottleneck is the post-processing, which finds small contours by border following and eliminate the respective small artifacts from the mask. On a common laptop with Intel Quad Core i5-3337U CPU clocked at 1.80 GHz, 8 GB of RAM and Windows 8.1, the average required time per image is 0.94 s, which is below the time required for the acquisition of the next PPT phase image, approximately 30 s.

5.6 Summary

This chapter presented an autonomous robotic system for thermographic segmentation of defective areas in the superficial inner bonding layers of CFRPs taking a

step forward compared to ongoing research. Indeed, previous work focused on the thermographic process, instead of the automatic thermo-image analysis. The proposed system uses a NDT&E technique named PPT, whose acquisition setup consists of a flash lamp and an infrared camera. The main point is the automatic thermo-image analysis, in particular the pairwise comparison of the acquired PPT phase images by means of automatic image processing techniques. It is performed in two steps: the segmentation of the adhesive bondings, which defines a region of interest, and the comparison of the pair of PPT phase images. Given the lack of a big dataset, we could not apply machine learning as in Chapter 2 and 3. Hence, we devised a novel segmentation algorithm based on local contrast adjustment via unsharp masking. This approach is capable of dealing with ISO or vertical banding noise and artifacts due to the typical pattern of CFRPs, thanks to the use of context and pre and post-processing phases. Both segmentation and comparison steps take advantage from the high level of knowledge of the experimental setup, which, by means of the back-projection from 2D to 3D introduced in Section 1.2 and exploited in the previous chapters, allows to reduce the number of false positives and to calculate the extent of the defective areas in world coordinates. Many tests have been performed on PPT phase images taken from a number of points of view such that the entire perimeter of five CFRP side blades is covered. The proposed method has proven to be effective and efficient.

6

Conclusions and Outlook

Visual scene understanding is still much more challenging for robots than for humans. In this thesis, we investigated over the semantic segmentation problem showing that motion and contextual cues can make it simpler for robots. In particular, we showed that single-view predictions can be integrated in a full 3D model in different scenarios, providing scene models for autonomous navigation and object models for autonomous quality inspection. However, despite all the advancements, it is still hard to deploy such systems in real scenarios. These techniques may not be robust enough to satisfy accuracy or safety requirements and, in order to work well, machine-learning based ones may require huge labeled datasets to be acquired, which require a huge workforce.

6.1 Summary

Models of scenes and objects can be enriched with the scene and object properties of interest, which can range from the color information in the most basic scenarios to material, objects or scene structures in the most sophisticated ones. Even if humans can easily perceive these properties, building these models manually can be impossible or tedious, most of all if accurate measurements are required. Furthermore, if the sensor moves, on the one hand, static methods may not work and, on the other hand, fusing the contributions of the multiple available view points in an effective way is non-trivial. We tackled these problems investigating the best ways to build scene and object models in an automatic and accurate way, fully exploiting the moving vision sensors mounted on service and industrial robots, like RGB-D, RGB or infrared camera sensors.

In this thesis, we built accurate scene models of indoor environments facing with the problem of learning the semantic labels from the perceived 3D structure. We proposed a batch approach, and a novel multi-view frame fusion scheme working in an incremental fashion and accumulating the single-view results. It allows the online multi-view semantic segmentation of single frames and the offline reconstruction of semantic maps. Our experiments showed the superiority of our approaches, which lead to a more accurate semantic labelling. Also, we developed a novel semantic segmentation approach based on the detector You Only Look Once (YOLO), which can further boost performances in combination with the 3D Entangled Forest classifier. We outperformed state-of-the-art approaches on several semantic classes. Finally, we proposed a novel approach inspired by semantic segmentation and mapping techniques to detect fallen people from a mobile robot with high recall and precision.

In addition, we built accurate object models of Carbon Fiber Reinforced Polymers, like car components, devising novel techniques for the image registration and mapping of carbon fiber orientations to 3D models with translational or 6-Degrees-of-Freedom robot motions, enabling the use of sensors that need multiple shots to perform a measurement in real-time and continuous motion. These

algorithms for multi-view reprojection takes inspiration from the multi-view frame fusion scheme for semantic segmentation and mapping. In this task, we succeeded in reaching an industry grade accuracy. Furthermore, we studied how to include not only the information on the superficial fiber orientations but also the inner defects due to the wrong distribution of the thermosetting polymer in the upper layers. In both cases, the robot work-cell needed to be calibrated with standard but tedious manual techniques. Thus, we presented also a fully automatic robot hand-eye calibration procedure not requiring any human intervention but capable of achieving the same performance level.

Many results of this research were successfully applied in the context of several national and international projects at the Intelligent Autonomous Systems Laboratory (IAS-Lab): the European projects COROMA, FibreMap and ThermoBot, the regional project Omitech, see Appendix A, and the international Mohammed Bin Zayed International Robotics Challenge (MBZIRC), see Appendix B.

6.2 Outlook

6.2.1 Service Robotics

Home robots are making their way into our homes. New products like Softbank's Pepper have been introduced into the market and many research platforms, e.g. the healthcare robots Pearl [119], ASTRO [120], Max [121], Hobbit [122] or our prototype O-Robot [103], have been proposed. Such robots aim at being friendly companions able to enhance our lives. Anyway, even if they do not need to satisfy the same precision required to industrial robots when performing manufacturing tasks, their actual deployment is not straightforward since service robots have to autonomously and safely navigate into the environment facing with potential changes and interacting with humans.

In this thesis, we focused on semantic segmentation and mapping techniques to find objects and understand the scenes, which is necessary to make their interaction with humans possible. The machine learning methods at their basis work if

properly trained. Training requires huge labeled datasets, whose acquisition and annotation require a huge workforce. To avoid manual annotation, not only semi-automatic annotation tools [111] but also online learning schemes [149] are being studied. Moreover, semantic models should include much more properties so as to make human-robot interaction easier and more fluent. How to include more object classes or object parts and how to model the relationship between the detected elements are still interesting problems to be addressed. Researchers are also working on new ways for integrating techniques from 3D reconstruction with recognition and learning [150]. Indeed, given the advances in both fields, how semantic information can be used to improve the dense matching process in 3D reconstruction techniques and how valuable is 3D shape information for the extraction of semantic information are interesting questions needing further exploration.

6.2.2 Industrial Robotics

With the use of industrial robots, product quality, production times, safety and savings can be significantly increased. With them, humans can be freed from tedious tasks like inspection and dangerous or heavy ones like manufacturing. Nevertheless, many tasks and sub-tasks have to be automatized in order to achieve fully automatic systems. In this thesis, we focused on the automation of the scanning process in two scenarios with carbon fiber parts, whose non-destructive testing and evaluation were of interest. Anyway, the fully automatic inspection would require also to detect and localize these parts, which is currently performed with the aid of humans. Furthermore, in general, other manufacturing tasks such as drilling, deburring, trimming, polishing or sanding, involve also the localization and subsequent grasping of the proper tools. As with service robots, industrial robots may have to autonomously and safely navigate into the environment facing with potential changes and interacting with humans.

As shown in Chapter 4 and Chapter 5, in a typical industry production plant, the robot is installed in a work-cell and the localization of a part to be scanned is performed via manual standard calibration techniques, which are tedious and

prone to errors. First, a tool tip is mounted on the robot end-effector and calibrated, e.g. with the 4-points method described in robot vendor manuals. Then, the part under inspection is localized by manually touching three reference points on it with the tip. Besides being a manual procedure, these points are hard to be determined in case of objects with complex shapes, i.e. without well-defined edges nor vertices, and must be correctly identified also in the respective CAD models. To make inspection or manufacturing tasks fully autonomous, we need robots out of work-cells, able to autonomously move in the factory. Thanks to the EU project COROMA introduced in Chapter 2, we are delving deeper into this application with the aim of bringing autonomous robotics into difficult industrial tasks. How to combine visual scene understanding techniques like semantic segmentation and object recognition in such scenarios, and how to infer the correct decisions from this semantic knowledge are interesting problems not yet addressed and matter of research.

Appendices

A

O-Robot: an Autonomous Robotic Platform for Ambient Assisted Living

In this appendix, we present our contributions to the development of O-Robot, an autonomous robotic platform for home care. This project was developed in collaboration with Omitech Srl, Italy[1] and Kynetics, Italy[2] with the aim of paving the way to the concept of Robotics-as-a-Service and creating a set of artificial intelligence services for house patrolling and human-robot interaction, in which the human is the elderly but also the remote user, e.g. a caregiver. This project is motivated by that fact that last years have seen a worldwide lengthening of life expectancy [123] and, as a consequence, an increment of advanced assistive solutions for integrated care models. In this context, Information and Communications Technology (ICT) can enhance home assistance services for elderly people and thus, the economical burden for health-care institutions. Indeed, recent studies report that 89% wish to stay at home for sentimental reasons or because they cannot afford nursing homes [151]. In addition, the shortage of professional

[1] https://www.omitech.it/
[2] https://www.kynetics.com/

caregivers is a well-known issue and relatives or friends must face with emotional distress negatively impacting also their productivity at work [152]. In this perspective, home robots will play a crucial role. Not only they will keep the house safe by monitoring and detecting anomalies or sources of hazards but they can also be companions able to enhance their social life, e.g. by better connecting them with their relatives and friends. Examples of recent projects that have tried to develop such a kind of systems are Hobbit [153], Astro [120, 154] and Giraffplus [155]. The difficulties in creating a robust and reliable prototype and the encountered challenges in computer vision and autonomous robotics have been well-explained in [153].

The remainder of the appendix is organized as follows. The prototype and the main software components are described in Section A.1 and A.2. Our main contributions are described in Section A.3. Finally, in Section A.4, conclusions and lessons learned are reported.

A.1 Hardware Configuration

Our prototype of home robot, shown in Figure A.1.1, is built on top of a commercial open-source mobile platform, the Turtlebot 2[3], which is already equipped with odometry, a gyro, bumpers, cliff sensors, wheel drop sensors and a docking IR receiver. Furthermore, this platform is a smart choice also because of the available API to interface with ROS and the existence of a worldwide community working with it. For our purposes, the robot control is accomplished by the Lenovo Y50-70 laptop with Ubuntu 14.04 and ROS Indigo. Furthermore, we have added an Hokuyo URG-04LX-UG01 2D scanning laser rangefinder and a Microsoft Kinect v2. The former is placed at a height of 15 cm so as to easily detect low obstacles; the latter at a height of almost 115 cm which is the best trade-off for the people detection at several different distances.

A.2 Software Modules

The robot software, see Figure A.2.1 for an overview, is mainly based on ROS. The main task is the house patrolling, i.e. the complete periodic visit of the house, which is implemented by means of the TurtleBot ROS packages for navigation,

[3] http://www.turtlebot.com/

Figure A.1.1: Hardware overview. The robot is built on top of the Turtlebot 2. We have added a Lenovo Y50-70 laptop, an Hokuyo URG-04LX-UG01 2D scanning laser rangefinder and a Microsoft Kinect v2.

mapping and localization. During this task, it can detect fallen people with our algorithm presented in Chapter 3. Additionally, it can detect and track people [156–158] and also recognize their faces [159]. It can also perform speech recognition by means of Google APIs[4] and remind the elderly to take a medicine periodically. Given that elderly people could suffer from severe physical disabilities and be unable to move, the robot can be controlled also by means of a non-invasive Brain-Computer-Interface. Indeed, controlling a mobile device by using human brain signals might improve their quality of life facilitating their interaction with relatives and friends located in different rooms. Furthermore, the robot can connect to a remote user, e.g. an elderly relative or caregiver, for reporting dangerous events like the detection of a fallen person, and, in a future implementation, for a call or

[4] https://cloud.google.com/speech/

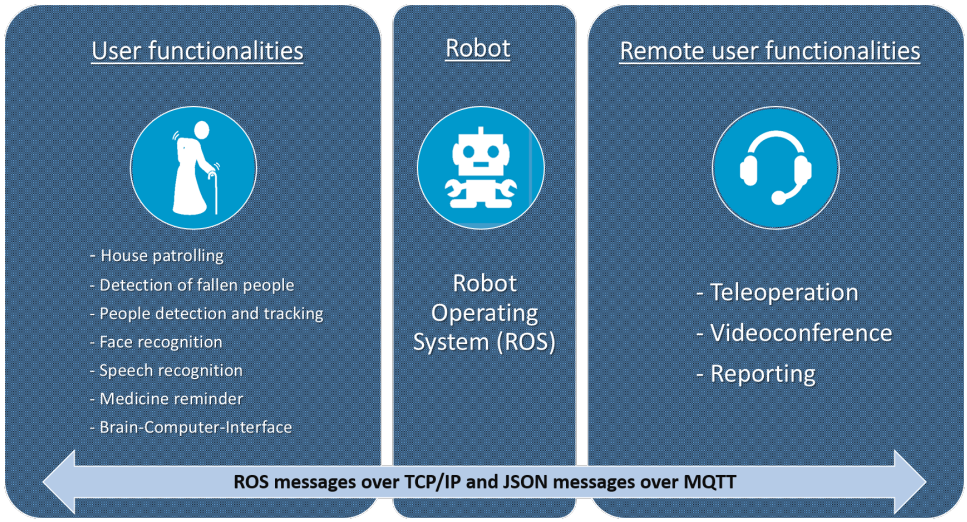


Figure A.2.1: Overview of the software modules implementing user and remote user functionalities.

a videoconference. The communication is implemented by means of JSON messages over the Message Queue Telemetry Transport (MQTT) protocol. Finally, the remote user can take the robot control so as to teleoperate it.

A.3 Main Contributions

Besides the hardware integration, we contributed developing the modules described in the following.

A.3.1 House patrolling

The robot periodically visits a list of predetermined way-points in the house. We adopted a state-of-the-art navigation approach based on a 2D static map of the environment, acquired by means of the 2D laser sensor. This map is built with the GMapping algorithm[5] [126, 127], which can create a 2D occupancy grid map from laser and pose data collected by the mobile robot. The robot localizes itself in the map with the Adaptive Monte Carlo Localization (AMCL) algorithm[6] [128]. Each way-point is reached by means of the ROS-package `move_base`[7], which

[5] http://wiki.ros.org/gmapping
[6] http://wiki.ros.org/amcl
[7] http://wiki.ros.org/move_base

links together a global and local planner to accomplish the navigation task.

A.3.2 Fallen person detection

This functionality has been extensively presented in Chapter 3. Our main contributions are recapped in the following:

- a real-time pure-3D approach suitable for real cluttered scenes;
- its integration with two basic robot functionalities, 2D mapping and navigation, in order to suppress false positives thanks to the a-priori knowledge of the environment and the availability of multiple view points;
- our RGB-D dataset of fallen people[8] consisting of several static and dynamic sequences with 15 different people acquired in 2 different environments.

A.3.3 Shared Control with Brain-Computer-Interface

Brain-Computer Interface (BCI) technology relies on the real-time detection of specific neural patterns in order to circumvent the brain's normal output channels of peripheral nerves and muscles [160] and thus, to implement a direct mind-control of external devices, e.g. mobile robots. In these systems, neural signals are recorded by non-invasive techniques, in our case Electroencephalography (EEG), and then, task-related brain-activity is translated into few commands, in our case discrete, to make the robot turn right or left. Despite the low number of commands provided by non-invasive BCIs (only two), driving mobile devices is possible even in complex situation with the help of a shared control approach [161–163]. The shared control approach is based on a seamless human-robot interaction in order to allow the user to focus his/her attention on the final destination and to ignore low level problems related to the navigation task (i.e., obstacle avoidance). We contributed in the development of a new ROS-based shared control system [164].

Although ROS is becoming the standard de facto for controlling different types of devices in the robotics community, it is still far to be a standard adopted in the BCI community and ad-hoc implementations are preferred. We explored a new ROS-based algorithm for semi-autonomous navigation and obstacle avoidance, able to make the shared-control safer and more reliable. For the map building process, two different methods have been exploited: GMapping [126, 127] and OctoMap [165]. GMapping builds a 2D occupancy map of the environment,

[8] http://robotics.dei.unipd.it/117-fall

while OctoMap creates a 3D scene representation, which can be down-projected to the ground so as to enrich the 2D occupancy map with higher obstacles visible by the RGBD sensor but not by the laser. This way, the localization module, AMCL [128], is based on the map built with GMapping, while the navigation one is based on the 2D down-projected map because of the richer representation of the environment. This way, the trajectory planner can take into account high obstacles and avoid collisions. The navigation algorithm is built on top of the ROS navigation packet `move_base`. It implements the following behaviour: the robot goes straight and turns right or left as soon as it receives a command from the BCI. The navigation goal is set at a fixed distance from the robot, e.g. 0.5 m, in front of the robot, at its right or left. In order to avoid unreachable navigation goals, we introduced a further check on the static map so as to discard navigation goals on the occupied map cells. It comprehends also a simple untrapping procedure making the robot turn of a fixed angle, e.g. 10°, and go behind.

In real experiments, this method proved to reduce user's cognitive workload, decreasing the number of commands necessary to complete the task and helping the user to keep attention for longer periods of time. The ratio between the number of commands in the two modalities (BCI with shared control and the manual without shared control) was 80.9% and the ratio between times was 114.5%. This means that without shared control, the BCI user has to send more commands to the robot, increasing the necessary cognitive workload.

A.4 Conclusions and Lessons Learned

In this appendix, we have presented O-Robot, an autonomous robot prototype for elder care assistance. Despite its apparent simplicity, this robot enabled research in many different fields leading to the development of algorithms which could be applied also to other robots. For a safer navigation, the next prototype should include sonar sensors able to work in the presence of glass surfaces, which are typically present in houses and offices but cannot be reliably sensed with low cost laser sensors. Furthermore, it would be interesting to add a robotic arm and a robotic hand, and integrate the semantic segmentation and mapping techniques presented in Chapter 2 to further boost human-robot interaction, e.g. for manipulation actions. Another robot comprehending a robotic arm and a hand is presented in the next appendix.

B

RUR53: an Unmanned Ground Vehicle for Navigation, Perception and Manipulation

This appendix describes our main contributions in the development of RUR53, the unmanned mobile manipulator robot of our team, the Desert Lion team, which competed at the Challenge 2 and the Grand Challenge of the first Mohamed Bin Zayed International Robotics Challenge[1] (Abu Dhabi, March 2017). We ranked third in the Gran Challenge in collaboration with the Czech Technical University in Prague, Czech Republic, the University of Pennsylvania, USA, and the University of Lincoln, UK. According to the competition requirements, the robot is able to freely navigate inside an outdoor arena; locate and reach a panel; recognize and manipulate a wrench; use this wrench to physically operate a valve stem on the panel itself. RUR53 is able to perform these tasks both autonomously and in teleoperation mode. This appendix provides an overview of the adopted hardware and software architectures with a focus on our contributions.

The remainder of the appendix is organized as follows. Section B.1 introduces the Mohamed Bin Zayed International Robotics Challenge. Section B.2 provides

[1] http://www.mbzirc.com/

an overview of the robot, in particular both the hardware and software architecture. Section B.3 focuses on our main contributions in the perception and teleoperation modules. Finally, Section B.4 describes our experience and the lessons learned during the Challenge.

B.1 The Mohamed Bin Zayed International Robotics Challenge

The Mohamed Bin Zayed International Robotics Challenge (MBZIRC) is an international robotics competition fostering research in advanced robotics applications like operations in disaster scenarios, oil and gas maintenance, manufacturing, construction, and housework. The first edition took place in Abu Dhabi in March 2017 and consisted of three challenges and a triathlon type Grand Challenge. Challenge 1 required an Unmanned Aerial Vehicle (UAV) to locate, track, and land on a moving vehicle. Challenge 2 required an Unmanned Ground Vehicle (UGV) to locate and reach a panel, and physically operate a valve stem on the panel. Challenge 3 required a team of UAVs to collaborate to search, locate, track, pick and place a set of static and moving objects. The Grand Challenge required a team of robots (UAVs and UGVs) to simultaneously compete in the combination of Challenges 1, 2, and 3.

B.2 Robot System Design

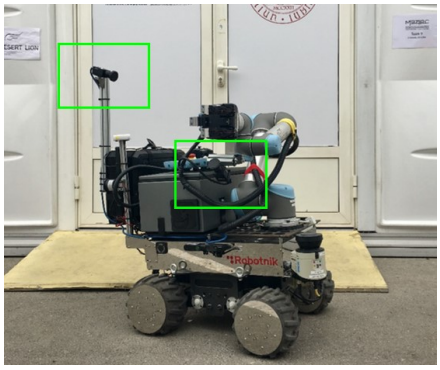
The robot RUR53 is an UGV able to operate in autonomous mode as well as in teleoperated mode. Indeed, despite the innovation introduced by autonomy, human intervention through teleoperation guarantees the possibility to recover the robot routine when facing with unforeseen situations or sensor failures. For each modality, the description of both the hardware and software architecture follows.

B.2.1 Hardware Architecture

The robot is visualized in Figure B.2.1. It is composed of a mobile Summit XL HL (Robotnik Automation S.L.L.[2]). This choice is a good trade-off between speed (up to 10.8 km/h), autonomy (up to 3 hours), cost, customizability and payload (up to 65 kg). The mobile base has been modified in order to install, on its top, a Universal

[2] http://www.robotnik.eu/

(a) RUR53 configuration in the autonomous mode.



(b) RUR53 configuration in the teleoperation mode.

Figure B.2.1: RUR53 hardware configuration in the arena. Cameras are surrounded by green boxes.

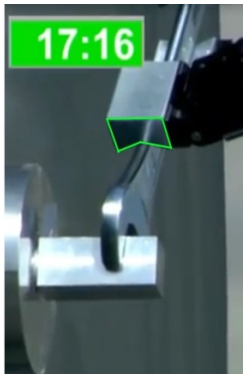
Robot UR5 manipulator arm (Universal Robots A/S[3]), equipped with a Robotiq Adaptive Gripper with 3 fingers (Robotiq[4]). It is called RUR53 because of the initials of each robot part manufacturer (**R**obotnik Summit XL HL - **U**niversal Robot UR**5** - **R**obotiq **3**-Finger Adaptive Gripper). These hardware components are all fully supported by ROS.

The Summit XL HL The Summit XL HL is equipped with a JETWAY JNF9J-Q87 board with 4 GB RAM. This board cannot provide the necessary computational power; for this reason, the robot comprehends an external computer with an i7-6700 CPU, 16 GB of RAM and a dedicated NVidia GTX 1070.

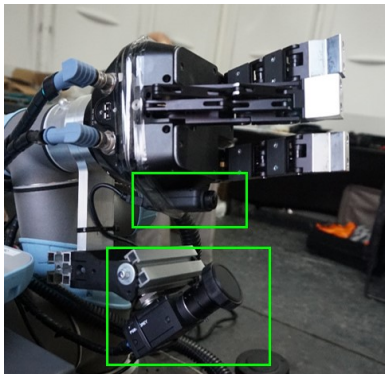
The manipulator arm has 6 DOFs, a payload of 5 kg at most, and a working radius of 850 mm. As the manipulation tasks involved lightweight objects, a cheap, lightweight robot, with a sufficient working range is the best choice. The gripper has a weight of 2.3 kg and generates a grip force in pinch mode ranging from 15 to 60 N $\pm$ 15%, with a maximum recommended payload of 10 kg. Its closing speed ranges from 22 to 110 mm/s. Having a 3-finger gripper permits the grasp of objects with different geometry and dimension, guaranteeing the generality of the implemented manipulation approach. In order to better grasp the wrench, the finger ends are customized (see Figure B.2.2(a)) by means of a groove able to canalize the grasped tool.

[3] http://www.universal-robots.com/

[4] http://robotiq.com/



(a) Custom Robotiq 3-finger gripper. The picture is taken from the match video.



(b) Gripper cameras in the teleoperation mode.

Figure B.2.2: Detailed view of RUR53 hardware configuration in the arena.

In order to perform obstacle avoidance and retrieve the panel, the mobile base is equipped with two 2D laser scanners (SICK Sensor Intelligence[5]) with an aperture angle of 270°, an angular resolution of 0.25°- 0.5°, and an operating range of 0.5-50 m. The combination of these sensors lets us inspect all the arena.

Visual sensing is performed by means of two Grasshopper3 2.8MP Mono USB3 cameras (Point Grey Research, Inc.[6]), which are more reliable in the Challenge environment with respect to low-cost RGB-D sensors, which are more sensitive to the outdoor light. The only characteristic that makes the robot autonomous configuration different from the teleoperation one is the location of the cameras. For the autonomous mode, the two Grasshopper3 cameras are located next to each other on a fixed support mounted above the gripper (see Figure B.2.1(a)) so as to guarantee the tool recognition even if the end-effector is very close to the panel. During teleoperation, instead, the pilot has to be perfectly aware of the robot state and of the environment so we mounted one camera on the mobile base (see Figure B.2.1(b)) for a panoramic view of mobile base, arm and environment, and two cameras under the gripper to clearly see the hand and the grasp with an inside-out perspective (see Figure B.2.2(b)).

[5] http://www.sick.com
[6] http://www.ptgrey.com/

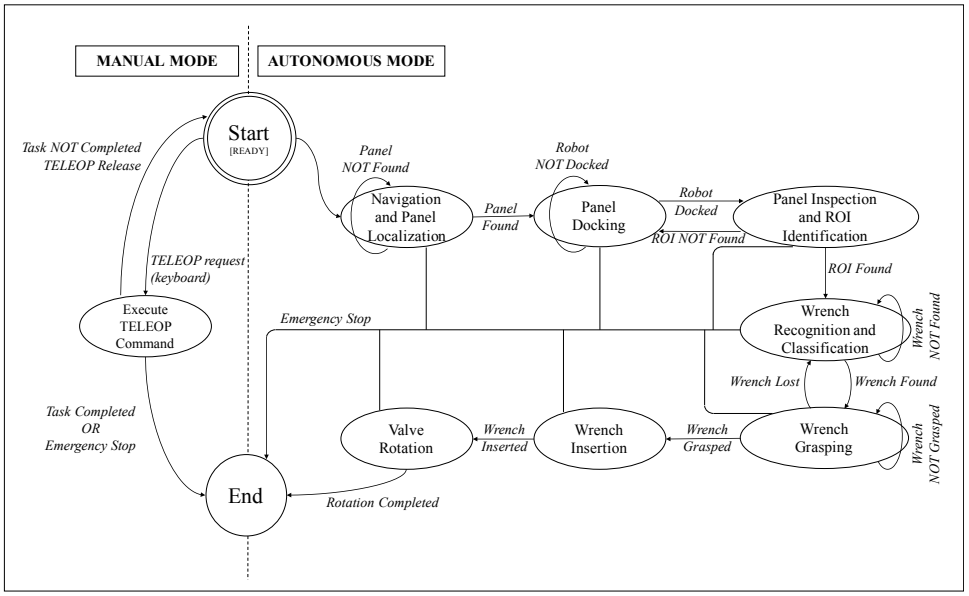
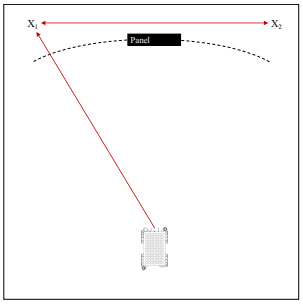


Figure B.2.3: RUR53 Finite State Machine.

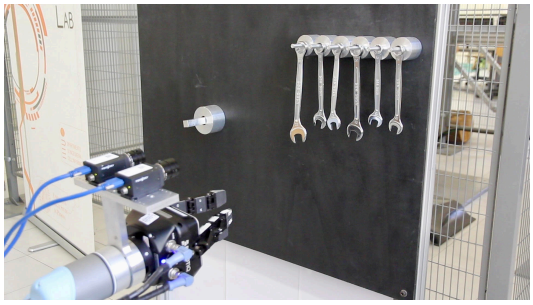
B.2.2 Software Architecture

RUR53 operation mode is formally modelled as a Finite State Machine. Two main modes can be identified: the autonomous mode (Figure B.2.3 - *right*) and the teleoperated one (Figure B.2.3 - *left*).

In the autonomous mode, RUR53 navigates within the arena searching for the panel. As shown in Figure B.2.4(a), the panel could be located along a curve in the farthest side of the arena and the presence of obstacles cannot be excluded. The panel detection takes in input the laser scan and consists in the detection of a line segment of a fixed known length. Once the panel is detected, the robot approaches and inspects it searching for two Regions of Interest (ROI), corresponding to the wrenches and the valve stem, respectively. The view of panel from the docking position is shown in Figure B.2.4(b). If no tool or wrench is found, RUR53 moves around the panel and inspects its other side. Otherwise, it examines the ROI in order to recognize and classify the right wrench.Once the wrench is found, RUR53 executes the actions managing the grasping and valve manipulation tasks. A recovery plan has been designed in order to manage the fault condition, called *wrench lost* in Figure B.2.3. In this case, RUR53 repeats the wrench recognition and grasp-



(a) The navigation path in the challenge arena and the way-points periodically visited until the panel detection.



(b) View of the panel from the docking position.

Figure B.2.4: Navigation and docking

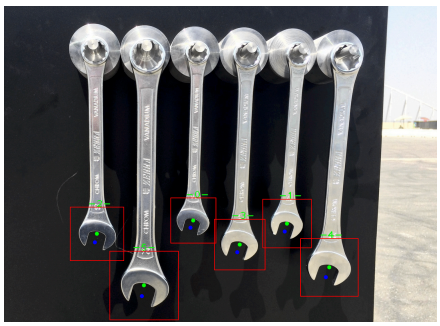
ing steps in order to detect another wrench suitable for the valve manipulation. According to specifications, only two wrenches could operate the valve: the recovery routine takes this information into consideration. An overview of the perception module output is provided in Figure B.2.6. Developing a module able to work under different illumination conditions and with metal surfaces is challenging. The detection of the wrenches, in particular their heads, and of the valve are performed by means of two Cascade of Boosted Classifiers trained on LBP features [59]. The pose estimations of the wrench and the valve are performed by means of a set of 2D/3D heuristic approaches able to segment the wrench, its head and the valve.

In the teleoperation mode, instead, velocity commands are sent to the robot (the mobile base, the robotic arm and the gripper) by means of the keyboard. Commands are executed until the task is completed or the teleoperation terminates.

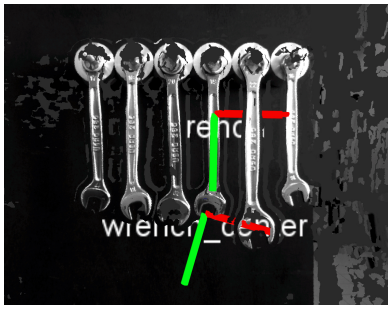
B.3 Main Contributions

B.3.1 Panel orientation estimation

Even if the robot is programmed to dock at a pre-determined position with respect to the panel, the perfect alignment of robot and panel is not guaranteed in practice. Hence, the actual docking angle between robot and panel, α, has to be measured and fed to the manipulation modules. We developed a panel angle measurement

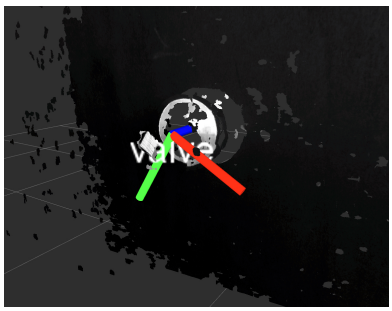


(a) Wrench detection on the arena panel.



(b) Wrench recognition and pose estimation.

Figure B.2.5: Navigation and docking



(a) Valve detection and pose estimation.

Figure B.2.6: Wrench and valve detection and pose estimation.

procedure which takes as input the laser scan and the robot transformation tree, and consists of four steps:

1. laser scan filtering;
2. line fitting with RANSAC [92];
3. calculation of α', which is the docking angle between 0° and 90°;
4. calculation of α, which is the docking angle in a wider and more useful range between 0° and 180°.

In particular, given that the robot position with respect to the panel is approximately known, part of the laser points framing the panel can be easily retrieved by

angle filtering. Then, a' can be calculated with the line-plane intersection formula, in which the line with coefficients (a, b, c) is the panel line estimated in the previous step and the plane with coefficients (A, B, C, D) is the robot side facing to the panel:

$$a' = \frac{Aa + Bb + Cc}{\sqrt{A^2 + B^2 + C^2}\sqrt{a^2 + b^2 + c^2}}. \tag{B.1}$$

In any case, the dot product of the plane vector and the line vector in Equation B.1 provides the complementary to 90°. Hence, to provide an angle in the wider range $[0 - 180]$ to the grasping module, a further step is required. First, the intersection between the panel line and the robot side plane is calculated. Then, the laser points are projected to the robot side plane. This way, we can distinguish between a' and $180 - a'$ by comparing the intersection point coordinates with the closest projected point coordinates.

Considering three different panel-robot configuration and about 220 laser scan measurements for each configuration, this angle can be estimated with an average error of 0.4° with respect to manual ground truth measurements and a standard deviation of 0.3°. These values are minor than the measurement uncertainty due to the manual ground truth measurements.

B.3.2 Grasp point estimation

As introduced in Section B.2.2, the perception module makes the robot capable of detecting and recognizing the wrench able to rotate the valve among the six randomly placed on the panel. Once it is recognized, another module can segment it so as to define the 3D grasping point on the wrench handle, see Figure B.2.6(b). This is achieved by means of the heuristic presented in the following.

This heuristic is based on a coarse wrench segmentation and allows to accurately find a 3D grasping point despite the heavy noise in the stereo-reconstructed point cloud due to the metallic surface of the wrenches and varying illumination conditions. Given the bounding box of the recognized wrench head, see Figure B.2.6(a), a 2D region of interest containing the wrench handle is determined by extending the region containing the head along the vertical axis. In particular, if the head bounding box is $bbox_{head} = (x, y, width, height)$, the bounding box of the handle is computed as $bbox_{handle} = (x, y - 2 \times height, width, 2 \times height)$. This bounding box is used to extract the respective 3D point cloud, which shall contain just part of the wrench points. Indeed, there is no need to segment all the wrench points. This point cloud is further process so as to remove possible outliers. First, strong outliers with a distance greater than 1 m with respect to the camera are re-

moved. Indeed, these are due to coarse errors of the stereo-reconstruction algorithm. Then, the mean distance from point to camera is calculated and points with a distance greater than 1.5 cm are considered outliers and left out. Finally, given this set of points, which are likely to be part of the wrench handle, the plane of the wrench handle is estimated by fitting a plane model by means of the RANSAC [92] algorithm with a distance threshold of 1 cm. Thus, given the remaining points, their centroid can be calculated and be the grasp point. The wrench is supposed to be parallel to the panel plane, whose inclinaton with respect to the robot is known thanks to the procedure described in the previous subsection.

For further robustness, the obtained values are further smoothed. In particular, the final values are calculated accumulating the single-view values over 10 frames and selecting the median among them.

We did not evaluate this module standalone. Nevertheless, we evaluated the wrench recognition and grasping routine by asking the robot to correctly recognize and pick up a target wrench for 50 trials. We hung random sets of six wrenches to the panel. These tools were selected among ten different types of wrench. Different sizes were considered. Their poses (positions and orientations) were modified at each trial as well as the type of the target wrench. We found out that the perception system correctly recognizes the wrench 92% of the time and picks it up 86% of the time. Unfortunately, 16% of the time, the grasp is not firm enough to operate on the valve.

B.3.3 Teleoperation

The teleoperation module includes three main software components:

- mobile base teleoperation;
- robotic arm teleoperation;
- gripper teleoperation.

These modules run in parallel and map robot movements and speed settings to different keyboard keys. This way, the pilot can control every robot movement by means of the keyboard only and the human-robot interface can be as efficient and capable as possible.

Our mobile base teleoperation is provided by the TurtleBot teleoperation package in ROS[7], which allows to control it in velocity and can be easily adapted to

[7] http://wiki.ros.org/turtlebot_teleop

work with our mobile base. This interface proved to be more effective than sending goals in position coordinates $(x, y, z, q_x, q_y, q_z, q_a)$ to the ROS navigation stack even if this can be done through the handy graphical interface provided by the rviz ROS package[8]. Indeed, this second modality does not allow the required precision during the panel docking phase when the pilot cannot see the robot because it is hidden by the panel itself.

The arm teloperation is provided by our own package. It allows to move the robotic arm, in this case its tool center point, with velocity commands with respect to a custom reference system, i.e. the arm base. This way, we could move the arm before grasping the wrench and to reach the valve. This module is also in charge of rotating the wrench around the valve. We performed the 360° turn of the valve with many small rotations of about 10°. The number of rotation degrees is settable online. This proved to be essential; we made smaller rotations (5°) when the grasp was not much firm and risked to be lost, while bigger rotations for firm grasps (15°). The Robotiq ROS industrial package[9] was used to open and close the gripper fingers.

Given that the working area was far from the team base station, where the pilot teleoperates the robot, and that both the wrenches and the valve were not facing towards it, the pilot was not able to clearly see the robot. Thus, we exploited the robot cameras for the task: two gray-scale cameras and an additional fish-eye color camera, the latter not being required in the autonomous mode. As already displayed in Figure B.2.1(b), one gray-scale camera was positioned in such a way that the arm and the environment were clearly visible. The other gray-scale camera and the fish-eye camera, see Figure B.2.2(b), were positioned just above the hand so as to see the hand and the grasp clearly with an inside-out perspective. Despite the lack of 3D data, having three different sources of feedback allowed the pilot to make the right decisions when accurately positioning robot, arm and gripper. A well-known limitation of video feedback is the bandwidth requirement. We overcame it by streaming compressed videos in Theora codec format.

B.4 Challenge Performances and Lessons Learned

In Abu Dhabi, during the first rounds of the challenge, the robot was unbalanced since, after the replacement of lithium batteries with heavier lead ones for technical issues, we almost reached the maximum mobile base payload. This complicated the autonomous navigation and panel detection routines: the laser scan plane was

[8] http://wiki.ros.org/rviz

[9] https://github.com/ros-industrial/robotiq/

not parallel to the ground. Thus, when searching for the panel, the barriers of the arena and the floor led to candidates of size comparable to the panel. Furthermore, our laser could not reliably sense the panel because of its appearance and the challenge environmental conditions. The material of the panel, its black color, and the high temperature (about 40°C) caused a really noisy detection of the panel laser scan: only part of the panel was visible and not from all the view points. Given these issues and despite the good performances of the autonomous perception system, we decided to face the Grand Challenge in collaboration with Czech Technical University in Prague, University of Pennsylvania, and University of Lincoln adopting the teleoperation mode. We ranked third.

C

A Fully Automatic Hand-eye Calibration

The hand-eye calibration problem consists in estimating the rotation and translation of the end effector or gripper of a robot, i.e. the hand, with respect to the camera mounted on it, i.e. the eye. A number of typical robot tasks like visual servoing [166], grasping and stereo vision requires the knowledge of this transformation or, at least, could highly benefit from it. This way, the robotic arm can move the camera with respect to a fixed reference system, e.g. the robot base or the workcell world reference. Unfortunately, in most applications the hand-eye transformation needs to be calculated several times during the life cycle of a system, thus requiring many manual interventions.

A well-known approach was proposed by Tsai et al. [135] in 1988. Such approach requires the user to acquire a large set of images (more than 50) framing a calibration pattern (usually a checkerboard) from different views and the corresponding set of robot poses. Its implementation is already available in ROS [22] and in the Visual Servoing Platform (ViSP) [167], a library for developing visual servoing systems. This task can be partially automatized by fixing the pattern position inside the robotic work-cell and saving a pre-determined robot motion. Even if this can be easily done with recent robotic arms, there are some drawbacks: it might not be possible to fix the pattern position, and this solution would not work after any change in the work-cell or robot-camera configuration. An automatic ap-

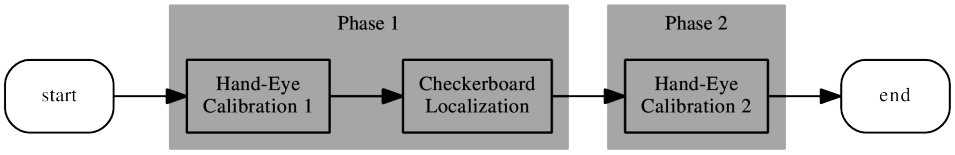


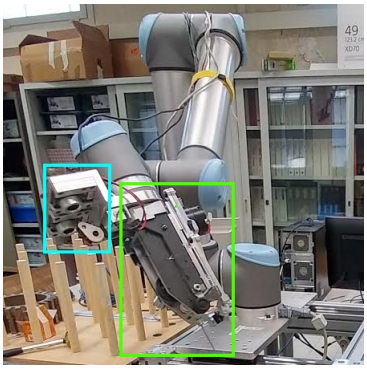
Figure C.0.1: The proposed two phase procedure. In the first phase, a raw hand-eye calibration is calculated with images grabbed while the robot is moving in the robot base or world reference system. Hence, the checkerboard can be localized. In the second phase, a refined hand-eye calibration is calculated using images grabbed with the robot moving in the checkerboard reference system.

proach automatizing the robot motion has been proposed by Tsai et al. [133] but, given that it requires the pattern calibration, i.e. the knowledge of the pattern position in the world, it shares the same drawbacks. Indeed, the pattern localization can be cumbersome. It is usually achieved by means of two standard procedures described in the robot vendor manuals: the calibration of a tip mounted on the end effector and the calibration of the work object, in this case the calibration pattern. The tip allows to accurately touch three points on the calibration pattern so as to define a Cartesian reference system on it.

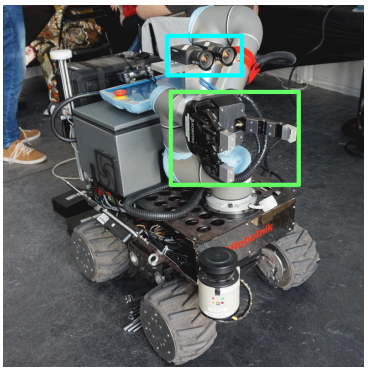
In this appendix, such standard approach is extended so as to perform both the calibration pattern localization and hand-eye calibration in a fully automatic way. This is a chicken-and-egg problem: to localize the pattern by means of the camera, the hand-eye calibration is needed; however, in order to automatize the hand-eye calibration, the pattern localization is also needed. Here, this is overcome by means of two phases, in each of which an hand-eye calibration is performed as shown in the overview in Figure C.0.1. In particular, our main contributions are:

- A fully automatic hand-eye calibration procedure, which does not require the knowledge of the calibration pattern 3D location;
- An open-source module[1] based on the framework ROS and the libraries ViSP and MoveIT! [168], which can be easily exploited on other robot platforms.

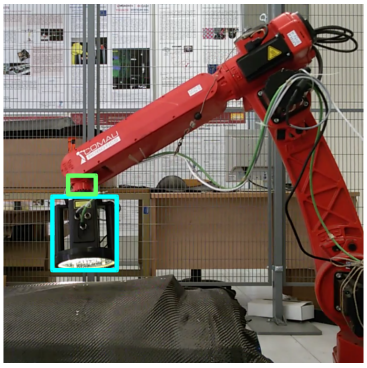
[1] http://robotics.dei.unipd.it/129-auto-hand-eye



(a) Hand-eye configuration for coil winding in EUROC.



(b) Hand-eye configuration in MBZIRC (Desert Lion Team).



(c) Hand-eye configuration in FibreMap.

Figure C.0.2: Three examples of hand-eye configuration: the hand is in the green box, the eye is in the cyan box. The robot setup in the following experiments is similar to that of the project EuRoC (a)

These results could speed up the research of the wide community working with robotic manipulators on complex and precise manufacturing tasks. In the challenge EuRoC[2] [18], the hand-eye transformation was necessary for visual servoing in tasks like car door assembly [169] and coil winding [170, 171]. In the challenge MBZIRC[3], many mobile manipulators were equipped with a camera on the robotic arm. The hand-eye transformation was necessary for grasping a wrench

[2] http://www.euroc-project.eu/
[3] http://www.mbzirc.com/

and rotating a valve stem. Furthermore, from our experience, the hand-eye transformation turns out to be useful also in other specific scenarios. In the context of the EU project FibreMap[4] [11], it allowed to inspect a carbon fiber preform in order to analyze its surface with a high accuracy and map carbon fibres on its model [10]. In ThermoBot[5], it allowed to subtract the background from thermographic images, thus improving the speed and reducing false positives when detecting defects in carbon fiber parts [12]. Different examples of robot hand-eye configurations are reported in Figure C.0.2. In our real experiments, the configuration in Figure C.0.2(a) has been adopted.

The remainder of the appendix is organized as follows. Section C.1 reviews the work related to the hand-eye calibration problem. Section C.2 describes our novel approach, first giving a picture of the entire workflow, then focusing on the two steps with an eye on the framework adaptability to other robots. Section C.3 deals with the system adaptability. In Section C.4, our methods are thoroughly evaluated in a simulated and a real environment. Finally, in Section C.5, conclusions are drawn and future directions of research identified.

C.1 Related Works

In the past, many methods have been proposed to perform the hand-eye calibration. They can be divided into two categories: approaches coupling hand-eye calibration with conventional robot kinematic model calibration like [172] and approaches decoupling hand-eye calibration from conventional robot kinematic model calibration like [173]. In [135], a new approach belonging to the second category is also proposed, which is faster and more accurate. This is now a common approach implemented in ROS and ViSP.

In [133], Tsai et al. discussed what are the main sources of error. In order to keep the errors low, they suggested to move the robot in different positions, the so-called stations, for each of which the robot is stopped and an image is acquired. They are chosen with a star-drawing technique giving a systematic way of generating an arbitrary number of view points with varying camera distances and angles. In our work, we used their calibration function and ensured the acquisition of images with varying distances and angles, but we did not implement their star-drawing technique. As previously discussed, even if the work by Tsai et al. has been a clear step towards the complete task automation, their procedure requires

[4] http://fibremap.eu/
[5] http://thermobot.eu/

the calibration pattern position to be known, in turn requiring the human intervention. This is overcome by our two phase approach.

More recently, in [174, 175], two extended hand-eye calibration approaches without the requirement of a calibration pattern have been presented. Indeed, in medical applications, an unsterile calibration pattern cannot be used. In [174], feature tracking and a structure-from-motion approach are exploited. This proved to be useful to simplify the calibration process, which has to be done before every surgical procedure. Nevertheless, this approach is not as accurate as the common one. In [175], the approach is claimed to be accurate. Anyway, it uses the surgery instruments with known CAD models as calibration objects. In our scenario, an object with a known CAD model is less practical than a checkerboard.

C.2 System Overview

This section focuses on the description of our automatic procedure. For ease of understanding, the main reference systems and transformations are depicted in Figure C.2.1. The symbols *B*, *H*, *C* and *P* are the initials of Base, Hand, Camera and Pattern, the names of the main reference systems. The symbols *BH*, *HE*, *EP* and *BP* stay for the respective rototranslations from the Base to the Hand, the Hand to the Eye, the Eye to the Pattern and the Base to the Pattern. Our goal is the estimation of *BP* and *HE* (in red).

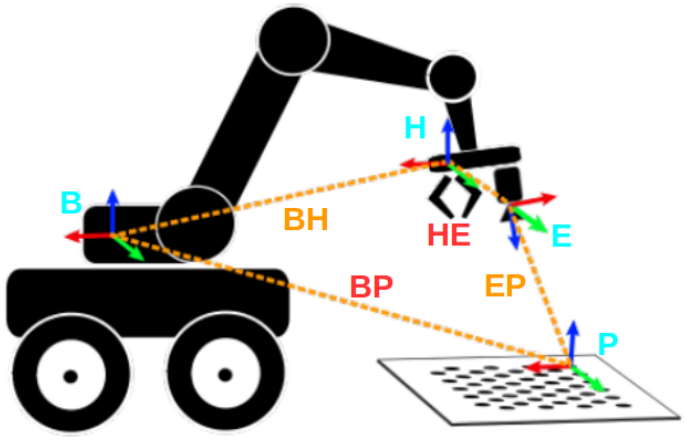
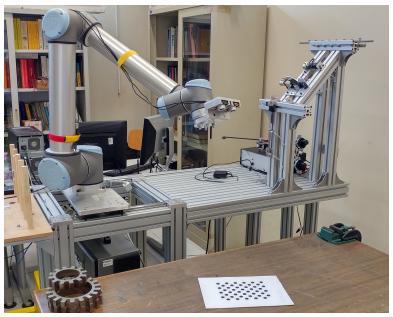


Figure C.2.1: Representation of the main reference systems (colored triads) and transformations (dot lines). Our goal is the estimation of the Base-Pattern (BP) and the Hand-eye transformation (HE).

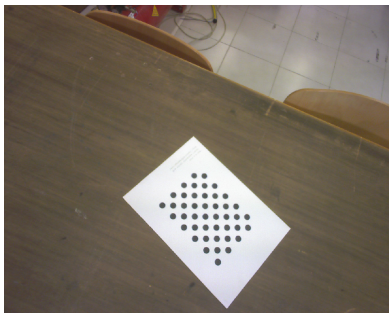
The automatic procedure can start after moving the robot to a starting position from which the calibration pattern is visible. Our calibration pattern is the asymmetrical circle pattern, currently supported in OpenCV [23], but could be also the classical black-white checkerboard with equally spaced squares. This procedure is characterized by two phases, displayed from left to right in Figure C.0.1. In the first phase, a raw hand-eye transformation (HE) is estimated from images grabbed while the arm is moving the hand (H) in the robot base or world reference system (B) with a stop-and-go motion. Indeed, the links BP and HE are both unknown so, instead of moving the camera (E) with respect to the the calibration pattern (P), we can only move the hand (H) with respect to the world/robot base (B). Thus, the actual path of the camera with respect to the checkerboard depends on the starting position. This means that, for instance, the actual number of acquired images framing the calibration pattern and the number of angles cannot be controlled, impacting the quality of the estimated hand-eye transformation, which, as a result, is also influenced by the starting position. Nevertheless, thanks to this first hand-eye transformation, the checkerboard (P) can be roughly localized. Thus, in the second phase, the robot camera (E) can be moved by the arm with respect to this reference system (P), finally providing a second hand-eye transformation (HE). This second phase gives the advantage of starting the procedure from a known point and moving the arm along a more controlled and repeatable path.

C.2.1 Pattern Localization

As previously mentioned, this automatic procedure starts from a position from which the calibration pattern falls in the camera Field of View (FoV). An example is reported in Figure C.2.2. From here, the robotic arm moves through several



(a) The robotic arm UR10 with a stereo-camera rigidly mounted on the gripper (EuRoC project setup).



(b) Checkerboard view: it is in the camera FoV.

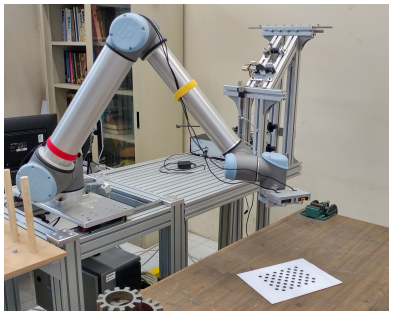
Figure C.2.2: An example of starting position of the pattern localization.

stations with a stop-and-go motion so as to avoid issues when synchronizing the robot-pose with the image. These stations are not chosen a priori. The robotic arm moves its gripper along each axis until the pattern stops being detected. In particular, at fixed steps, a check is performed to verify that the pattern is still detected, hence visible. If so, the acquired image can be used to calibrate and locate the checkerboard, and the robot can keep moving along that axis. Additionally, more stations can be added by rotating the gripper around the gripper axis. Collision avoidance and target reachability is guaranteed by the MoveIt! library.

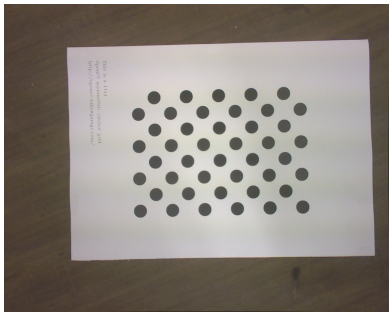
At the end of this phase, the pattern localization is performed from a set of rectified images $\{I_i\}$ framing the calibration pattern and a set of Base to Hand transformations $\{BH_i\}$ with $i \in \{0, \dots, N-1\}$. For each image, the transformation Eye to Pattern EP can be estimated from the 3D-2D point correspondences, e.g. by means of the iterative method based on the Levenberg-Marquardt optimization available in the OpenCV library. From $\{BH_i\}$ and $\{EP_i\}$, the hand-eye calibration method by Tsai et al. [135] leads to the estimation of HE. Thus, the pattern calibration location can be easily estimated as $BP = BH_{N-1} \times HE \times EP_{N-1}$, where BH_{N-1} is a valid robot pose and EP_{N-1} the respective valid Eye to Pattern transformation.

C.2.2 Automatic Acquisition

Once a first hand-eye HE is estimated and the checkerboard P is located in the work-cell, the camera can be moved with respect to the checkerboard independently from its location in the workspace along a controlled and repeatable path. As shown in Figure C.2.3, the acquisition can always start from the same view of the checkerboard.



(a) The robotic arm UR10 with a stereo-camera rigidly mounted on the gripper (EuRoC project setup).



(b) Checkerboard view: it is in the camera FoV and clearly visible.

Figure C.2.3: An example of starting position of the automatic acquisition.

In this phase, the robot moves mimicking a typical acquisition performed by a human and, as suggested by Tsai et al. in [133], varying distances and angles are considered. The robot moves again in a stop-and-go fashion stopping with a fixed stride to check that the calibration pattern keeps being visible. A scheme representing the camera movements is reported in Figure C.2.4. In particular:

- the camera E scans a number of planes *num_p*, which are parallel to the checkerboard but outdistanced of a fixed step *d_p* starting from a minimum distance *d_min_p*. For instance, *num_p* can be equal to 3, *d_p* to 0.05 m and *d_min_p* to 0.40 m;
- the plane scan is performed from a position 0, almost in the center. Then, the camera is moved along the checkerboard axis x and y, between the position 1 and 2 or 3 and 4. The terminal positions are not fixed. Indeed, as in the

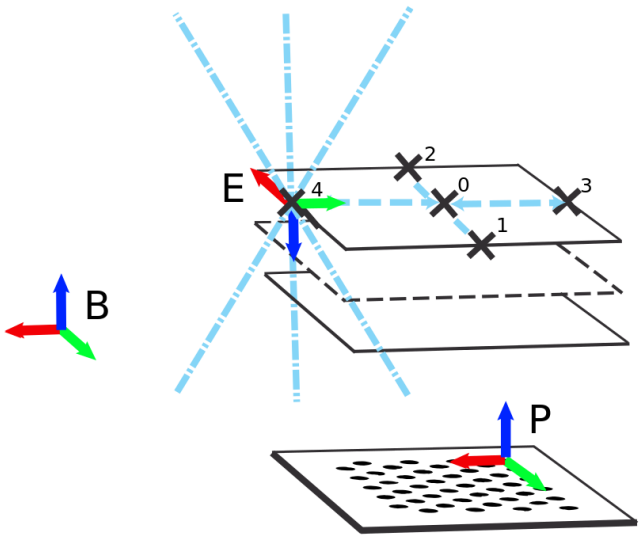


Figure C.2.4: The camera can be moved with respect to the checkerboard independently from its location in the workspace along a controlled and repeatable path.

previous phase, the camera is moved along an axis until the checkerboard stops being in the FoV;

- each plane can be scanned with varying configurable angles. This is shown in Figure C.2.4, in which the camera E at the position 4 is rotated with three different angles around the axis y. Rotations around other axes can be set too. For instance, one plane can be scanned several times, first with all the angles set to 0, then setting a rotation angle of 10° around the axis x, then 10° around the axis y and so on.

Again, collision avoidance and target reachability is guaranteed by the MoveIt! library. If any of the stations cannot be reached, the robot continues moving towards the subsequent one. From the new sets $\{BH_i\}$ and $\{EP_i\}$, the hand-eye calibration method by Tsai et al. [135] leads to the estimation of HE.

C.3 System Adaptability

This approach has been implemented by means of widely adopted libraries/frameworks ViSP, ROS, MoveIt!, OpenCV and PCL. Currently, the MoveIt! library supports 65 robots. Thus, even if this implementation has been tested on the Universal Robot UR10 only, it can be potentially used with many others. We selected an approach based on well-known libraries and frameworks for being able to work with different robots and cameras with almost no need for code and system modifications. Indeed, our approach obtains the hand-eye calibration starting solely by camera and robot. Standard procedures integrated in the ROS framework provide robot and camera information to our algorithm. A configuration file stores parameters related to different characteristics of the system.

Camera intrinsic parameters are collected by using a *topic*[6]. In the majority of the sensors available in ROS, the *topic* is published directly by the camera driver, and it can be updated by using a camera calibration procedure already available in ROS. Anyway, if no information about intrinsic parameters is exposed by the camera driver, the message firm is publicly available on the ROS website.

Similarly, the system looks for a robot model to learn its physical structure, know about kinematic chains, and understand dynamics. The physical structure is represented by means of the Unified Robot Description Format (URDF)[7]. In the URDF, the robot is composed of joints and links: part dimensions, motion limits, weights, visual and collision shapes complete the robot description. Kinematic chains are defined inside the Semantic Robot Description Format (SRDF)[8]. Moreover, SRDF is usually accompanied by a series of files providing information about collision avoidance, inverse kinematics, robot controllers. MoveIT! is able to exploit such information simply by referring to a specific kinematic group. In our system, it is possible to select the correct kinematic group by changing a parameter in the configuration file. Dynamics is fundamental for a correct simulation in the Gazebo environment. This information can be integrated directly in the URDF file and it is the basis for obtaining realistic tests with physics engines.

[6] http://wiki.ros.org/Topics
[7] http://wiki.ros.org/urdf
[8] http://wiki.ros.org/srdf

C.4 Experimental Results

To test the automatic procedure, a scenario similar to the one of the EuRoC project has been selected. The robotic arm is the Universal Robot UR10, collaborative and with 6 degrees of freedom. A stereo-camera composed of two PC webcams is rigidly mounted on it. One of them is taken as a reference, and the hand-eye calibrations are computed with respect to it. In the following, the results of experiments performed in both simulated and real test-beds are discussed.

C.4.1 Simulation Experiments

Thanks to the simulation environment, the ground-truth hand-eye transformation and checkerboard can be easily retrieved. To assess the generality of the system, it was tested with the robot starting from 3 different positions (Test A) and with 3 different checkerboard positions in the work-cell (Test B). The 6 different configurations are reported in Figure C.4.1 and C.4.2, respectively. In total, 12 hand-eye transformations and 6 checkerboard locations have been estimated.

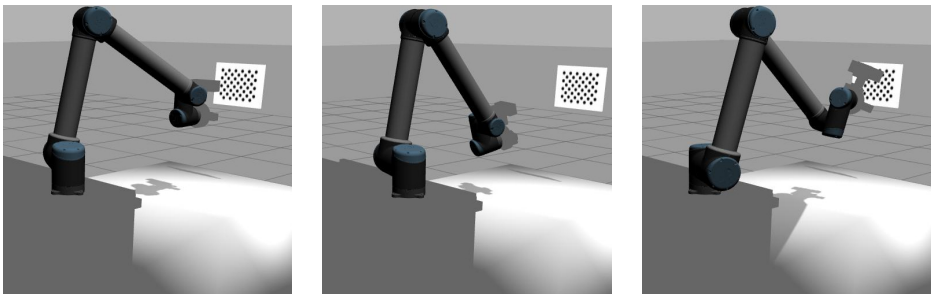
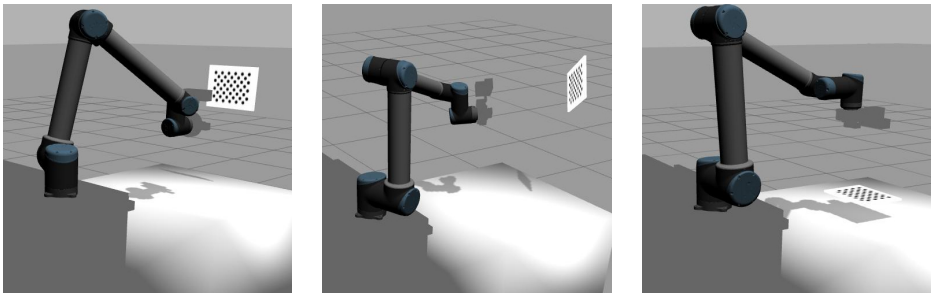


Figure C.4.1: Different starting positions with varying inclinations of the camera. The calibration pattern stays on a vertical plane.

With regard to Test A, the main results are summed up in Table C.4.1. The hand-eye transformations have been compared in both translation and angle, in particular Δt is the Euclidean distance in meters while Δq the difference vector between two quaternions in degrees. In addition to the ground truth, the HE transformations have been compared with a baseline consisting in the HE calculated with a manually determined motion defined by 10 waypoints at varying heights



(a) **(b)** **(c)**

Figure C.4.2: Different checkerboard positions: on a vertical, horizontal and inclined plane.

and angles. The baseline, the hand-eye calculated in the first phase and the hand-eye calculated in the second phase are almost equivalent with a standard deviation of only 0.0017 m and 0.1016°. Interestingly, in this ideal simulated scenario, the second hand-eye calibration leads to worse performance even if in the second phase the camera is closer to the calibration pattern. This implies that the image quality is higher but also that the number of acquired images is minor since the camera is staying closer to the checkerboard and moving with a fixed step from one position to the next until the checkerboard is visible. In this ideal scenario, the second effect is clearly prominent. Finally, as shown in Table C.4.2, the differences in the checkerboard localization are also minimal. The worst entry deviates from the ground-truth of about 0.0159 m (Position 2) and 1.1412° (Position 1). As expected, these deviations are higher since the error on the hand-eye is accumulated with the errors in the robot positioning (BH_{N-1} transformation) and, mainly, the error in the checkerboard positioning with respect to the camera (EP_{N-1} transformation).

With regard to Test B, the main results are summed up in Table C.4.3. Even if the robot starts from 3 different positions, both phases of the calibration procedure succeed giving results similar to those already presented.

C.4.2 Real Experiments

The entire procedure has been executed from 5 different positions, which have been randomly chosen around the checkerboard at a distance of about 1.00 m. In

Table C.4.1: Hand-eye transformations in the simulation experiment Test A: translations and rotations in quaternions. The robot starts from three different positions. The checkerboard is vertical.

HE	x	y	z	qx	qy	qz	qw	Δt	Δq
Ground truth	0.0842	0.0959	0.1139	0.5879	-0.3928	0.5879	-0.3928	0.0000	0.0000
Baseline	0.0887	0.0936	0.1087	0.586417	-0.390973	0.589555	-0.39445	0.0072	1.5200
Phase 1 - Position 1	0.0879	0.0914	0.1133	0.5880	-0.3928	0.5878	-0.3929	0.0058	1.4806
Phase 1 - Position 2	0.0866	0.0940	0.1105	0.5888	-0.3929	0.5886	-0.3904	0.0045	1.5720
Phase 1 - Position 3	0.0884	0.0916	0.1119	0.5877	-0.3943	0.5868	-0.3935	0.0063	1.2554
Phase 2 - Position 1	0.0927	0.0945	0.1128	0.5885	-0.3931	0.5875	-0.3923	0.0086	1.4433
Phase 2 - Position 2	0.0924	0.0942	0.1130	0.5886	-0.3930	0.5876	-0.3922	0.0084	1.4893
Phase 2 - Position 3	0.0931	0.0942	0.1126	0.5885	-0.3931	0.5875	-0.3923	0.0091	1.5119

Table C.4.2: Base-checkerboard transformations in the simulation experiment Test A: translations and rotations in quaternions. The robot starts from three different positions. The checkerboard is vertical.

BP	x	y	z	qx	qy	qz	qw	Δt	Δq
Ground truth	0.8000	1.5000	1.4000	-0.7071	0.0000	0.0000	0.7071	0.0000	0.0000
Position 1	0.7997	1.4965	1.3977	-0.7066	0.0002	-0.0002	0.7075	0.0042	1.1412
Position 2	0.7864	1.4928	1.3957	-0.7082	-0.0011	-0.0015	0.7059	0.0159	1.1248
Position 3	0.8012	1.4929	1.3964	-0.7063	0.0011	0.0016	0.7078	0.0080	1.1411

Table C.4.3: Hand-eye transformations in the simulation experiment Test B: translations and rotations in quaternions. The checkerboard is positioned in 3 different places: vertical, horizontal and on an inclined plane.

HE	x	y	z	qx	qy	qz	qw	Δt	Δq
Ground truth	0.0842	0.0959	0.1139	0.5879	-0.3928	0.5879	-0.3928	0.0000	0.0000
checkerboard 1	0.0927	0.0945	0.1128	0.5885	-0.3931	0.5875	-0.3923	0.0086	1.4433
checkerboard 2	0.0928	0.0943	0.1129	0.5885	-0.3930	0.5875	-0.3922	0.0088	0.9978
checkerboard 3	0.0914	0.0943	0.1130	0.5885	-0.3932	0.5875	-0.3924	0.0074	1.3638

the second phase, the robot has scanned 3 planes at the distances of 0.40 m, 0.45 m and 0.50 m, from which the checkerboard is better focused. One of these trials has been already shown in Figure C.2.2 and C.2.3. In total, 10 hand-eye transformations, 5 for each of the two phases, and 5 checkerboards have been estimated. In Table C.4.4, the main results are recapped. Given the lack of the ground-truth,

Table C.4.4: Hand-eye transformations in the real experiments and their standard deviations in the two phases.

HE	x	y	z	qx	qy	qz	qw	std(t)	std(q)
Phase 1 - Position 1	-0.2394	-0.0280	-0.0961	0.5141	0.4865	0.5205	0.4776		
Phase 1 - Position 2	-0.2527	-0.0341	-0.0880	0.4856	0.5081	0.5162	0.4894		
Phase 1 - Position 3	-0.2394	-0.0237	-0.1095	0.5067	0.4917	0.5152	0.4859	0.0113	1.8703
Phase 1 - Position 4	-0.2452	-0.0293	-0.0934	0.4993	0.5001	0.5183	0.4816		
Phase 1 - Position 5	-0.2372	-0.0368	-0.0965	0.5004	-0.4933	-0.5223	-0.4831		
Phase 2 - Position 1	-0.2521	-0.0215	-0.1075	0.4974	0.5055	0.4783	0.5180		
Phase 2 - Position 2	-0.2500	-0.0191	-0.1091	0.4973	0.5031	0.4775	0.5211		
Phase 2 - Position 3	-0.2450	-0.0289	-0.1117	0.4950	0.5093	0.4807	0.5143	0.0054	0.5324
Phase 2 - Position 4	-0.2517	-0.0181	-0.1096	0.4960	0.5059	0.4745	0.5225		
Phase 2 - Position 5	-0.2505	-0.0194	-0.1096	0.4974	0.5054	0.4763	0.5199		

no direct comparison can be performed. Anyway, it can be seen that, as expected, the standard deviations of the hand-eye translations and rotations obtained after the first phase (respectively 0.0113 m and 1.8703°) are much higher than the standard deviations in the second phase (respectively 0.0054 m and 0.5324°). This is due to the fact that the second phase moves the robot along a known path in the checkerboard reference system.

C.5 Conclusions

This appendix presented an automatic system for localizing the calibration pattern and performing the hand-eye calibration taking a step forward in comparison to existing methods. Indeed, previous works focused on optimizing the hand-eye calibration instead of trying to fully automatize the procedure from the beginning to the end. This tool has a wide range of applicability, also in this thesis we present two robots which can benefit from this tool: the inspection robot in Chapter 4 and the mobile robot for navigation, perception and manipulation in Appendix B. The proposed approach is based on two phases, in each of which an hand-eye calibration is performed. The second phase allows acquiring images in a controlled setup by moving the camera with respect to the calibration pattern. Both phases were tested in simulated and real scenarios showing that stable results can be obtained after the second phase. As a final contribution to the research community, the toolbox is released online. We plan to use this tool in the upcoming projects and keep on improving it. We would like to test and compare different tool and cam-

era paths in the two phases in order to investigate on the relation between path and overall accuracy.

References

[1] J. Prankl, A. Aldoma, A. Svejda, and M. Vincze, "Rgb-d object modelling for object recognition and tracking," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 96–103, IEEE, 2015.

[2] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[3] P. Sturgess, K. Alahari, L. Ladicky, and P. H. Torr, "Combining appearance and structure from motion features for road scene understanding," in *BMVC 2012-23rd British Machine Vision Conference*, BMVA, 2009.

[4] L.-J. Li, R. Socher, and L. Fei-Fei, "Towards total scene understanding: Classification, annotation and segmentation in an automatic framework," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 2036–2043, IEEE, 2009.

[5] L. Fei-Fei, C. Koch, A. Iyer, and P. Perona, "What do we see when we glance at a scene?," *Journal of Vision*, vol. 4, no. 8, pp. 863–863, 2004.

[6] L. Fei-Fei, A. Iyer, C. Koch, and P. Perona, "What do we perceive in a glance of a real-world scene?," *Journal of vision*, vol. 7, no. 1, pp. 10–10, 2007.

[7] J. M. Wolfe, "Visual memory: What do you know about what you saw?," *Current biology*, vol. 8, no. 9, pp. R303–R304, 1998.

[8] M. Antonello, D. Wolf, J. Prankl, S. Ghidoni, E. Menegatti, and M. Vincze, "Multi-view 3d entangled forest for semantic segmentation and mapping," in *Robotics and Automation (ICRA), 2018 IEEE International Conference on*, IEEE, 2018.

[9] M. Antonello, M. Carraro, M. Pierobon, and E. Menegatti, "Fast and robust detection of fallen people from a mobile robot," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4159–4166, Sept 2017.

[10] M. Antonello, M. Munaro, and E. Menegatti, "Efficient measurement of fibre orientation for mapping carbon fibre parts with a robotic system," in *IAS-14, Shanghai, China*, pp. 757–769, 2016.

[11] M. Munaro, M. Antonello, M. Moro, C. Ferrari, E. Pagello, and E. Menegatti, "Fibremap: Automatic mapping of fibre orientation for draping of carbon fibre parts," in *IAS-13 Workshop on ROS-Industrial in European Research Projects, Padova, Italy*, pp. 272–275, July 2014.

[12] M. Antonello, S. Ghidoni, and E. Menegatti, "Autonomous robotic system for thermographic detection of defects in upper layers of carbon fiber reinforced polymers," in *Automation Science and Engineering (CASE), 2015 IEEE International Conference on*, pp. 634–639, IEEE, 2015.

[13] M. Vincze, M. Bajones, M. Suchi, D. Wolf, A. Weiss, D. Fischinger, and P. da la Puente, "Learning and detecting objects with a mobile robot to assist older adults in their homes," in *European Conference on Computer Vision*, pp. 316–330, Springer, 2016.

[14] M. Bajones, D. Wolf, J. Prankl, and M. Vincze, "Where to look first? behaviour control for fetch-and-carry missions of service robots," *arXiv preprint arXiv:1510.01554*, 2015.

[15] D. Wolf, M. Bajones, J. Prankl, and M. Vincze, "Find my mug: Efficient object search with a mobile robot using semantic segmentation," *arXiv preprint arXiv:1404.5765*, 2014.

[16] X. Zhang and Y. LeCun, "Text understanding from scratch," *arXiv preprint arXiv:1502.01710*, 2015.

[17] Q. Liu, R. Li, H. Hu, and D. Gu, "Building semantic maps for blind people to navigate at home," in *Computer Science and Electronic Engineering (CEEC), 2016 8th*, pp. 12–17, IEEE, 2016.

[18] B. Siciliano, F. Caccavale, E. Zwicker, M. Achtelik, N. Mansard, C. Borst, M. Achtelik, N. O. Jepsen, R. Awad, and R. Bischoff, "Euroc-the challenge

initiative for european robotics," in *ISR/Robotik 2014; 41st International Symposium on Robotics; Proceedings of*, pp. 1–7, VDE, 2014.

[19] F. Maninchedda, C. Häne, B. Jacquet, A. Delaunoy, and M. Pollefeys, "Semantic 3d reconstruction of heads," in *European Conference on Computer Vision*, pp. 667–683, Springer, 2016.

[20] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning*, pp. 2048–2057, 2015.

[21] K. Vanhoey, C. E. P. de Oliveira, H. Riemenschneider, A. Bódis-Szomorú, S. Manén, D. P. Paudel, M. Gygli, N. Kobyshev, T. Kroeger, D. Dai, *et al.*, "Varcity-the video: the struggles and triumphs of leveraging fundamental research results in a graphics video production," in *ACM SIGGRAPH 2017 Talks*, p. 48, ACM, 2017.

[22] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, p. 5, 2009.

[23] G. Bradski, "The opencv library," *Doctor Dobbs Journal*, vol. 25, no. 11, pp. 120–126, 2000.

[24] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1–4, IEEE, 2011.

[25] G. Gaël, J. Benoît, *et al.*, "Eigen v3." http://eigen.tuxfamily.org, 2010.

[26] F. S. Cohen, Z. Fan, and S. Attali, "Automated inspection of textile fabrics using textural models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 8, pp. 803–808, 1991.

[27] S. Odemir, A. Baykut, R. Meylani, A. Erçil, and A. Ertuzun, "Comparative evaluation of texture analysis algorithms for defect inspection of textile products," in *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, vol. 2, pp. 1738–1740, IEEE, 1998.

[28] P. Bonsma, I. Bonsma, A. E. Ziri, S. Parenti, P. M. Lerones, J. L. Hernández, F. Maietti, M. Medici, B. Turillazzi, and E. Iadanza, "Inception standard for heritage bim models," in *Euro-Mediterranean Conference*, pp. 590–599, Springer, 2016.

[29] A. Martinovic, J. Knopp, H. Riemenschneider, and L. Van Gool, "3d all the way: Semantic segmentation of urban scenes from start to end in 3d," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4456–4465, 2015.

[30] N. Silberman and R. Fergus, "Indoor scene segmentation using a structured light sensor," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pp. 601–608, IEEE, 2011.

[31] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *European Conference on Computer Vision*, pp. 746–760, Springer, 2012.

[32] C. Couprie, C. Farabet, L. Najman, and Y. Lecun, "Indoor semantic segmentation using depth information," in *First International Conference on Learning Representations (ICLR 2013)*, pp. 1–8, 2013.

[33] A. Hermans, G. Floros, and B. Leibe, "Dense 3d semantic mapping of indoor scenes from rgb-d images," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 2631–2638, IEEE, 2014.

[34] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "Semanticfusion: Dense 3d semantic mapping with convolutional neural networks," *arXiv preprint arXiv:1609.05130*, 2016.

[35] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1352–1359, 2013.

[36] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik, "Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation," *International Journal of Computer Vision*, vol. 112, no. 2, pp. 133–149, 2015.

[37] V. Vineet, O. Miksik, M. Lidegaard, M. Nießner, S. Golodetz, V. A. Prisacariu, O. Kähler, D. W. Murray, S. Izadi, P. Pérez, *et al.*, "Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 75–82, IEEE, 2015.

[38] D. Kochanov, A. Ošep, J. Stückler, and B. Leibe, "Scene flow propagation for semantic mapping and object discovery in dynamic street scenes," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pp. 1785–1792, IEEE, 2016.

[39] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1915–1929, 2013.

[40] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, "Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture.," in *ACCV (1)*, pp. 213–228, 2016.

[41] D. Eigen and R. 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*, pp. 2650–2658, 2015.

[42] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla, "Understanding real world indoor scenes with synthetic data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4077–4085, 2016.

[43] D. Wolf, J. Prankl, and M. Vincze, "Fast semantic segmentation of 3d point clouds using a dense crf with learned parameters," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 4867–4873, IEEE, 2015.

[44] D. Wolf, J. Prankl, and M. Vincze, "Enhancing semantic segmentation for robotics: the power of 3-d entangled forests," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 49–56, 2016.

[45] A. Anand, H. S. Koppula, T. Joachims, and A. Saxena, "Contextually guided semantic labeling and search for three-dimensional point clouds,"

The International Journal of Robotics Research, vol. 32, no. 1, pp. 19–34, 2013.

[46] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-d mapping with an rgb-d camera," *IEEE Transactions on Robotics*, vol. 30, no. 1, pp. 177–187, 2014.

[47] J. Stückler, N. Biresev, and S. Behnke, "Semantic mapping using object-class segmentation of rgb-d images," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 3005–3010, IEEE, 2012.

[48] Z. Zhao and X. Chen, "Semantic mapping for object category and structural class," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 724–729, IEEE, 2014.

[49] Z. Zhao and X. Chen, "Building 3d semantic maps for mobile robots using rgb-d camera," *Intelligent Service Robotics*, vol. 9, no. 4, pp. 297–309, 2016.

[50] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *arXiv preprint arXiv:1612.00593*, 2016.

[51] L. P. Tchapmi, C. B. Choy, I. Armeni, J. Gwak, and S. Savarese, "Segcloud: Semantic segmentation of 3d point clouds," *arXiv preprint arXiv:1710.07563*, 2017.

[52] K. Tateno, F. Tombari, and N. Navab, "When 2.5 d is not enough: Simultaneous reconstruction, segmentation and recognition on dense slam," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pp. 2295–2302, IEEE, 2016.

[53] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 309–314, 2004.

[54] V. Lempitsky, P. Kohli, C. Rother, and T. Sharp, "Image segmentation with a bounding box prior," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 277–284, IEEE, 2009.

[55] J. Hosang, R. Benenson, and B. Schiele, "How good are detection proposals, really?," *arXiv preprint arXiv:1406.6962*, 2014.

[56] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 4, pp. 814–830, 2016.

[57] C. Papageorgiou and T. Poggio, "A trainable system for object detection," *International Journal of Computer Vision*, vol. 38, no. 1, pp. 15–33, 2000.

[58] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[59] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[60] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Computer Vision and Pattern Recognition*, 2014.

[61] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[62] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[63] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[64] A. Kanezaki and T. Harada, "3d selective search for obtaining object candidates," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 82–87, IEEE, 2015.

[65] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, 2016.

[66] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," *arXiv preprint arXiv:1612.08242*, 2016.

[67] S. Wang, S. Zabir, and B. Leibe, "Lying pose recognition for elderly fall detection," *Robotics: Science and Systems VII*, vol. 345, 2012.

[68] M. Volkhardt, F. Schneemann, and H.-M. Gross, "Fallen person detection for mobile robots using 3d depth data," in *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, pp. 3573–3578, IEEE, 2013.

[69] K. Nishi and J. Miura, "A head position estimation method for a variety of recumbent positions for a care robot," in *Proceedings of the 6th Int. Conf. on Advanced Mechatronics*, 2015.

[70] J. Perry, S. Kellog, S. Vaidya, J.-H. Youn, H. Ali, and H. Sharif, "Survey and evaluation of real-time fall detection approaches," in *High-Capacity Optical Networks and Enabling Technologies (HONET), 2009 6th International Symposium on*, pp. 158–164, IEEE, 2009.

[71] Q. Li, J. Stankovic, M. Hanson, A. Barth, J. Lach, and G. Zhou, "Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information," in *Wearable and Implantable Body Sensor Networks, 2009. BSN 2009. Sixth International Workshop on*, pp. 138–143, IEEE, 2009.

[72] J. Boyle and M. Karunanithi, "Simulated fall detection via accelerometers," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pp. 1274–1277, IEEE, 2008.

[73] U. Lindemann, A. Hock, M. Stuber, W. Keck, and C. Becker, "Evaluation of a fall detector based on accelerometers: A pilot study," *Medical and Biological Engineering and Computing*, vol. 43, no. 5, pp. 548–551, 2005.

[74] M. Popescu, Y. Li, M. Skubic, and M. Rantz, "An acoustic fall detector system that uses sound height information to reduce the false alarm rate," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pp. 4628–4631, IEEE, 2008.

[75] A. Yazar, F. Erden, and A. E. Cetin, "Multi-sensor ambient assisted living system for fall detection," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 14)*, pp. 1–3, Citeseer, 2014.

[76] A. Williams, D. Ganesan, and A. Hanson, "Aging in place: fall detection and localization in a distributed smart camera network," in *Proceedings of the 15th ACM international conference on Multimedia*, pp. 892–901, ACM, 2007.

[77] R. Cucchiara, A. Prati, and R. Vezzani, "A multi-camera vision system for fall detection and alarm generation," *Expert Systems*, vol. 24, no. 5, pp. 334–345, 2007.

[78] S. Ghidoni, S. M. Anzalone, M. Munaro, S. Michieletto, and E. Menegatti, "A distributed perception infrastructure for robot assisted living," *Robotics and Autonomous Systems*, vol. 62, no. 9, pp. 1316–1328, 2014.

[79] S. Ozdemir, A. Baykut, R. Meylani, A. Ercil, and A. Ertuzun, "Comparative evaluation of texture analysis algorithms for defect inspection of textile products," in *Pattern Recognition, International Conference on*, vol. 2, pp. 1738–1738, IEEE Computer Society, 1998.

[80] R. Schmitt, C. Mersmann, and A. Schoenberg, "Machine vision industrialising the textile-based frp production," in *6th Int. Symposium on Image and Signal Processing and Analysis (ISPA)*, pp. 260–264, IEEE, 2009.

[81] L. Shi and S. Wu, "Automatic fiber orientation detection for sewed carbon fibers," *Tsinghua Science & Technology*, vol. 12, no. 4, pp. 447–452, 2007.

[82] S. Zambal, W. Palfinger, M. Stöger, and C. Eitzinger, "Accurate fibre orientation measurement for carbon fibre surfaces," *Pattern Recognition*, vol. 48, no. 11, pp. 3324–3332, 2015.

[83] M. Kass and A. Witkin, "Analyzing oriented patterns," *Computer vision, graphics, and image processing*, vol. 37, no. 3, pp. 362–385, 1987.

[84] B. B. Chaudhuri, P. Kundu, and N. Sarkar, "Detection and gradation of oriented texture," *Pattern recognition letters*, vol. 14, no. 2, pp. 147–153, 1993.

[85] A. Miene, A. Herrmann, and M. Göttinger, "Quality assurance by digital image analysis for the preforming and draping process of dry carbon fiber material," in *SAMPE Europe Conference, Paris*, pp. 348–353, 2008.

[86] J. Chang and J. Fisher, "Analysis of orientation and scale in smoothly varying textures," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 881–888, IEEE, 2009.

[87] H. Fernandes and X. Maldague, "Fiber orientation assessment in complex shaped parts reinforced with carbon fiber using infrared thermography," *Quantitative InfraRed Thermography Journal*, vol. 12, no. 1, pp. 64–79, 2015.

[88] E. Weigl, S. Zambal, M. Stöger, and C. Eitzinger, "Photometric stereo sensor for robot-assisted industrial quality inspection of coated composite material surfaces," in *Twelfth International Conference on Quality Control by Artificial Vision*, vol. 9534, pp. 95341D–95341D–8, Proc. SPIE, 2015.

[89] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke, "Registration with the point cloud library: A modular framework for aligning in 3-d," *IEEE Robotics & Automation Magazine*, vol. 22, no. 4, pp. 110–124, 2015.

[90] P. J. Besl, N. D. McKay, *et al.*, "A method for registration of 3-d shapes," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[91] W. Cheung and G. Hamarneh, "*n*-sift: *n*-dimensional scale invariant feature transform," *IEEE Transactions on Image Processing*, vol. 18, no. 9, pp. 2012–2021, 2009.

[92] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[93] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[94] C. Ibarra-Castanedo, J. Piau, S. Guilbert, N. Avdelidis, M. Genest, A. Bendada, and X. Maldague, "Comparative study of active thermography techniques for the nondestructive evaluation of honeycomb structures," *Research in Nondestructive Evaluation*, vol. 20, no. 1, pp. 1–31, 2009.

[95] M. Ishikawa, H. Hatta, Y. Habuka, R. Fukui, and S. Utsunomiya, "Detecting deeper defects using pulse phase thermography," *Infrared Physics & Technology*, vol. 57, pp. 42–49, 2013.

[96] D. Almond, B. Weekes, T. Li, S. Pickering, E. Kostson, J. Wilson, G. Tian, S. Dixon, and S. Burrows, "Thermographic techniques for the detection of cracks in metallic components," *Insight-Non-Destructive Testing and Condition Monitoring*, vol. 53, no. 11, pp. 614–620, 2011.

[97] H. Berglind and A. Dillenz, "Detecting glue deficiency in laminated wood - a thermography method comparison," *NDT & E International*, vol. 36, no. 6, pp. 395–399, 2003.

[98] W. Swiderski and V. Vavilov, "Ir thermographic detection of defects in multi-layered composite materials used in military applications," in *Infrared and Millimeter Waves, 2007 and the 2007 15th International Conference on Terahertz Electronics. IRMMW-THz. Joint 32nd International Conference on*, pp. 553–556, IEEE, 2007.

[99] L. Boquete, S. Ortega, J. Miguel-Jiménez, J. Rodríguez-Ascariz, and R. Blanco, "Automated detection of breast cancer in thermal infrared images, based on independent component analysis," *Journal of medical systems*, vol. 36, no. 1, pp. 103–111, 2012.

[100] J. Head and R. Elliott, "Infrared imaging: making progress in fulfilling its medical promise," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 21, no. 6, pp. 80–85, 2002.

[101] S. Goferman, L. Zelnik-Manor, and A. Tal, "Context-aware saliency detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 10, pp. 1915–1926, 2012.

[102] S. Ghidoni, M. Antonello, L. Nanni, and E. Menegatti, "A knowledge-based approach to crack detection in thermographic images," in *13th Int. Conf. on Intelligent Autonomous Systems (IAS-13)*, 2014. In press.

[103] M. Carraro, M. Antonello, L. Tonin, and E. Menegatti, "An open source robotic platform for ambient assisted living," *Artificial Intelligence and Robotics (AIRO)*, 2015.

[104] N. Castaman, E. Tosello, M. Antonello, N. Bagarello, S. Gandin, M. Carraro, M. Munaro, R. Bortoletto, S. Ghidoni, E. Menegatti, *et al.*, "Rur53: an unmanned ground vehicle for navigation, recognition and manipulation," *arXiv preprint arXiv:1711.08764*, 2017.

[105] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3d semantic parsing of large-scale indoor spaces," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2016.

[106] J. Papon, A. Abramov, M. Schoeler, and F. Worgotter, "Voxel cloud connectivity segmentation-supervoxels for point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2027–2034, 2013.

[107] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[108] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[109] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*, pp. 740–755, Springer, 2014.

[110] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, IEEE, 2009.

[111] D. Thanh Nguyen, B.-S. Hua, L.-F. Yu, and S.-K. Yeung, "A robust 3d-2d interactive tool for scene segmentation and annotation," *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 2017.

[112] Q.-Y. Zhou, S. Miller, and V. Koltun, "Elastic fragments for dense scene reconstruction," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 473–480, 2013.

[113] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison, "Elasticfusion: Dense slam without a pose graph," Robotics: Science and Systems, 2015.

[114] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, "Meshlab: an open-source mesh processing tool," in *Eurographics Italian Chapter Conference* (V. Scarano, R. D. Chiara, and U. Erra, eds.), The Eurographics Association, 2008.

[115] M. Corsini, P. Cignoni, and R. Scopigno, "Efficient and flexible sampling with blue noise properties of triangular meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 6, pp. 914–924, 2012.

[116] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," *IEEE transactions on visualization and computer graphics*, vol. 5, no. 4, pp. 349–359, 1999.

[117] X. Ren, L. Bo, and D. Fox, "Rgb-(d) scene labeling: Features and algorithms," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 2759–2766, IEEE, 2012.

[118] I. Armeni, A. Sax, A. R. Zamir, and S. Savarese, "Joint 2D-3D-Semantic Data for Indoor Scene Understanding," *ArXiv e-prints*, Feb. 2017.

[119] M. E. Pollack, L. Brown, D. Colbry, C. Orosz, B. Peintner, S. Ramakrishnan, S. Engberg, J. T. Matthews, J. Dunbar-Jacob, C. E. McCarthy, *et al.*, "Pearl: A mobile robotic assistant for the elderly," in *AAAI workshop on automation as eldercare*, vol. 2002, pp. 85–91, 2002.

[120] F. Cavallo, M. Aquilano, M. Bonaccorsi, R. Limosani, A. Manzi, M. Carrozza, and P. Dario, "On the design, development and experimentation of the astro assistive robot integrated in smart environments," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 4310–4315, IEEE, 2013.

[121] H.-M. Gross, S. Mueller, C. Schroeter, M. Volkhardt, A. Scheidig, K. Debes, K. Richter, and N. Doering, "Robot companion for domestic health assistance: Implementation, test and case study under everyday conditions in private apartments," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 5992–5999, IEEE, 2015.

[122] D. Fischinger, P. Einramhof, K. Papoutsakis, W. Wohlkinger, P. Mayer, P. Panek, S. Hofmann, T. Koertner, A. Weiss, A. Argyros, *et al.*, "Hobbit, a care robot supporting independent living at home: First prototype and lessons learned," *Robotics and Autonomous Systems*, vol. 75, pp. 60–78, 2016.

[123] W. He, D. Goodkind, and P. Kowal, "An aging world: 2015," tech. rep., 2016.

[124] S. Lord, C. Sherrington, H. Menz, and J. Close, *Falls in older people: risk factors and strategies for prevention*. Cambridge University Press, 2007.

[125] S. C. Stein, F. Wörgötter, M. Schoeler, J. Papon, and T. Kulvicius, "Convexity based object partitioning for robot applications," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 3213–3220, IEEE, 2014.

[126] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 2432–2437, IEEE, 2005.

[127] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[128] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," *AAAI/IAAI*, vol. 1999, pp. 343–349, 1999.

[129] K. Ikeuchi, "Determining surface orientations of specular surfaces by using the photometric stereo method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 6, pp. 661–669, 1981.

[130] A. Baumberg, "Blending images for texturing 3d models.," in *BMVC*, vol. 3, p. 5, 2002.

[131] J. Kim and D. Lee, "Effect of fiber orientation and fiber contents on the tensile strength in fiber-reinforced composites," *Journal of nanoscience and nanotechnology*, vol. 10, no. 5, pp. 3650–3653, 2010.

[132] J. Fuhr, J. Baumann, F. Härtel, P. Middendorf, and N. Feindler, "Effects of in-plane waviness on the properties of carbon composites–experimental and numerical analysis," *CompTest 2013-Book of Abstracts*, p. 61, 2013.

[133] R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand/eye calibration," *Robotics and Automation, IEEE Transactions on*, vol. 5, no. 3, pp. 345–358, 1989.

[134] N. Carlon, N. Boscolo, S. Tonello, and E. Menegatti, "Smart check 3d: An industrial inspection system combining 3d vision with automatic planning of inspection viewpoints," in *Integrated Imaging and Vision Techniques for Industrial Inspection*, pp. 377–392, Springer, 2015.

[135] R. Y. Tsai and R. K. Lenz, "Real time versatile robotics hand/eye calibration using 3d machine vision," in *Robotics and Automation, 1988 IEEE International Conference on*, pp. 554–561, IEEE, 1988.

[136] M. Antonello, A. Gobbi, S. Michieletto, S. Ghidoni, and E. Menegatti, "A fully automatic hand-eye calibration system," in *European Conference on Mobile Robotics (ECMR)*, 2017.

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

[138] J. F. Hughes, A. Van Dam, J. D. Foley, and S. K. Feiner, *Computer graphics: principles and practice*. Pearson Education, 2013.

[139] T. Möller and B. Trumbore, "Fast, minimum storage ray-triangle intersection," *J. Graph. Tools*, vol. 2, pp. 21–28, Oct. 1997.

[140] S. Kumar, D. Manocha, B. Garrett, and M. Lin, "Hierarchical back-face culling," in *7th Eurographics Workshop on Rendering*, pp. 231–240, 1996.

[141] D. Meagher, "Geometric modeling using octree encoding," *Computer graphics and image processing*, vol. 19, no. 2, pp. 129–147, 1982.

[142] A. Kelly, *Concise encyclopedia of composite materials*. Elsevier, 2012.

[143] R. Smith, "Composite defects and their detection," *Materials Science and Engineering*, vol. 3, pp. 103–143, 2009.

[144] X. Maldague and S. Marinetti, "Pulse phase infrared thermography," *Journal of Applied Physics*, vol. 79, no. 5, pp. 2694–2698, 1996.

[145] S. Marinetti, Y. Plotnikov, W. Winfree, and A. Braggiotti, "Pulse phase thermography for defect detection and visualization," in *Nondestructive Evaluation Techniques for Aging Infrastructures & Manufacturing*, pp. 230–238, International Society for Optics and Photonics, 1999.

[146] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[147] P. Azad, T. Gockel, and R. Dillmann, *Computer Vision: principles and practice*. Elector International Media BV 2008, 2008.

[148] S. Suzuki, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32–46, 1985.

[149] B. Lakshminarayanan, D. M. Roy, and Y. W. Teh, "Mondrian forests: Efficient online random forests," in *Advances in neural information processing systems*, pp. 3140–3148, 2014.

[150] K. Tateno, F. Tombari, and N. Navab, "Large scale and long standing simultaneous reconstruction and segmentation," *Computer Vision and Image Understanding*, vol. 157, pp. 138–150, 2017.

[151] L. Jeannotte, M. J. Moore, *et al.*, *The State of aging and health in America 2007*. Merck Company Foundation, 2007.

[152] P. Rashidi and A. Mihailidis, "A survey on ambient-assisted living tools for older adults," *Biomedical and Health Informatics, IEEE Journal of*, vol. 17, no. 3, pp. 579–590, 2013.

[153] D. Fischinger, P. Einramhof, K. Papoutsakis, W. Wohlkinger, P. Mayer, P. Panek, S. Hofmann, T. Koertner, A. Weiss, A. Argyros, *et al.*, "Hobbit, a care robot supporting independent living at home: First prototype and lessons learned," *Robotics and Autonomous Systems*, 2014.

[154] F. Cavallo, M. Aquilano, M. Bonaccorsi, I. Mannari, M. Carrozza, and P. Dario, "Multidisciplinary approach for developing a new robotic system for domiciliary assistance to elderly people," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pp. 5327–5330, IEEE, 2011.

[155] S. Coradeschi, A. Cesta, G. Cortellessa, L. Coraci, J. Gonzalez, L. Karlsson, F. Furfari, A. Loutfi, A. Orlandini, F. Palumbo, *et al.*, "Giraffplus: Combining social interaction and long term monitoring for promoting independent living," in *Human System Interaction (HSI), 2013 The 6th International Conference on*, pp. 578–585, IEEE, 2013.

[156] M. Munaro, F. Basso, and E. Menegatti, "Openptrack: Open source multi-camera calibration and people tracking for rgb-d camera networks," *Robotics and Autonomous Systems*, vol. 75, pp. 525–538, 2016.

[157] M. Munaro, A. Horn, R. Illum, J. Burke, and R. B. Rusu, "Openptrack: People tracking for heterogeneous networks of color-depth cameras," in *IAS-13 Workshop Proceedings: 1st Intl. Workshop on 3D Robot Perception with Point Cloud Library*, pp. 235–247, 2014.

[158] M. Munaro and E. Menegatti, "Fast rgb-d people tracking for service robots," *Autonomous Robots*, vol. 37, no. 3, pp. 227–242, 2014.

[159] K. Koide, E. Menegatti, M. Carraro, M. Munaro, and J. Miura, "People tracking and re-identification by face recognition for rgb-d camera networks.," in *Mobile Robots (ECMR), 2017 European Conference on*, 2017.

[160] U. Chaudhary, N. Birbaumer, and A. Ramos-Murguialday, "Brain-computer interfaces for communication and rehabilitation," *Nature Reviews Neurology*, vol. 12, no. 9, pp. 513–525, 2016.

[161] R. Leeb, S. Perdikis, L. Tonin, A. Biasiucci, M. Tavella, M. Creatura, A. Molina, A. Al-Khodairy, T. Carlson, and J. dR Millán, "Transferring brain–computer interfaces beyond the laboratory: successful application control for motor-disabled users," *Artificial intelligence in medicine*, vol. 59, no. 2, pp. 121–132, 2013.

[162] L. Tonin, R. Leeb, M. Tavella, S. Perdikis, and J. d. R. Millán, "The role of shared-control in bci-based telepresence," in *Systems Man and Cybernetics*

(SMC), 2010 IEEE International Conference on, pp. 1462–1466, IEEE, 2010.

[163] R. Leeb, L. Tonin, M. Rohm, L. Desideri, T. Carlson, and J. d. R. Millán, "Towards independence: a bci telepresence robot for people with severe motor disabilities," *Proceedings of the IEEE*, vol. 103, no. 6, pp. 969–982, 2015.

[164] G. Beraldo, M. Antonello, A. Cimolato, E. Menegatti, and L. Tonin, "Brain-computer interface meets ros: A robotic approach to mentally drive telepresence robots," *arXiv preprint arXiv:1712.01772*, 2017.

[165] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[166] B. Siciliano and O. Khatib, *Springer handbook of robotics*. Springer, 2016.

[167] E. Marchand, F. Spindler, and F. Chaumette, "Visp for visual servoing: a generic software platform with a wide class of robot control skills," *IEEE Robotics and Automation Magazine*, vol. 12, pp. 40–52, December 2005.

[168] I. A. Sucan and S. Chitta, "Moveit!," *moveit. ros. org*, 2016.

[169] S. Michieletto, F. Stival, F. Castelli, and E. Pagello, "Teaching door assembly tasks in uncertain environment," in *ISR 2016: 47st International Symposium on Robotics; Proceedings of*, pp. 1–7, VDE, 2016.

[170] F. Stival, S. Michieletto, and E. Pagello, "How to deploy a wire with a robotic platform: Learning from human visual demonstrations," in *27th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2017); Proceedings of*, ELSEVIER, 2017.

[171] S. Michieletto, F. Stival, F. Castelli, M. Khosravi, A. Landini, S. Ellero, R. Landó, N. Boscolo, S. Tonello, B. Varaticeanu, C. Nicolescu, and E. Pagello, "Flexicoil: Flexible robotized coils winding for electric machines manufacturing industry," in *ICRA 2017 Workshop on Industry of the future: Collaborative, Connected, Cognitive*, 2017.

[172] G. Puskorius and L. Feldkamp, "Calibration of robot vision," in *Proc. IEEE Int. Conf. on Robotic and Automation*, 1987.

[173] Y. C. Shiu and S. Ahmad, "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form ax= xb," *ieee Transactions on Robotics and Automation*, vol. 5, no. 1, pp. 16–29, 1989.

[174] J. Schmidt, F. Vogt, and H. Niemann, "Calibration–free hand–eye calibration: a structure–from–motion approach," in *Joint Pattern Recognition Symposium*, pp. 67–74, Springer, 2005.

[175] K. Pachtrachai, M. Allan, V. Pawar, S. Hailes, and D. Stoyanov, "Hand-eye calibration for robotic assisted minimally invasive surgery without a calibration object," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pp. 2485–2491, IEEE, 2016.