1 2 9 0

UNIVERSIDADE Đ
COIMBRA

Filipe Marques Barreto da Cerveira Pinto

OBJECT VERIFICATION AND REGISTRATION OF THE COMPLETE POSE ON A SEMANTIC MAP BUILT BY A MOBILE ROBOT

Dissertação no âmbito do Mestrado Integrado em Engenharia Eletrotécnica e de Computadores,
na especialização de Automação, orientada pelo Professor Doutor Rui Paulo Pinto da Rocha
e apresentada ao Departamento de Engenharia Eletrotécnica e de Computadores
da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro de 2019

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE Đ
COIMBRA

Filipe Marques Barreto da Cerveira Pinto

Object Verification and Registration of the Complete Pose on a Semantic Map Built by a Mobile Robot

Dissertation supervised by Professor Doctor Rui Paulo Pinto da Rocha and submitted to the Electrical and Computer Engineering Department of the Faculty of Sciences and Technology of the University of Coimbra, in partial fulfillment of the requirements for the Degree of Master in Electrical and Computer Engineering, branch of Automation.

Supervisor:
Prof. Rui Paulo Pinto da Rocha

Coimbra, September 2019

This work was developped in collaboration with:

1 2 9 0

UNIVERSIDADE DE
COIMBRA

INSTITUTE OF SYSTEMS AND ROBOTICS
UNIVERSITY OF COIMBRA

DEEC

INGENIARIUS

To my adorable Gina
and family.

Acknowledgments

This dissertation is the result of constant enthusiasm, persistence and commitment during the past eleven months, which could never have been fulfilled without the support and motivation of people close to me.

I thank my family, especially my parents, Clara and Amílcar, and my brother, João Pedro, for their unconditional love, care and support throughout my entire life, especially during this journey. To my grandma, Aida, for her kindness, concern, affection and love. To my grandparents, Alice and Né, for your support and for transmitting me important values. I also thank my girlfriend, Rita, for her love, friendship and support, which were crucial to the fulfilment of this dissertation.

I thank my supervisor, Professor Rui Rocha, for giving me the opportunity to experience a challenge but enriching problem as well as for the continuous encouragement, guidance and for always being an excellent teacher.

Thanks to all the members of Ingeniarius, STOP project and Mobile Robotics Lab, for the good mood and mutual help. Special thanks to Dr. Gonçalo Martins, for all the shared knowledge, motivation, and guidance and friendship.

Last but not least, I thank Gina for being Gina.

Abstract

Since the dawn of humankind, security is an essential asset for human beings. With the latest advances in technology, it is possible to automate security tasks by using patrolling and surveillance robots, therefore improving the overall well-being of people.

In order to perform advanced and useful tasks in everyday environments, such as patrolling, search and rescue, interaction with humans to provide assistance, or object manipulation, robots need to know where they are, what is around them, and which is the arrangement and category of objects next to them. A solution to this problem is based on semantic robotic mapping, which involves building an enhanced representation of the environment that entails not only geometrical information but also semantic information of the objects populating the environment, including their categorisation, location, orientation, description, etc. The application of semantic mapping to mobile robots enables a more qualitative and richer description of the robots surroundings, aiming to improve navigation, task planning, and human-robot interaction.

This dissertation addresses the problem of endowing a mobile robot with the ability to detect and classify objects in indoor structured environments and register their complete pose in a semantic map of the environment. The robot perception system was developed within the STOP project [1], which aims at developing technology based on several scientific contributions on distributed multi-robot patrolling, and developing innovative technical features in order to adapt surveillance robots to real-world scenarios, namely the automatic perception of abnormal situations, as well as resilient operation during long periods of time.

In this dissertation, raw sensor data provided by a RGB-D camera, which collects information of the robot's workspace, is used to build a semantic map.

[1] `http://stop.ingeniarius.pt`

The most important contribution of this work, concerning the state of the art, is the ability to estimate the complete pose of the objects, i.e. not only their position but also their orientation.

This dissertation work encompassed three major milestones. The first one was to attain an object detection and classification system based on a centralised architecture using an artificial neural network. The second major milestone consisted in to develop an object classification system based on a distributed architecture. In order to achieve a distributed architecture, a novel heuristic method for object classification using point clouds and occupancy grids, without resorting to deep learning techniques, was developed. For the experimental validation of the system, which represents the last milestone, a long-term test and a benchmarking were carried out.

Keywords: Mobile Robotics, Object Detection and Classification, Object Complete Pose, Patrolling Robots, Semantic Mapping.

Resumo

A segurança afigura-se, desde o alvorecer da humanidade, como uma questão primordial para o ser humano. Graças aos mais recentes avanços tecnológicos, é possível automatizar tarefas de segurança utilizando robôs de patrulhamento e vigilância, melhorando, deste modo, o bem-estar geral da população.

Com o objetivo de desempenhar tarefas úteis e avançadas em ambientes do dia a dia, tais como patrulhamento de edifícios, busca e salvamento, interação com humanos de forma a garantir assistência ou manuseamento de objetos, os robôs necessitam de saber a sua localização, o que se encontra em seu redor e qual a distribução espacial e a categoria dos objetos que se encontram junto deles. A solução para este problema reside no mapeamento semântico robótico, o qual pressupõe uma construção melhorada da representação do ambiente, o que implica não só informação geométrica, mas também informação semântica relativamente aos objetos existentes no ambiente, incluindo a sua classe, posição, orientação, descrição, estado, etc. A aplicação do mapa semântico aos robots móveis proporciona uma melhor e mais rica descrição do meio envolvente, visando melhorar a navegação, planeamento de tarefas, bem como a interação do homem-robô.

Esta dissertação aborda, fundamentalmente, a questão de como dotar um robot móvel da capacidade de detetar e classificar objetos em ambientes internos e estruturados, registando a pose completa dos mesmos num mapa semântico do ambiente. O sistema de perceção do robô foi desenvolvido no contexto do projeto STOP[2], o qual visa desenvolver tecnologia baseada em vários contributos científicos na área do patrulhamento multi-robô distribuído, e o desenvolvimento de características técnicas inovadoras com o objetivo de adaptar robôs de vigilância a cenários de mundo real, nomeadamente a perceção automática de situações anómalas, bem como a capacidade de operação resiliente durante longos períodos de tempo.

[2] http://stop.ingeniarius.pt

Nesta dissertação, dados provenientes da câmera RGB, os quais contém informação do ambiente onde o robô se encontra, são utilizados na construção do mapa semântico.

A contribuição mais importante deste trabalho, em relação ao estado da arte, é a capacidade de estimar a pose completa dos objetos, ou seja, não apenas a sua posição, mas também a sua orientação.

Este trabalho de dissertação englobou três objetivos principais. O primeiro objetivo consistiu em alcançar um sistema de detecção e classificação de objectos baseado numa arquitectura centralizada utilizando uma rede neuronal artificial. O segundo grande objetivo consistiu em desenvolver um sistema de classificação de objeto baseado numa arquitectura distribuída. De forma se conseguir uma arquitetura distribuída foi desenvolvido um novo método heurístico de classificação de objetos usando nuvens de pontos e grades de ocupação, sem recorrer a técnicas de aprendizagem profunda. Para a validação experimental dos sistemas, a qual representa o último objectivo, foi realizado um teste de longa duração e um estudo de comparativo.

Palavras-chave: Robôs Móveis, Deteção e Classificação de Objetos, Pose Completa de Objetos, Robôs de Patrulhamento, Mapeamento Semântico.

List of Figures

List of Tables

Contents

1

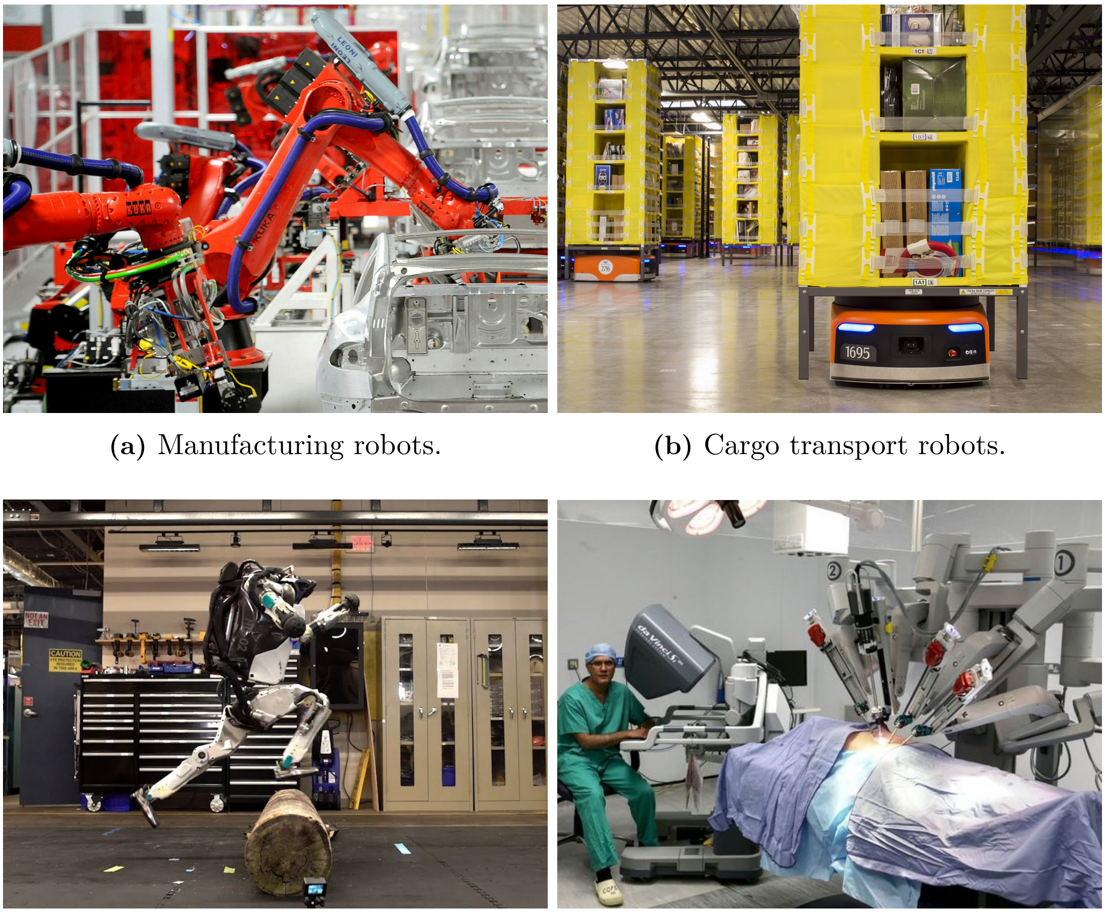
Introduction

“If every tool, when ordered, or even of its own accord, could do the work that befits it (...) then there would be no need either of apprentices for the master workers or of slaves for the lords.” - Aristotle, 322 BC. The human curiosity for machines capable of replacing human labour at performing repetitive tasks, dangerous and heavy work and an increased manufacturing efficiency is not recent. Over the past centuries, through automation and robotics, humans have simplified and improved mankind’s overall quality of life by replacing human labour with specialised machines to perform potentially dangerous tasks, thus helping humans in their daily tasks and improving manufacturing performance as well as reducing costs.

Robots are becoming increasingly present in our lives and they are here to stay. At the beginning of the Industrial Age, early robots were simple machines, limited to perform repetitive and tedious manufacturing tasks in a much more efficient and accurate way than humans. These simple machines are becoming highly capable robots, with the ability to perform different and more complex tasks in uncertain and dynamic environments.

Perception is an essential skill of mobile robots which empowers them to move autonomously in the workspace and make decisions based on the information that they extract from the environment, such as nearby objects and occupation of the space. The ability to perceive allows robots to expand the range of complex tasks they can perform. Some examples of complex tasks that perception enables robots to complete are: patrolling, search and rescue, interaction with humans to provide assistance and object manipulation.

Figure 1.1 represents several robots capable of performing differentiated tasks that contribute to the improvement of the quality of life of human beings and, in general, to the progress of mankind.



(a) Manufacturing robots.

(b) Cargo transport robots.

(c) Atlas advanced humanoid robot.

(d) da Vinci Surgical System.

Figure 1.1: Example of highly capable robots of performing differentiated and complex tasks.

1.1 Relevance of Study

Safety and security have always been a top priority for all living beings. From the smallest ant to the largest mammals, all animals seek to be safe and, therefore, are continuously on the alert for situations of possible danger. The same happens with humans, as the feeling of safety and security is essential so that it is possible to live peacefully, without the feeling of imminent danger. This dissertation is framed in one of the many important fields of contemporary security, security in facilities.

The major research topic of this dissertation addresses the problem of endowing a team of mobile robots with the ability to detect and classify objects in indoor structured environments and register their pose in a semantic map of the environment. More specifically, this work aims at developing an efficient and precise robot perception system capable of detecting and classifying objects of interest in the robots planar workspace, as well as registering the pose of these objects with reference to a global frame of the environment that the robot is patrolling. The registration of the information of the objects in the semantic map consists of storing information regarding the three degrees of freedom (DoF), X, Y and Z, for the objects' position, three DoF, *Roll*, *Pitch* and Yaw, for the objects' orientation and the state of the recognised object.

Benefiting from the knowledge of objects position and orientation, it is possible to assess the state of relevant objects in the context of facilities security. Hence, the system can verify if relevant objects, such as fire extinguishers, are in their correct position, are missing or if they are obstructed. It is also possible to infer the state, open or closed, of doors and windows, as well as exploiting the known orientation of objects to evaluate how open the doors and windows are.

1.2 Main Objectives

This dissertation encompasses several important challenges. Developing an efficient and precise robot perception system capable of detecting and classifying objects of interest in the robot's planar workspace is one of the major objectives of this work, as well as a main challenge. Besides identifying and classifying objects, assessing the objects' pose, which is composed of objects' position and orientation, is also one of the main goals of this dissertation.

In order to keep the acquired information in a model of the environment, it is necessary to create a semantic map of it, which consists of an enhanced representation of the environment that entails not only geometrical information but also semantic information of the objects populating the environment, including their categorisation, position, orientation and description. Although the creation of the semantic map was not the main focus in this dissertation, it is also one of its goals.

Another challenge of this dissertation consists of endowing the system with the ability to, through the analysis of the position and orientation of objects, detect anomalies in the environment and alerting the user in case an object is not in the place where it is supposed to be or if its orientation has changed during a restricted time of the day.

1.3 Outline of the dissertation

This document is divided into several chapters. It starts by presenting in detail the framework of this dissertation, explaining exactly the main objectives, the resources, and requirements, as well as its limitations. The document continues with the description of the implemented architectures, explaining in detail all the modules that compose them. The document ends with the presentation and discussion of the obtained results, as well as the conclusion.

The subsequent chapter presents, in detail, the specific problem that this dissertation proposes to solve, explaining its framework and main objectives.

2

Problem Definition

This dissertation was developed within the Seguranças robóTicos coOPerativos (STOP) project[1], which has the objective of deploying autonomous mobile robot teams in large indoor spaces frequented by people or inhabited, such as shopping malls, stores, open office spaces and services, museums, large warehouses, military facilities, etc., to carry out patrolling and surveillance missions. The developed technology is intended to help humans in monotonous, tedious or repetitive tasks associated with the supervision, monitoring and surveillance of infrastructures, framed in the concept of multi-robot patrolling for the security of buildings and facilities.

In addition to detecting the presence of unauthorised persons in restricted areas, one of the many abilities that surveillance robots must have is to detect anomalies, i.e. security issues, related to relevant objects.

It is essential for facilities' safety to ensure that there are no anomalies related to relevant objects, such as doors, fire extinguishers, and windows. At specific times, such as night time, doors and windows should be properly closed. Fire extinguishers must always be in the position where they are supposed to be, ready for any emergency, and, as the fire escape doors, they must be properly unobstructed.

In order to detect anomalies, a surveillance robot must be capable of recognising and assessing the state of the surrounding objects, classifying them and registering their pose and state in a map of the environment.

Fig. 2.1 gives a general overview of the problem definition, illustrating the main challenge of this dissertation: obtaining the position of objects of interest as well as their orientation.

[1] `http://stop.ingeniarius.pt`

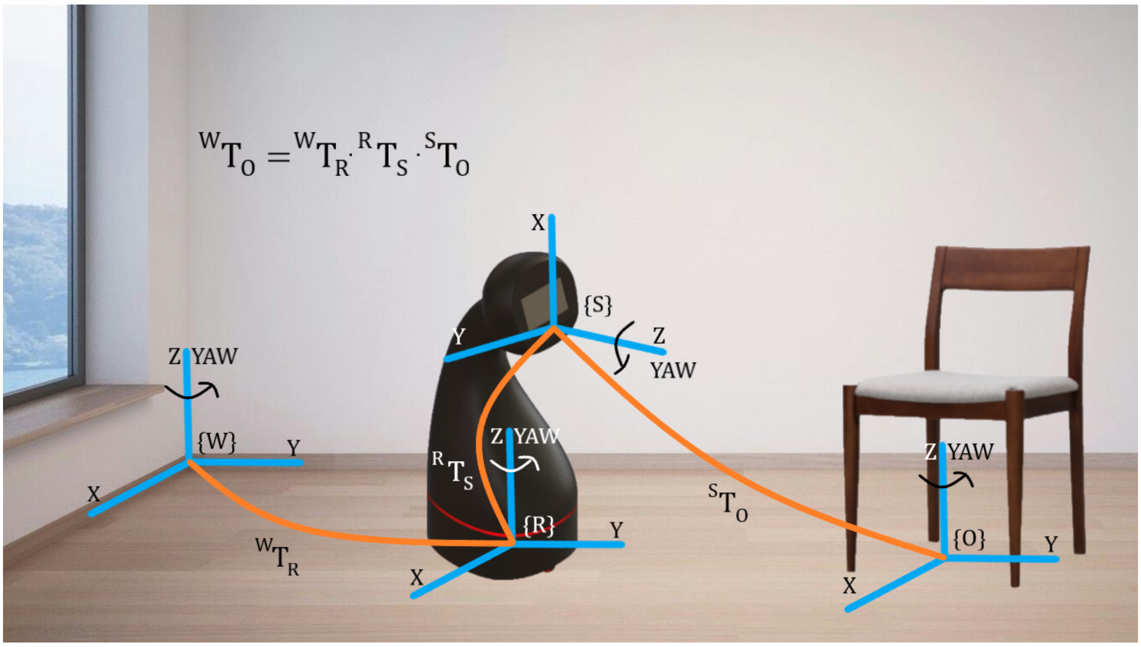


Figure 2.1: Representative illustration of the main challenge of this dissertation and the described problem.

The main challenge of this dissertation work is to detect and identify relevant objects for the security, surveillance and patrolling of facilities in the robots workspace (e.g. a chair, as depicted in the figure) and, for each of those objects, estimate the transformation $^{S}T_{O}$ in order to be able to register its pose in the global reference frame, {W}.

After the computation of transformation $^{S}T_{O}$, the object registration w.r.t. to the robot base {R} and w.r.t. the global reference frame {W} is straightforward, assuming that the transformation between the base of the mobile robot and the RGB-D sensor is fixed and the robot is assumed to be localised[2] w.r.t. global reference frame {W}. The registered objects are used in building and updating a semantic map of the environment to be used by the robot in the surveillance task.

Although the objectives of this dissertation were already summarised in sec. 1.2, they can be more systematically defined as follows:

- Develop an efficient and precise robot perception system capable of detecting and classifying objects of interest in the robots planar workspace;
- Estimation of the pose of detected objects, three DoF for the objects' position, X, Y, and Z and three DoF for the objects' orientation, *Roll*, *Pitch* and *yaw*;

[2] The robot's localization problem is not within the scope of this dissertation.

- Build a semantic map of the indoor environment with information regarding the position and orientation of objects of interest.
- Create a detector of anomalies capable of alerting the user in case an object is not in the place where it is supposed to be, is obstructed, or in case its orientation has been changed during a restricted time of the day.

2.1 Resources, Requirements and Limitations

The system herein presented includes its own limitations, requirements and constraints. Consequently, the architectures developed throughout this dissertation inherited and took into consideration requirements and restrictions of the STOP project.

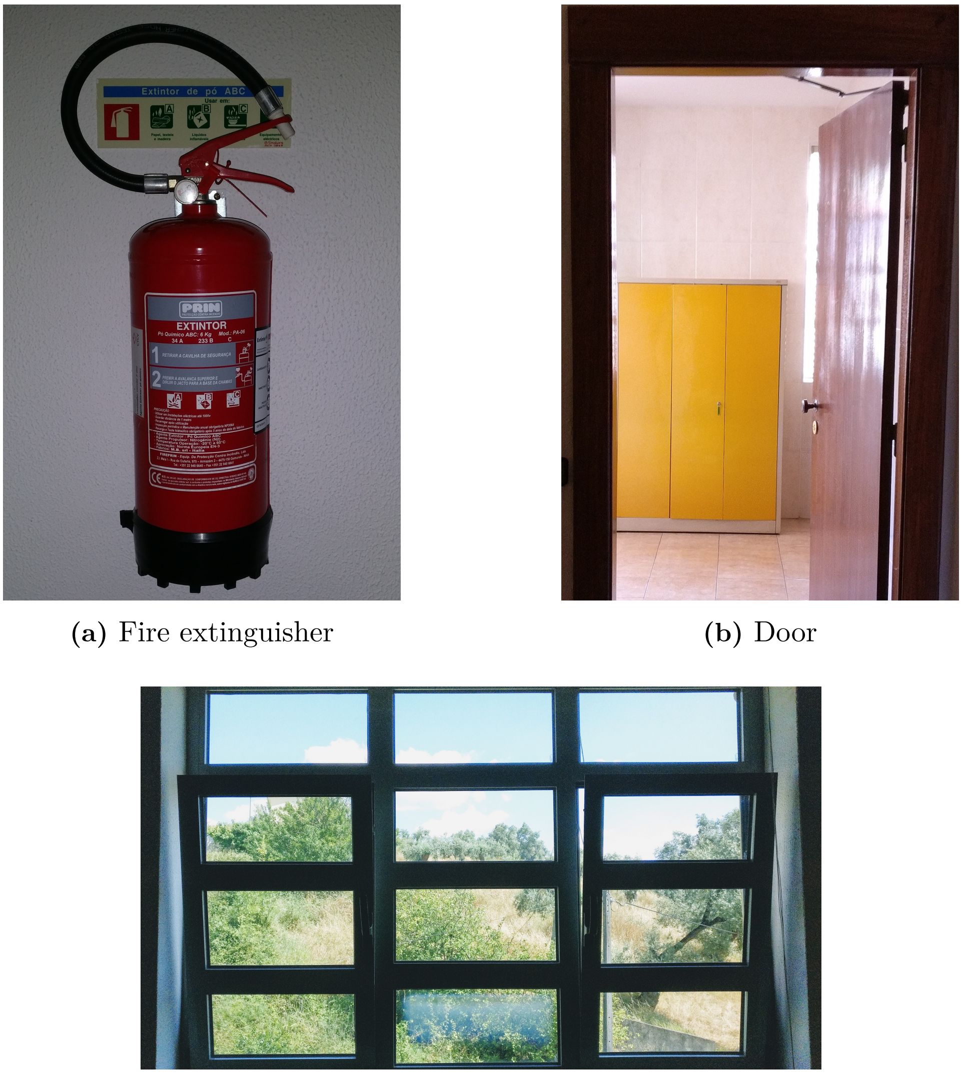
The STOP project architecture is based on multi-robot perception, where a team of mobile robots performs cooperative monitoring and surveillance tasks. The patrolling team comprises essentially one central server computer and three mobile robots, which are connected through a wireless network. The three mobile robots are assumed to be homogeneous and provide a given degree of redundancy and spatial distribution to make the robotic team capable of patrolling effectively a fairly large indoor structured environment[3].

The patrolling test area of the STOP project is well defined and known from the outset. Consequently, the classes of objects that are relevant to be detected and have their status inferred, in the context of security and patrolling of the facility, are also defined from the beginning and the position where each one of these object is supposed to be is known. Knowing the location of relevant objects at the outset is a valuable asset, since it opens the possibility of implementing tailored methods to solve the problem. Within the scope of the STOP project, the main relevant objects that require their condition to be assessed are: fire extinguishers, doors and windows.

Figures 2.2a and 2.2b depict, respectively, a fire extinguisher and a door, which can be found in the test facility. Figure 2.2c depicts a typical window that can be found in the test facilities.

The server computer stands out as the only computation node of the system endowed with a dedicated graphics processing unit (GPU), NVIDIA GTX 1070 Ti, an essential piece

[3] Experiments have been carried out in an environment whose size is 60 x 10 meters.



(a) Fire extinguisher

(b) Door

(c) Window

Figure 2.2: Examples of objects relevant to the safety of the facility that can be found in the test area.

of hardware to run software modules that require large amounts of parallel processing, such as running Convolutional Neural networks. It also contains a Intel core I7 central processing unit (CPU).

Patrolling robots have the purpose of moving along indoor environments of interest, acquiring useful information in the format of colour and depth images, to gauge the state of objects relevant to the safety of facilities. These mobile robots do not have a powerful graphics card, such as the one found on the server computer, possessing only an Intel core i5 6th generation CPU, powerful enough to perform lighter tasks. All three patrolling robots contain, among other sensors, a RGB-D camera. This camera sensor is the main sensor in this dissertation work, as it is used to collect data, colour and depth images of the environment in the neighbourhood of the mobile robots. Figures 2.3 and 2.4 depict, respectively, one of the patrolling robots and the server computer.

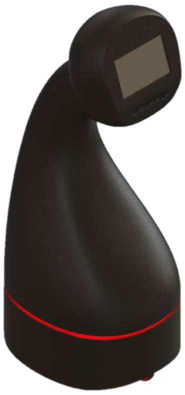


Figure 2.3: Illustration of one of the patrolling robots.

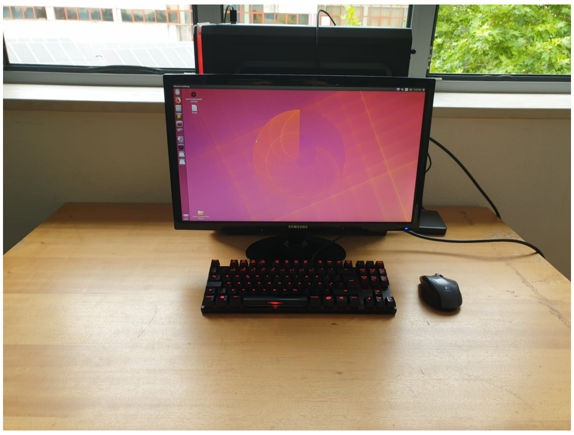


Figure 2.4: Server computer of the patrolling team

One of the main requirements of the STOP project system is scalability. This requirement is essential so that, in the future, it will be possible to increase the number of mobile patrolling robots without major hardware and software design changes. This requirement impacts two fundamental parts of the system: bandwidth usage so that robots can communicate among each other and with the server; and the use of server processing resources that, due to the fact that the server computer is the only computational node endowed with a powerful GPU, it limits the use of deep learning to perform object recognition and classification.

Another major asset for the STOP project system is the Wi-Fi communication network, which supports wireless communication in real time among the server and mobile robots. Although this is a very important asset, it is something that cannot be entirely reliable due to its limitations in the testing facility. It has been found that robots lose access to network frequently due to a few dead zones. Also, in certain areas of the testing facility, wireless connection reaches high latency peaks, causing a delay in data transmission and a decrease in the connection's bandwidth, thus restricting the system from harnessing the full potential of the wireless network.

The lighting conditions of the test facility are poor. This condition causes less contrast in the colours of the environment, impairing the collection of accurate colour information by the patrolling robots. Figure 2.5 depicts the area where the tests took place.

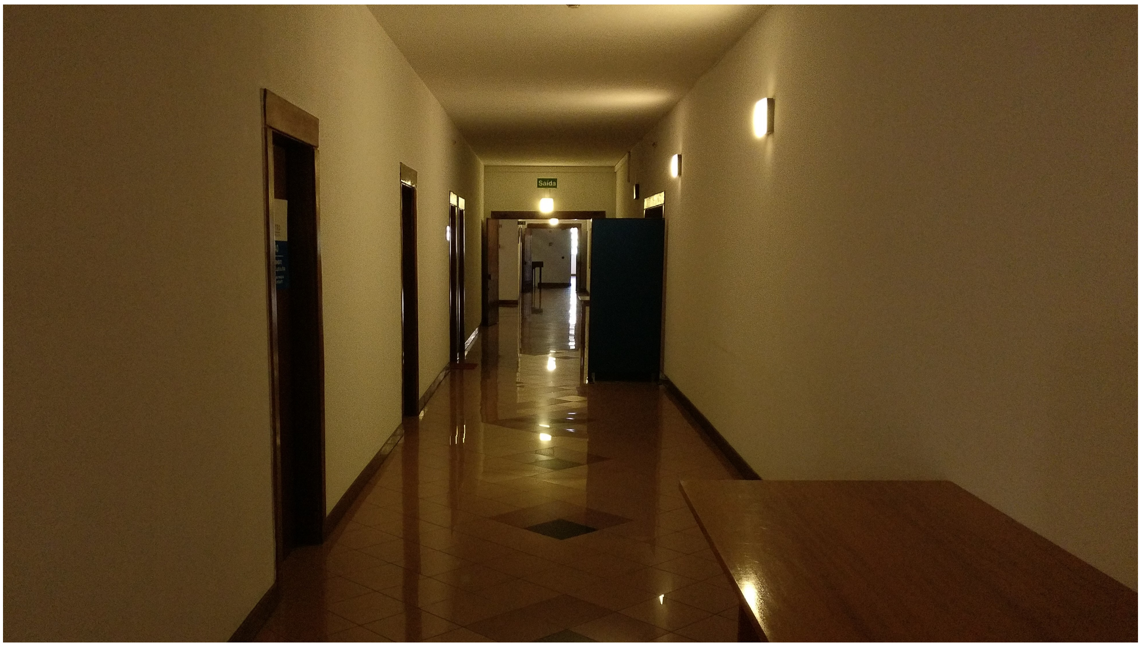


Figure 2.5: Photo of the area where the tests took place.

2.2 Base Software and Middleware

All computation nodes of the system, i.e. the three mobile robots and the server computer, run the Ubuntu operating system, a free and open-source Linux distribution. On top of Ubuntu, the Robot Operating System[4] is used. ROS is a flexible framework for developing robot software. It consists of a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behaviour across a wide variety of robotic platforms. ROS was built from the ground up to encourage collaborative robotics software development, such as the one developed throughout the STOP research project. It was designed specifically for groups to collaborate and build upon each other's work. ROS has several versions. The one used in this work was ROS Kinetic due to the fact that ROS kinetic is the most advanced version at the start date of the STOP project.

[4] `http://www.ros.org/`

2.3 Data Acquisition

The RGB-D camera is the main sensor used throughout the development of this work. It provides data from the robot's workspace, namely colour and depth information, in order to be further processed. This type of camera captures a fairly good resolution colour image and depth estimation per-pixel, at a low cost, making it a good asset for research with mobile robots and computer vision applications. Each of the three patrolling robots are equipped with an Orbbec Astra RGB-D[5] camera. This camera provides high-end responsiveness, depth measurement, smooth gradients and precise contours, as well as the ability to filter out low-quality depth pixels. Table 2.1 presents its main specifications according with information provided by its manufacturer.

Parameter	**Value**
Range	0.6m - 8m
FOV	60°H x 49.5°V x 73°D
RGB Image Resolution	640 x 480 @30fps
Depth Image Resolution	640 x 480 @30fps
Size	165mm x 30mm x 40mm
Temperature	0°C 40°C
Connection and Power Supply	USB 2.0
Power Consumption	<2.4W
Operating Systems	Android/Linux/Windows 7/8/10
SDK	Astra SDK or OpenNI
Microphones	2 (built in)

Table 2.1: Orbbec Astra technical specifications.

Figure 2.6 shows the Orbbec Astra RGB-D sensor installed on top of each one of the mobile patrolling robots. The RGB-D camera collects data from the environment and the camera's driver publishes it in ROS topics. Among the different published topics there are, for example, colour and depth images, camera parameters, point clouds as well as images in compressed format.

Figure 2.6: Orbecc Astra RGB-D sensor (reproduced from the Orbbec website).

[5] `https://orbbec3d.com/product-astra-pro/`

Fig. 2.7 depicts the covered patrolling robot. Fig. 2.8 depicts the uncovered patrolling robot and highlights its Orbecc Astra RGB-D sensor.

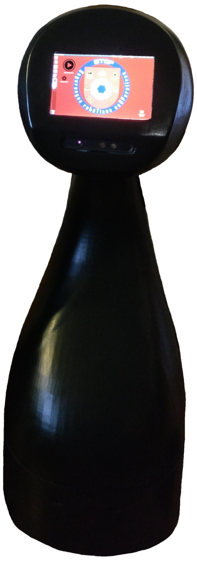


Figure 2.7: Covered patrolling robot.

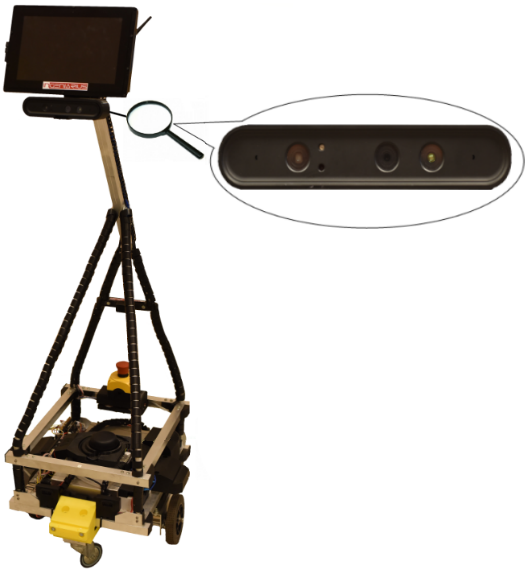


Figure 2.8: Uncovered patrolling robot.

2.4 Summary

This chapter began by introducing the reader to the STOP project, in which this dissertation is tailored from. Then it presented the major research topics of this dissertation as well as its goals. In the next section, the available resources, requirements, limitations of the STOP project, and the characteristics of the test facility are described. Afterwards, the software and middleware that underpins the entire system was introduced. This chapter concluded with a detailed presentation of the main sensor used in this dissertation.

The following chapters, chapters 3 and 4, present two different architectures which were designed to fulfil the requirements of the object detection module of the STOP project, taking into account the restrictions and limitations of the system, not disregarding the latest state-of-the-art methods, and aiming to achieve an efficient and precise robot perception system.

3

Centralised Architecture based on Deep Learning

Object classification, Semantic Segmentation and Object Detection are frequently misinterpreted terms. In object classification, a certain class is usually assigned to an image or point cloud. In opposition, with Semantic Segmentation, labelling is performed for each individual pixel or point. Qi [10] proposed *PointNet Deep Learning on Point Sets for 3D Classification and Segmentation*, which is divided in classification and semantic segmentation. The *PointNet Classification* method assigns one and only one label to the point cloud being classified. In opposition, *PointNet Semantic Segmentation* assigns a label to each element, i.e point, of the point cloud. In the case of object detection, the classification is accomplished by placing bounding boxes around the objects, assigning them a label. Redmon [13] proposed *YOLO You Only Look Once*, which is a perfect example of an object detection system.

This chapter presents one of the two architectures developed throughout this dissertation work: the centralised architecture. Both architectures were designed to address the problem defined in chapter 2.

The centralised architecture uses deep learning techniques. Deep learning techniques involve large amounts of processing, thus requiring a powerful GPU, something that is only available on the server computer, according to the assumptions presented in section 2.1 of the system. As a result, most of the software modules comprising this architecture run on the server computer, centralising therein the objects' perception relative to every robot in the patrolling team. Therefore, the use of deep learning approaches requires the system architecture to be centralised.

This chapter starts by presenting the diagram of the developed centralised architecture. The architecture is divided into several modules and, as a result, this chapter has different sections for each one of the modules comprising the architecture. Each of these subsequent sections starts by presenting, individually and in detail, the specific subproblem that led to the need for the implementation of that module, followed by a review of the state-of-the-art methods for solving that subproblem and, finally, the procedure implemented to solve it.

3.1 System Architecture

Fig. 3.1 represents a diagram of the centralised system architecture developed in this dissertation. It was designed to meet the requirements of the object detection module of the STOP project, taking into account the restrictions and limitations of the system described in sec. 2.1, not disregarding the latest state-of-the-art methods. As can be seen in the diagram, the raw data collected by the patrolling robots is transmitted through a wireless connection to the server, where it is subsequently processed, thus making this software system centralized. The diagram presented below intends to give the reader an overview of the centralized architecture. It will be referred multiple times throughout this chapter to be explained in detail.

In order to better understand the decisions taken during the development of this centralised architecture, it is necessary to understand the core of it. Hence, the following section explains the core of this architecture, i.e. the object detection method implemented.

3.2 Object Detection and Classification

Nowadays, object detection is widely used, whether to perform surveillance and patrolling tasks, assist in the autonomous driving of vehicles, object recognition, detection of anomalies in a restricted environment or to endow the robots with more sophisticated capabilities.

There are several algorithms to perform real-time object recognition and classification

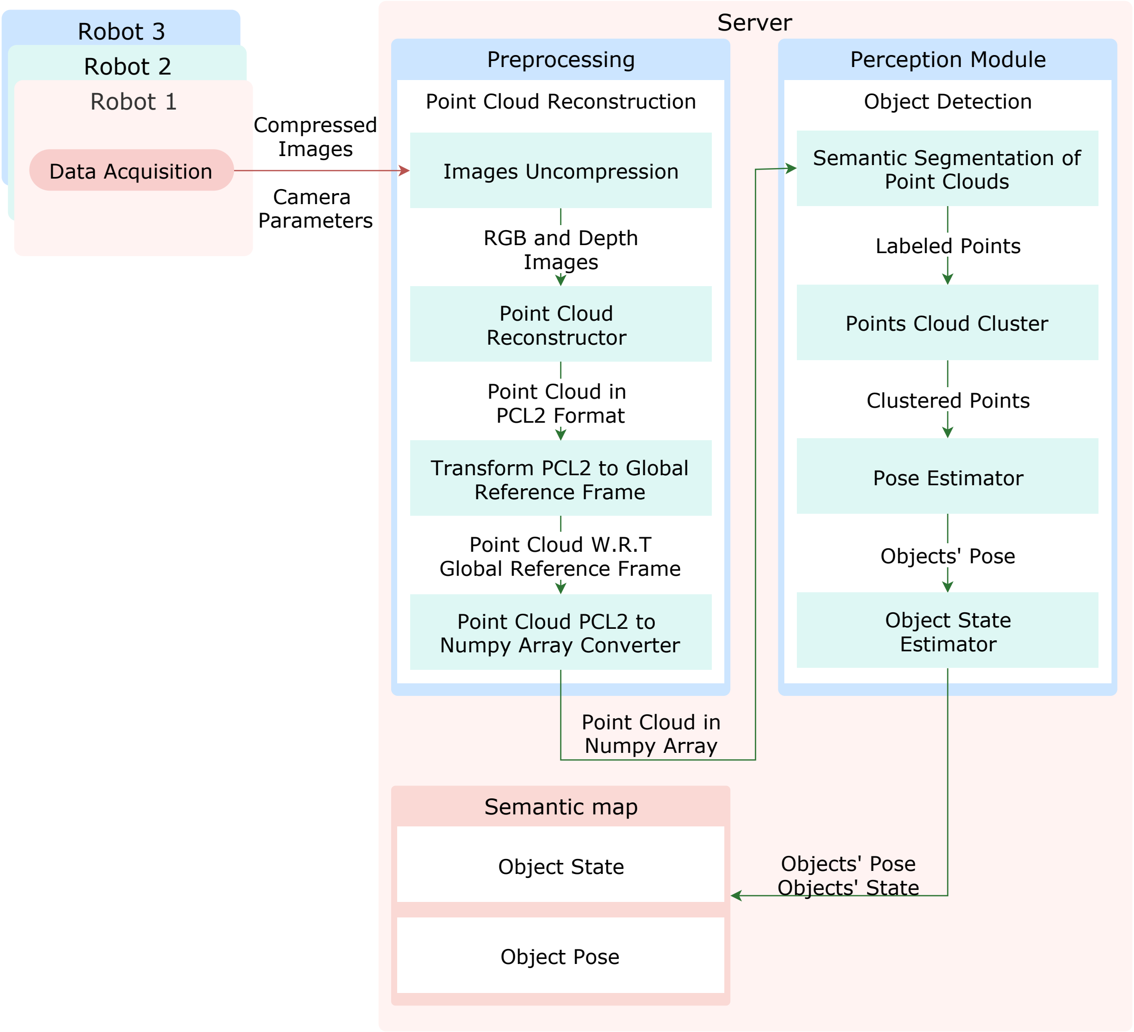


Figure 3.1: Diagram of the centralised architecture based on deep learning.

based on colour images. Most of them are based on computer vision and deep learning techniques that require a powerful GPU. Consequently, the amount of processing power required to run deep learning algorithms usually surpasses the computational power available in a mobile robot, making it necessary to rely on the resources available on the server computer, as shown in Fig. 3.1.

Object detection, object classification, semantic segmentation and object detection are frequently misinterpreted terms. Object detection classification is accomplished by placing bounding boxers around the objects, assigning them a label. Redmon [13] proposed (*YOLO – You Only Look Once: Unified, Real-Time Object Detection*, which is a state-of-the-art, real-time object detection system. This technique applies a single neural network to a single RGB image. The image is divided into regions, predicting probabilities and bounding boxes related to each region. The predicted probabilities weigh the bounding boxes. *YOLO* has been in constant improvement, and it is currently in its third version, *YOLOv3*, which is extremely fast and accurate.

Object classification consists of labelling an image or point cloud with just one label. In opposition, with semantic segmentation labelling, each individual pixel of an image or element, i.e point, of a point cloud gets labelled. Qi [10] proposed *PointNet – Deep Learning on Point Sets for 3D Classification and Segmentation*. PointNet is composed of a classification technique, which takes as input a point cloud and outputs a label for the entire input, and semantic segmentation, which takes as input a colored point cloud and outputs a label per point for each point of the input. PointNet is a unified deep learning architecture that directly takes colour and depth information in the format of colour point clouds as input. It outputs either class labels for the entire input or per point segment/part labels for each point of the input. This method stands out for its good test results achieved on the *Stanford Large-Scale 3D Indoor Spaces Dataset (S3DIS)* [1]. This technique, as other aforementioned, also had improvements. Qi [11] also proposed *PointNet++ – Deep Hierarchical Feature Learning on Point Sets in a Metric Space*, which significantly improves the classification and segmentation achieved by the first version of PointNet in exchange for higher processing power requirements, i.e. a significantly higher GPU resource usage. Due to the higher processing power requirements, this technique was not implemented.

Girshick [4] proposed *R-CNN: Region-based Convolutional Networks for Accurate Object Detection and Segmentation*, a state-of-the-art Convolutional Neural Network (CNN) -based deep learning object detection technique that uses selective search to extract from images 2000 regions, calling it regions proposals. These regions are fed into a CNN in order to extract features, which are subsequently evaluated by a Support Vector Machine (SVM) in order to assess the object classification. This method cannot be executed in real-time as it takes a large amount of time to evaluate each test image. Girshick also proposed a technique based on *R-CNN* with several improvements, *Fast R-CNN* [3], which allows to perform object detection and classification in a shorter period of time than *R-CNN*. This technique's improvement advanced farther than Girshick's work with the proposal of Ren, *Faster R-CNN*, [14], a state-of-the-art method capable of performing real-time object

[1] `http://buildingparser.stanford.edu/method.html`

detection.

Recently, Park [9] proposed *Multi-Task Template Matching for Object Detection, Segmentation and Pose Estimation Using Depth Images*. This method stands out for not requiring a further CNN training so that the system can infer the 6D pose and segmentation of new objects, as well as for outperforming baseline methods that use colour and depth information.

Qin [12] proposed *ThunderNet: Towards Real-time Generic Object Detection*, a light-weight two-stage generic object detector focused on efficiency but still capable of achieving good accuracy. This method is based on RGB images and uses the input resolution of 320*320 pixels. On PASCAL VOC and COCObenchmarks, when compared with light-weight one-stage detectors, *ThunderNet* achieves superior performance with only 40% of the computational cost.

Amongst the object detection methods presented above, *PointNet* was the one that stood out the most due to its good results, relative low GPU resources consumption as well as for using colour and depth information captured by the RGB-D camera. The *PointNet* method is divided into three parts: Classification, Part segmentation, and Semantic Segmentation. The *PointNet Semantic Segmentation* algorithm was the method adopted in this work to perform the task of detecting and classifying objects of interest. It takes as input a coloured point cloud and outputs a properly labelled point cloud per point. Figure 3.2 uses an image reproduced from the PoinNet paper [2] to show an example of the output of the *PointNet Semantic Segmentation* method for different input point clouds.

PointNet Semantic Segmentation method requires as input a numpy array with seven columns and an undefined number of rows. Table 3.1 shows the input format of this method. The first three columns contain information about the spatial position of points, the following three columns contain colour information and the last column contains ground truth information about the label of the points, where the integer numbers represent a specific class. The information in the last column is only used to evaluate *PointNet Semantic Segmentation* performance in the case of a test with a suitable data set, such as the Stanford Large-Scale 3D Indoor Spaces Dataset (S3DIS).

The output of the *PointNet Semantic Segmentation* method is a numpy array with

[2] `https://arxiv.org/pdf/1612.00593.pdf`

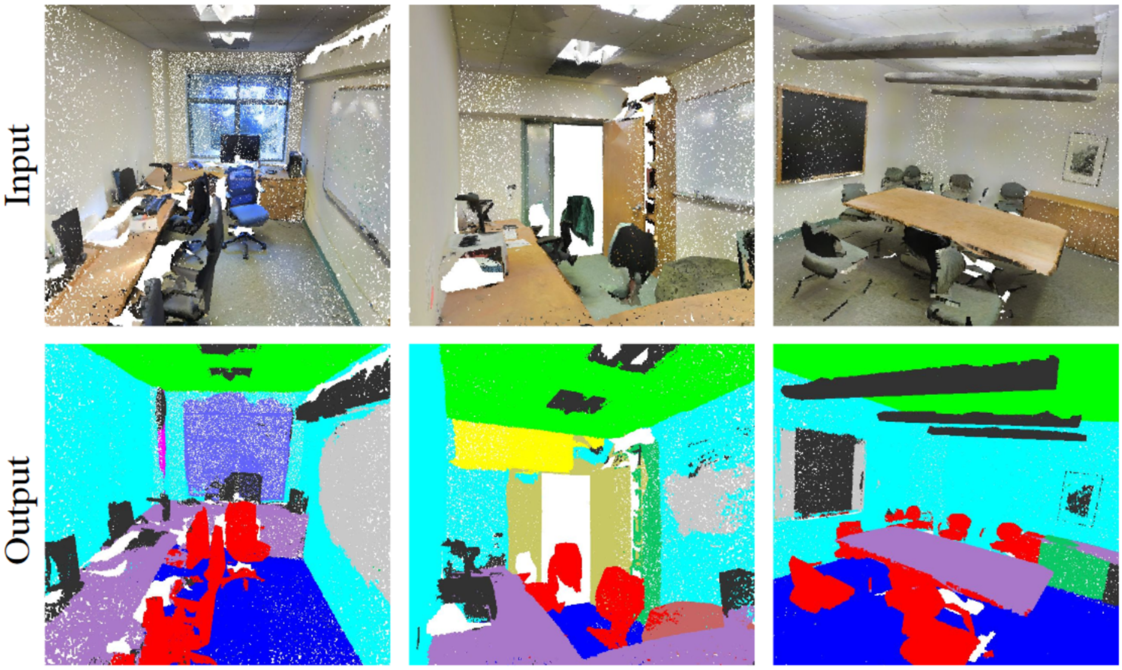


Figure 3.2: Example of PointNet semantic segmentation input and output, extracted from the authors' paper.

	Coordinates			Colour			
	X	Y	Z	R	G	B	Ground Truth
Points	1.42	2.32	2.12	100	100	100	10
	1.23	2.53	2.09	244	122	121	12
	1.44	2.42	2.11	255	0	0	12
	...	...	...	...	...	...	...

Table 3.1: PointNet semantic segmentation input numpy array.

six columns and an undefined number of rows. Table 3.2 shows the output format of the this method. The first three columns contain information about the spatial position of points and the last three columns contain the label of the points in a colour codification. Each colour represents a different label. For example, the colour red (255, 0, 0) means that *PointNet Semantic Segmentation* assessed that the point belongs to the class of chairs.

3.3 Preprocessing

This section explains in detail all the necessary steps between the acquisition of data through the RGB-D sensor and the input of data in the object detection presented in the previous section, i.e. the *PointNet Semantic Segmentation*. The process begins with the data acquisition through the RGB-D sensor presented in sec. 2.3. Through the Orbbec

	Coordinates			Colour		
	X	Y	Z	R	G	B
Points	1.42	2.32	2.12	255	0	0
	1.23	2.53	2.09	0	255	0
	1.44	2.42	2.11	0	0	255
	...	...	...	...	...	...

Table 3.2: PointNet semantic segmentation output numpy array.

Astra's drivers, the data recorded is published in different ROS topics. Among the various types of information published are colour and depth images, camera parameters, point clouds, as well as colour and depth images in compressed format.

Considering that the server computer is the only element of the system endowed with the required hardware to run algorithms based on deep learning, like the *PointNet*, it is necessary to transmit the captured data to the server computer through the Wi-Fi connection. *PointNet Semantic Segmentation* takes as input a coloured point cloud, which is published by the RGB-D camera's driver in the ROS topics *camera/depth_registered/points* (point cloud containing RGB data).

Other ROS topics are published by the RGB-D camera's driver:

- */camera/rgb/image_ rect_color* for the rectified RGB image;
- /camera/depth_registered/hw_registered/image_rect/compressedDepth for the depth image registered w.r.t RGB camera;
- */camera/depth/image_rect* for the rectified depth image;
- */camera/rgb/image_rect_color/compressed* for the RGB image compressed through the 32FC1 format.

Bearing in mind that one of the main requirements of the STOP project system is scalability, and given that transmitting point clouds through the wireless connection consumes prohibitive amounts of bandwidth, doing this is unfeasible. The solution adopted was to take advantage of the colour and depth compressed images provided by the camera driver, as the transmission of compressed images requires a much lower bandwidth.

Table 3.3 presents the required bandwidth to transmit coloured point clouds, colour and depth images and compressed colour and depth images. Comparing the original, provided by the RGB-D camera driver, against the reconstructed point cloud, it was also found that the images compression does not have a significant impact on the quality of the reconstructed point clouds.

Tipe of data	**Average Bandwidth Required**		**Average Frequency**
Colour Point Cloud	290.49MB/s		30Hz
Colour Images	36.85MB/s	64,78MB/s	30Hz
Depth Images	27.93MB/s		
Compressed Colour Images	743.73KB/s	1.15MB/s	30Hz
Compressed Depth Images	410.75KB/s		

Table 3.3: Comparison of bandwidth requirements for the transmission of images acquired by the Orbbec Astra RGB-D sensor.

However, transmitting compressed images leads to another problem because *PointNet Semantic Segmentation* only accepts as input coloured point clouds. The preprocessing module depicted in figure 3.1 presents the adopted approach to transform compressed colour and depth images into coloured point clouds. As it is possible to observe, the preprocessing module on the server computer starts with the decompression of the received images. In order to decompress colour and depth images, the ROS package *image_transport*[3] was used, which provides helpful tools for image and video transmission in low-bandwidth conditions.

Then, the decompressed images enter the next module, *Point Cloud Reconstruction*, which uses colour and depth images, along with camera parameters, to build a coloured point cloud. This module is based on the ROS package *depth_image_proc* [4] which contains nodelets that help in the processing of depth images.

The next module, *Transform PCL2 to Global Reference Frame*, as its name suggests, transforms the point clouds from the RGB camera frame to the global reference frame. This module is of great importance because, after the point cloud is processed by *PointNet Semantic Segmentation*, the objects are immediately referenced w.r.t. the global reference frame. There is an intermediate step in transforming point clouds from the sensor reference frame to the global reference frame: the transformation to the robot base. Since the robot is assumed to be localised w.r.t. the global reference frame $\{W\}$, and the transformation between the sensor reference frame $\{S\}$ and the robot base $\{R\}$ is fixed, calculating the

[3] `http://wiki.ros.org/image_transport`

[4] `http://wiki.ros.org/depth_image_proc`

transformation ${}^{W}T_S$ is straightforward.

Equation 3.1 shows the transformation that occurs in the *Point Cloud PCL2 to Numpy Array Converter* module. This last module of preprocessing transforms a point cloud that is in PCL2 format to a numpy array with seven columns and arbitrary high number of rows. The first three columns contain information about the spatial position of points, the following three columns contain points colour information and, the last column is filled with zeros as the objective is not to evaluate the performance of the Point semantic segmentation. Table 3.4 shows the output format of the preprocessing module.

$$ {}^{W}T_S =^{W} T_R . {}^{R}T_S \tag{3.1} $$

	Coordinates			Colour			
	X	Y	Z	R	G	B	Ground Truth
Points	1.42	2.32	2.12	100	100	100	0
	1.23	2.53	2.09	244	122	121	0
	1.44	2.42	2.11	255	0	0	0
	...	...	...	...	...	...	...

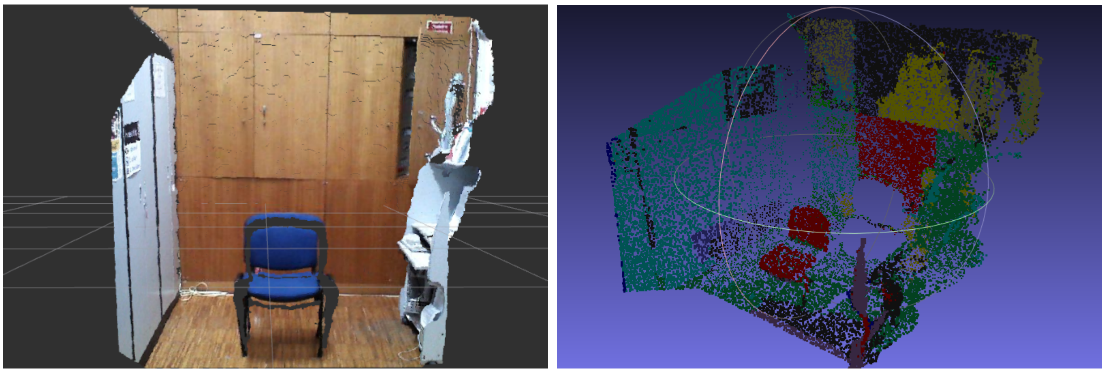
Table 3.4: Output of *Preprocessing Module*.

3.4 Perception Module

This section presents in detail the submodules that comprise the *Perception Module* represented in the diagram of the centralised architecture based on deep learning (see Fig. 3.1), besides the *Object Detection* submodule which was described in sec. 3.2.

Figure 3.3 exhibits the input and output of the PointNet Semantic Segmentation for a colour point cloud acquired at Mobile Robotics Lab (MRL), at ISR Coimbra. The PointNet Semantic Segmentation considers that each of the different colours belonging to the points corresponds to a different class of objects, i.e the red points belongs to chairs, the turquoise points belong to walls, etc. With the information provided by PointNet Semantic Segmentation, it is possible to assess the output label for each one of the points. However, it is impossible to know how many objects are in the area or the localisation of them.

Looking closely at the input and output images in Fig. 3.3, it is possible to observe that *PointNet Semantic Segmentation* is far from being perfect. However, it is still capable of correctly identifying the points that belong to the chair.



(a) Input

(b) Output

Figure 3.3: PointNet Semantic Segmentation at MRL, ISR Coimbra.

3.4.1 *PointNet Semantic Segmentation* - Training

After analysing an example provided by the authors of PointNet based on the *S3DIS – Stanford Large-Scale 3D Indoor Spaces Dataset* (see the footnote in sec. 3.1) which input data is required by Pointnet to segment a point cloud, PointNet was trained with a dataset acquired in the test facility (see Fig. 2.5). The *meshlab*[5] tool was used to trim raw point clouds points that are irrelevant w.r.t. the objects of interest.

3.4.2 Point Clouds' Clustering

The submodule *Point Cloud Cluster* was implemented to solve the aforemetioned problem. *Point Cloud Cluster* takes as input a point cloud in which all the points belong to just one class of objects. Figure 3.4 gives an example of the input of this method, in which it is only visible red points, i.e. all the points that *PointNet Semantic Segmentation* considers that belong to the class of chairs.

In this real environment case scenario, *PointNet Semantic Segmentation* is disappointing as the vast majority of points are wrongly classified as belonging to a chair.

[5] `http://www.meshlab.net/`

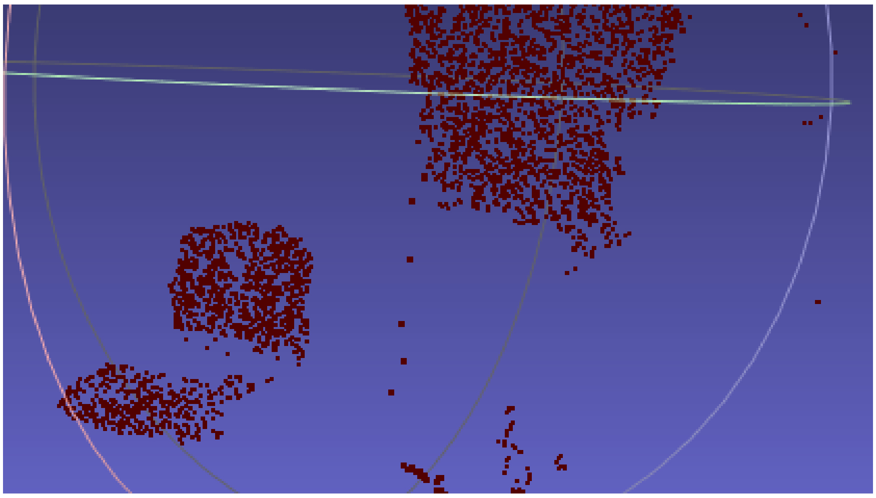


Figure 3.4: Output of PointNet Semantic Segmentation.

Clustering techniques, in addition to allowing us to determine how many objects of the same class are in a given space, also have the ability to improve the overall results of the perception system.

The assortment of techniques to perform clustering is vast as several distinct clustering algorithms have been proposed in the last few years. Ester [2] presented *DBSCAN – Density-Based Spatial Clustering of Applications with Noise*, which became a widely used clustering technique. This technique defines clusters by iteratively computing the distance of each point to its neighbourhood points, given a minimum number of points and a minimum distance threshold. Ng [8] proposed *Spectral Clustering*, a easy way to implement a clustering algorithm with good experimental results on distinct challenging clustering problems. Ankerst [1] presented *OPTICS: Ordering Points To Identify the Clustering Structure*, a versatile clustering technique able to identify clusters with varying densities and requiring very little parameter tuning. Müllner [6] presented *Modern hierarchical, agglomerative clustering algorithms*, where efficient hierarchical agglomerative clustering algorithms, from a practical point of view, are presented. Fig. 3.5 presents a comparison between the clustering algorithms that best fit the problem aforementioned.

Among all the clustering techniques methods presented above, *DBSCAN – Density-Based Spatial Clustering of Applications with Noise* was the one that stood out the most due to its good clustering results and fast performance. Therefore, this was the algorithm adopted in this work to cluster the output of *PointNet Semantic Segmentation*. By defining the DBSCAN parameters per class of object, such as the maximum distance between two

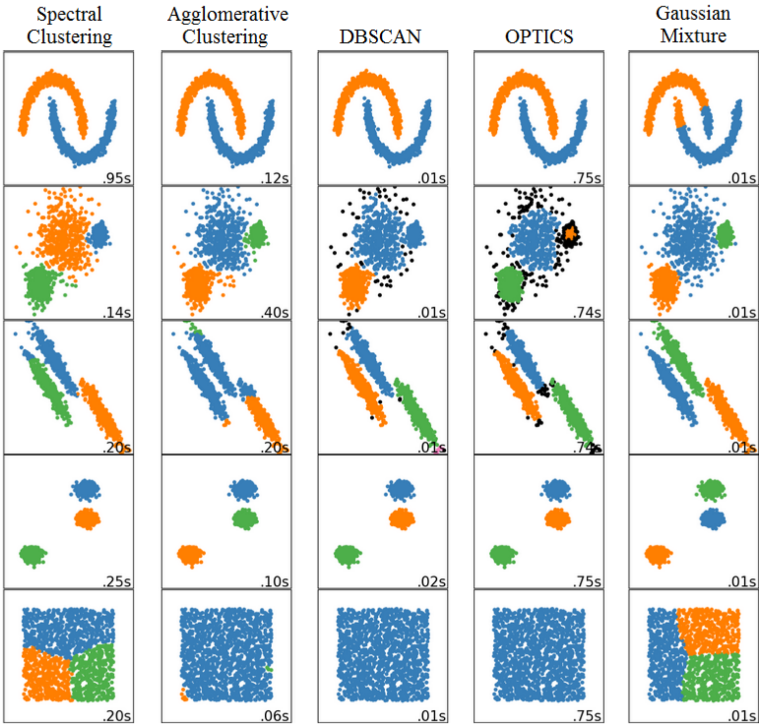


Figure 3.5: Performance comparison of different clustering methods. Adapted from: Scikit-Learn Plot Cluster Comparison.

points for one to be considered as in the neighbourhood of the other and number of points in a neighbourhood for a group of points to be considered as a cluster, it is possible to discard the clusters that clearly do not correspond to the object we are analysing.

The results of the implementation of the clustering algorithm, DBSCAN, are shown in Fig. 3.6. The figure shows a marker placed exactly where the chair is located. Thus, it is concluded that the DBSCAN is capable of dealing with the vast majority of false positives given by the *PointNet Semantic Segmentation*.

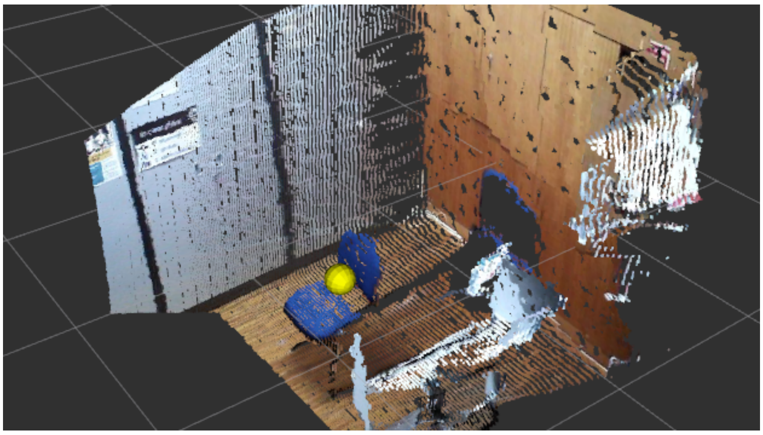


Figure 3.6: Chair localised after the implementation of DBSCAN clustering method.

3.5 Preliminary tests of the centralised architecture.

In order to ensure the correct operation of the system, preliminary tests were conducted in a real-world environment, at the 3rd floor of the "Centro Tecnológico da Cerâmica e do Vidro (CTCV)" facilities.

Unfortunately, under these conditions, besides the fact that PointNet Semantic Segmentation provides a lot of false positives, it is also not capable of providing some true positives, as is the case in the laboratory testing environment. Therefore, the DBSCAN cannot operate under these circumstances and the results are below expectations.

The only factor that changes between the test environment in the laboratory and the real-world environment, at the 3rd floor of the "Centro Tecnológico da Cerâmica e do Vidro (CTCV)" facilities, is the lighting conditions of the environment, which is poor at the test facilities. Therefore, it is concluded that good lighting of the environment is a crucial factor for the proper functioning of the *PointNet Semantic Segmentation of PointNet.*

3.6 Summary

This chapter presents the centralised architecture and the modules that compose it. Relevant techniques for detecting and classifying objects are mentioned, as well as clustering methods. Preliminary test results performed in the laboratory environment and in the real-world environment were presented.

Unfortunately, improving lighting conditions was not an option. Therefore, to handle the problem of the lack of light in the test environment, a new architecture based only on PointClouds without colour information was developed. This architecture is presented in detail in chapter 4.

-

4

Distributed Architecture based on Spatial Template Matching

Unlike the system implemented in the centralised architecture, in which the data processing is carried out on the server computer, the architecture presented in this chapter is based on the concept that the data processing necessary for the correct operation of the system is carried out individually by each of the patrolling robots. Since the data processing is carried out on the patrolling robots, the developed system is a distributed data processing system.

In order to be possible to implement a distributed architecture, the system cannot rely on deep learning techniques that require a discrete GPU, since the robots do not possess one. Therefore, the architecture presented in this chapter consists of an alternative heuristic approach to perform object recognition and classification without resorting to deep learning based techniques.

The distributed architecture takes advantage of the fact that the map of the test facility, as well as the location of the objects of interest relating to the security of the building, are known from the outset.

At the core of the distributed architecture is a *Classifier based on Spatial Template Matching* (CboSTM), which was developed throughout this dissertation and stands out for not relying on any deep learning techniques. The CboSTM infers the label of objects through the analysis of three-dimensional point clouds.

4.1 Architecture

Fig. 4.1 presents the distributed system architecture diagram developed throughout this dissertation, which stands on the CboSTM. As it is possible to perceive in the diagram, this architecture relies on a distributed system approach in which each one of the patrolling robots processes its own data.

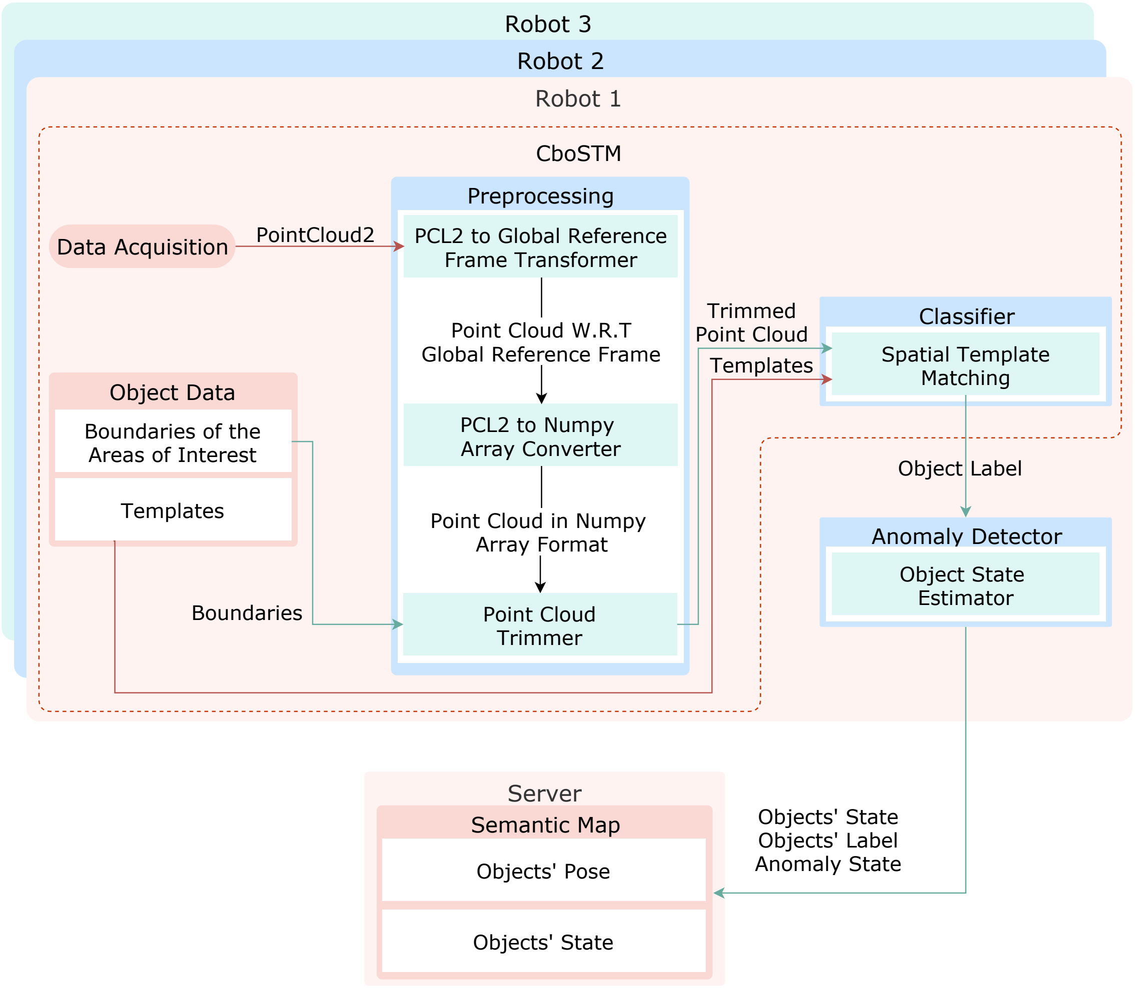


Figure 4.1: Diagram of the distributed architecture which has at its core the *Classifier based on Spacial Template Matching*.

The distributed architecture consists of two main modules. The *CboSTM* is composed of a set of submodules that, as a whole, are able to infer the label of the object that is present in a three-dimensional point clouds. The *Anomaly Detector* checks whether the inferred class represents an anomaly. Last but not least, the *Semantic Map* stores all extracted information, such as the status and pose of objects and detected anomalies.

4.1.1 Classifier based on Spatial Template Matching

The *Classifier based on Spatial Template Matching* is the core of the distributed architecture. Essentially, *CboSTM* consists of three main submodules. The *Preprocessing* has the purpose of processing the three-dimensional point clouds according to the *Classifier* requirements. The *Classifier* receives the preprocessed point cloud and infers the object label. The *Object Data*, has the purpose of storing the data related to the objects of interest, such as templates and the boundaries of the areas of interest, which are known at the outset.

4.1.1.1 Preprocessing

The object classification process begins when the patrolling robot is in a position close to an object of interest. The robot positions itself in an appropriate pose to inspect the object and, using the RGB-D camera, starts collecting data, three-dimensional point clouds without colour information.

Fig. 4.2 presents an image of the type of information collected by the patrolling robot, a raw three-dimensional point cloud. In it, it is possible to perceive a fire extinguisher, which corresponds to an object of interest, and a box, which is not an object of interest.

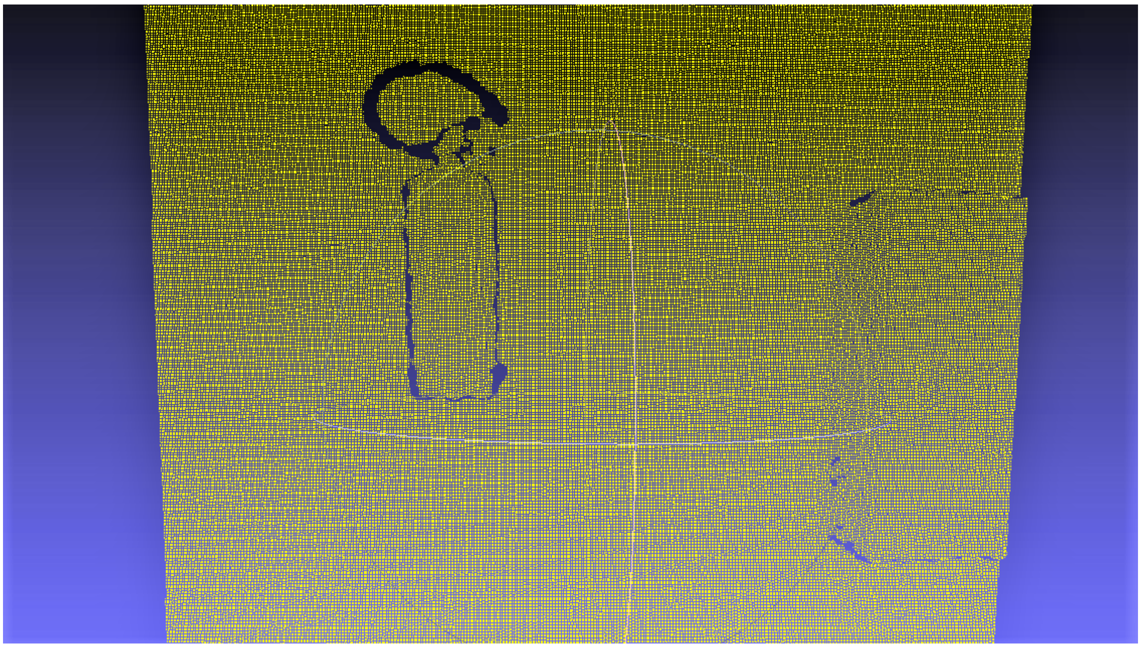


Figure 4.2: Example of raw point cloud captured by the patrolling robots.

After being collected by the RGB-D camera, the raw three-dimensional point cloud goes through the preprocessing stage, as represented in the diagram of Fig. 4.1, being the input of the *PCL2 to Global Reference Frame Transformer* submodule. The input point cloud is in PCL2 message type and spatial reference w.r.t the RGB-D sensor base link. The first stage of the *Preprocessing* module transforms the input point cloud from its RGB-D sensor base link, {S}, to the global reference frame, {W}. Since the robots' position on the map is known, as well as the transformation between the camera link and the robot base link, the point cloud registration w.r.t. the global reference frame is straightforward.

Equation 4.1 represents the transformation, per point, of the point cloud from RGB-D sensor base link to the global reference frame.

$$^{W}T_{P} =^{W} T_{R} *^{R} T_{S} *^{S} T_{P} \tag{4.1}$$

In order to be easier to manipulate the point cloud data attached to PointCloud2 message, the second stage of the preprocessing extracts all the points contained in the PointCloud2 message and saves them in a *Numpy Array*.

By taking advantage of the knowledge of the location of the objects of interest, it is possible to trim point clouds and to discard the points that are irrelevant to inferring the state of objects. Therefore, the remaining points only contain the information that is indispensable for inferring the state of the object to be inspected. The selection of relevant points is conducted based on the information provided by the user, which defines a set of boundaries delimiting the areas of interest. Using the limits defined by the user, the point cloud is trimmed, discarding all the points outside of the boundaries and keeping only the points considered relevant to assess the objects' position and orientation.

Fig. 4.3 presents an image of the point cloud after it has been preprocessed. As it is possible to observe, all the points that do not add relevant information to inferring the state of the object were trimmed.

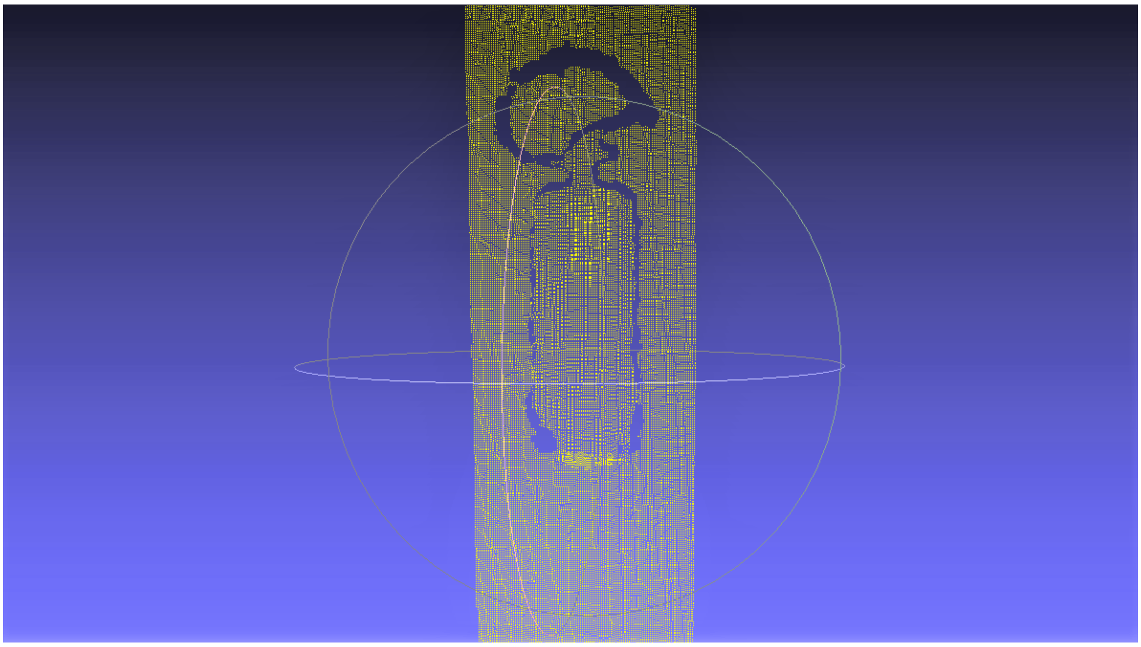


Figure 4.3: Fire extinguisher point cloud after being preprocessed.

4.1.1.2 Classifier

After preprocessed, the trimmed point cloud is transferred to the *Classifier* module, which has the purpose of recognising the object that is present in the point cloud.

The classifier is based on the analysis of the spatial arrangement of the point cloud elements, i.e. points. It is composed of three distinct modules, which are equally crucial for the system correct operation.

The diagram of Fig. 4.4 presents, in detail, the architecture behind the classifier used in this section. As it is possible to observe in this figure, the classifier receives as input a properly trimmed point cloud, which is in a *Numpy Array* with three columns and an undefined number of rows. Each row represents one point of the point cloud. The three columns correspond to the coordinates (X, Y, Z) in which the points are located.

4.1.1.3 Build 3D Voxel Grid

To easily comprehend the idea behind the *CboSTM* it is necessary to imagine a point cloud overlapped on a voxel grid. The first submodule of the *Classifier* creates a three-dimensional voxel grid according to the dimensions of the trimmed point cloud. Fig. 4.5 depicts a preprocessed point cloud, the input of the *CboSTM*, in which is possible to

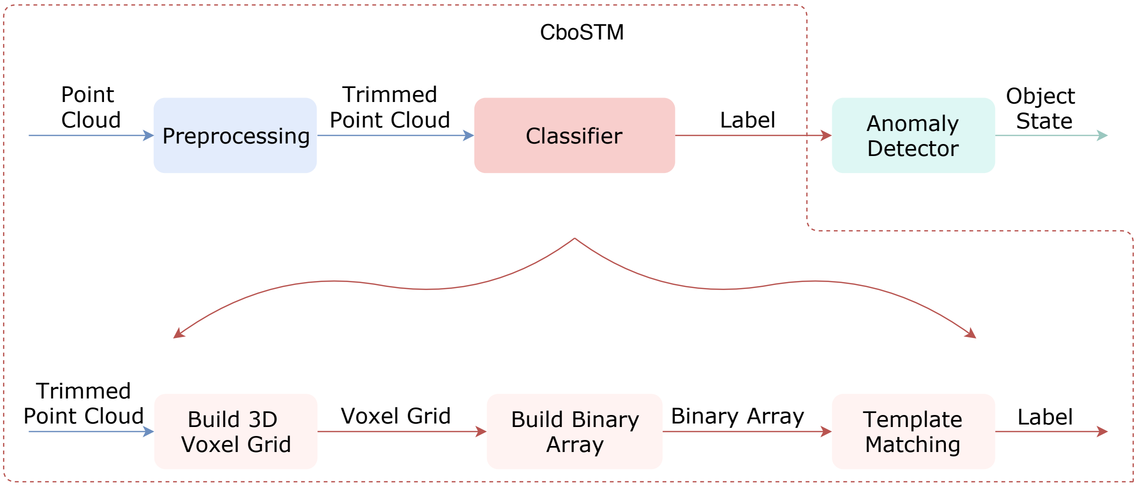


Figure 4.4: Diagram of *Classifier based on Spatial Template Matching*.

recognise a chair. On the right, Fig. 4.6, which is merely illustrative, depicts a three-dimensional voxel grid. The voxel grid parameters, number of voxels per axis and size of the voxels, are automatically adjusted according to the dimensions of the preprocessed point cloud and the parameters defined by the user.

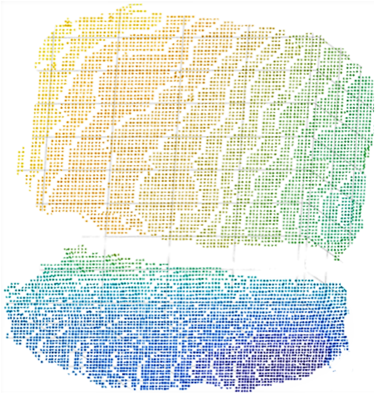


Figure 4.5: Point cloud of a chair trimmed by the preprocessing module.

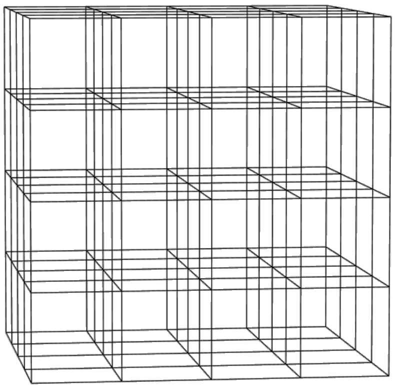


Figure 4.6: Empty voxel grid created according to the dimensions of the preprocessed point cloud.

Figure 4.7 intends to illustrate the method above described, presenting a 3D voxel grid with points inside that belongs to an area where it is possible do recognise a chair.

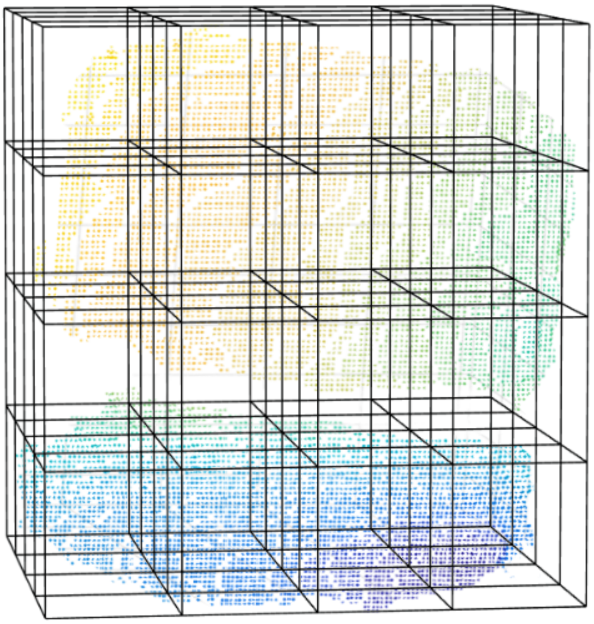


Figure 4.7: Voxel grid filled with a chair point cloud.

4.1.1.4 Build Binary Array

The second module of the classifier, *Build Binary Array*, receives as input the three-dimensional voxel grid containing points inside. The voxel status, i.e. either filled with points or empty, is the information that the classifier parses in order to infer the label of object present in the point cloud. In order for the method to be simpler and computationally lighter to parse the information, voxels are arranged by indexes and the voxels' status information of the occupation grid is stored in an indexed binary one-dimensional array. Each element of the binary array corresponds to the voxel state. If the voxel contains points, the element contains the value 1. Otherwise, in case the voxel does not contain any point inside, the element contains the value 0. Therefore, the generated binary array is no more than a template, a signature, of the characteristic spatial arrangement of the points, which is different for each class of objects. Fig. 4.8 presents, in an illustrative way, the method used to store occupation grid data.

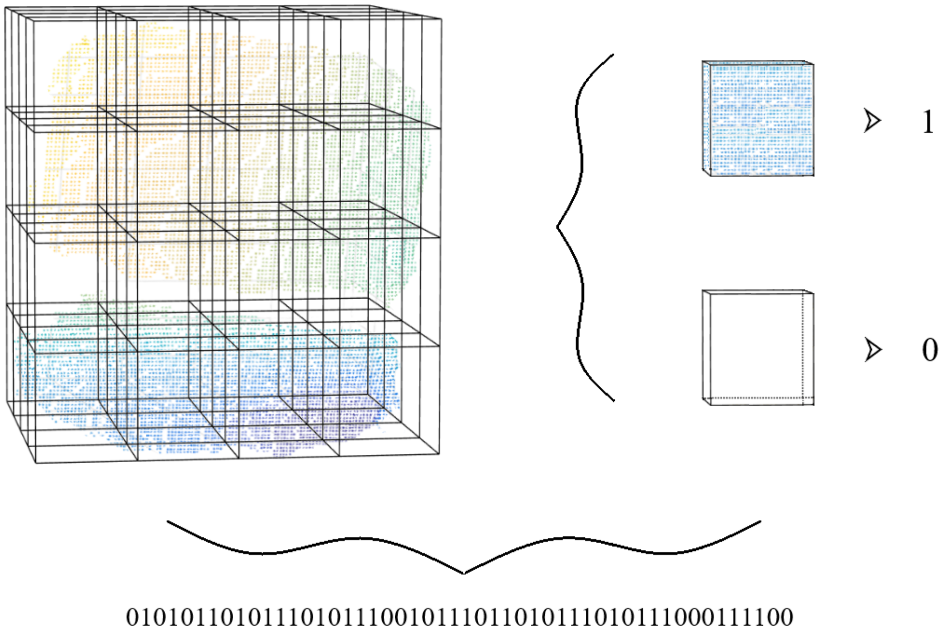


Figure 4.8: Occupation grid storing information method.

4.1.1.5 Template Matching

The last module of the classifier, *Template Matching*, is intended to compare the template created by the previous module in the classifier pipeline, *Build Binary Array*, with the templates of different objects and parse the similarities between them. The classifier infers the point cloud label according to the template label with the most matches.

Table 4.1 helps to understand how the label is selected by the classifier. In this table, the template matching results are presented for the following template: 0111000111. The table is just representative, in real scenario, each template has, at least, 250 elements.

Binary Array:	0111000111	
Label	Template	Number of Matches
Window_Open_45	0101100000	5
Window_Open_25	1010011001	3
Laptop	0111001111	9

Table 4.1: Example of how the template is selected using template matching.

As the laptop is the label with the higher number of matches, the classifier outputs that the parsed template corresponds to the laptop label with 90% of certainty.

4.1.1.6 Object State Estimator

The *Object State Estimator* module receives the output from the classifier and, taking into account the user-defined parameters, is able to assess whether the inferred label represents an anomaly in the system. Information about the objects' pose and its state are then stored in the semantic map of the environment.

4.1.1.7 Template Recording

The capture of the templates of each object works in a similar way to the system architecture presented. However, instead of being the classifier to infer the label, it is the user who manually introduces the label corresponding to the template.

4.2 Centralised vs Distributed Architecture

The distributed and centralised architectures developed throughout this dissertation are quite different, each having strengths and weaknesses. The main differences between the two architectures are presented below.

- The possibility of the system operating in a distributed way allows to increase the number of patrolling robots without having to increase the number of server computers to process the information coming from the robots. Therefore, it is possible to reduce the hardware and electricity costs. It also increases significantly the system scalability, allowing larger patrol robot teams.

- The distributed system allows robots to continue patrolling and to perform surveillance and inspection tasks even when they temporarily lose the wireless connection to the server, making the system more resilient to network failures;

- Unlike the centralised architecture, the distributed architecture does not rely on deep learning techniques, the patrolling robots have the means for processing the information acquired by themselves, avoiding the need to transfer large amounts of data to the server in order to be processed. The only data that needs to be transmitted throughout the network is the information regarding objects' states. Therefore, it is possible to use a connection with lower bandwidth or to increase the number of robots without the necessity to improve the wireless network;

- The object classification technique based on spatial template matching implemented in distributed architecture is capable of executing using only a small percentage of the robots' CPU. The processing power used by this technique is equivalent to the amount of processing power used by the function that comprises colour and depth images. Therefore, this architecture does not require greater use of the robots' CPU;
- The object classification technique based on spatial template matching implemented in distributed architecture does not use colour information to classify objects. This characteristic makes the system much more resilient to the low light conditions present in the test facilities, as well as to the illumination changes during night and day;
- The centralised architecture has the advantage of being more versatile and does not depend on information provided at the outset, such as the location of objects of interest.

4.3 Summary

In this chapter, the distributed architecture and all the modules that compose it were presented in detail. The basis of the distributed architecture, *Classification based on Spatial Template Matching*, and the theory behind the operation of the technique are presented. It was also presented a comparison between the distributed architecture and the centralised architecture, highlighting their differences and the advantages of each one of the architectures.

Chapter 5 describes the experiences carried out in order to experimentally validate the performance of the system.

5

Experimental Validation

Similar to every other classification method, in order to validate and evaluate the performance of the developed system, it needs to be tested. Hence, three very distinct tests were conducted. The first test consisted of a long-term experiment in which the system ran for a total of one hundred hours. The second test consisted of a benchmark between the classifier developed throughout this dissertation work, *Classifier based on Spatial Template Matching*, and a state-of-the-art object classification technique based on point clouds, *PointNet Classification*. During the 100h test, due to the inherent characteristics of the analysed objects, it was not possible to validate the system functionality to infer the objects' complete pose. Therefore, the last test validates the system's functioning for the detection of objects' complete pose.

5.1 One Hundred Hour Test

System's tests were inlaid in the test period of the STOP project, which was conducted in a real-world environment, at the 3rd floor of the "Centro Tecnológico da Cerâmica e do Vidro (CTCV)" facilities, between 15th and 19th July, totalling a testing period of one hundred hours, continuously.

5.1.1 Methodology

During one hundred hours, a team of three mobile patrolling robots and one server computer performed surveillance, inspection and patrolling tasks in a large pre-selected indoor environment. In this area there were six crucial objects to the security of the facility: three fire extinguishers, two doors, and one window. The object classification system ran for a total of three hundred hours, one hundred hours per patrolling robot, non-stop, without any critical flaw.

Due to previous knowledge of the test environment structure, some assumptions were made. It is known, at the outset, the location of objects of interest. Consequently, the robots only need to verify the areas of interest, such as the areas where fire extinguishers should be present. This knowledge limits the range of classes among which the object classification system has to distinguish, e.g. when a robot is inspecting a place where a window can be found, there are only two possible outcomes: the window is open or the window is closed. The same logic applies to other objects of interest.

One of the features of the developed system is the possibility of this being tuned while it is in operation. Therefore, during the one hundred hours test, the system underwent several adjustments. At the beginning of the test, all three robots had the same labels' templates, which were created by robot number two. While the robots were patrolling, new templates were recorded by each of the three mobile robots. Therefore, the new labels' templates were composed of the templates recorded by each of the robots and the initial templates that were recorded by robot number two. By the test's end, all three mobile robots were using only the templates recorded by themselves. The constant templates' modifications allow us to understand how the system behaves in different circumstances.

To successfully validate the functionality of the developed system, it must be capable of assessing, with high accuracy, the state of the three different classes of objects aforementioned. The system's validation requirements, according to the different objects, are mentioned below:

- Fire extinguisher: When a mobile robot is inspecting the position where a fire extinguisher is supposed to be, it should be capable to infer the state of the fire extinguisher, assessing if it is present, missing or obstructed. If the fire extinguisher is recognised, it means that the fire extinguisher is present. If instead of a fire extinguisher, the system recognises a wall, it means that the fire extinguisher is missing. In case the robot doesn't recognise neither a wall or a fire extinguisher, it means that there is an unrecognised object in the position where the fire extinguisher is supposed to be and the system reports that the fire extinguisher is obstructed.
- Doors: When a mobile robot is inspecting the position where a relevant door is, the

system should be capable to infer the state of it, assessing if the door is open or closed. In case the door is open, the system should also be capable to estimate, with high accuracy and a resolution of 22.5°, how open the door is.

- Windows: The large window, (Fig. 2.2c), present in the test environment is divided into two smaller windows. When the mobile robot is inspecting the window's state, the system should infer both windows' orientation, assessing if they are closed or open.

From the analysis of the classifier predictions and true conditions of the test, several confusion matrices were built. Those matrices contain enough information to calculate a variety of performance metrics, which allows for a richer performance analysis of the developed system. In order to evaluate the system predictions, four evaluation metrics were used. The applied metrics were: Accuracy, Precision, Recall and F1-Score.

When robots are inspecting fire extinguishers and doors, the object recognition system has to distinguish among more than two classes. Therefore, in order to evaluate the system's performance for those objects, a multi-class classification was used. Unlike doors and extinguishers, in the case of windows, the system only needs to distinguish between two states: open window or closed window. Therefore, in this case, a binary classification system was used.

The accuracy formula for multi-class classification is defined in equation 5.1.

$$Accuracy = \frac{\sum Correct\ Predictions}{\sum Total\ population} \tag{5.1}$$

The accuracy formula for binary classification is defined in equation 5.2.

$$Accuracy = \frac{\sum True\ Positive\ + \sum True\ Negative}{\sum Total\ Number\ of\ Examples} \tag{5.2}$$

The precision formula is defined in equation 5.3.

$$Precision = \frac{\sum True\ Positives}{\sum True\ Positives + \sum False\ Positives} \tag{5.3}$$

The recall formula is defined in equation 5.4.

$$Recall = \frac{\sum True\ Positive}{\sum True\ Positive + False\ Negative} \tag{5.4}$$

The F1-Score formula is defined in equation 5.5.

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5.5}$$

The aforementioned evaluation metrics are based on the following concepts:

- True positive - The classification system correctly predict a class.
- True negative - The classification system correctly predict a different class.
- False positive - The classification system mistakenly predicted a class.
- False negative - The classification system mistakenly predicted a different class.

During the one hundred hour test, the three robots inspected a large number of objects, thus performing a large number of classifications. Thus, in order to simplify the evaluation of the system's performance, the data analysis followed the following structure:

- During the last twelve operating hours, all classifications made by the three mobile robots were evaluated.
- For the rest of the time, for every four hours of operation, one hour was evaluated.

5.1.2 Results

In the one hundred hours test it was impossible to have balanced conditions across all labels due to the fact that test was conducted in a real life environment. F1-score is a good metric to evaluate the performance of classifiers when data is unbalanced.

In order for the results to be easily understood, evaluated and explained, they are presented in the form of confusion matrices, tables of results and bar charts. Due to space limitations, bar charts can be found in the appendix. The following tables present the confusion matrices for each of the three distinct models. These matrices were built through the analysis of classifier predictions during the hundred hour test.

		Predicted Condition	
		Open	Closed
True Condition	Open	799	240
	Closed	194	1098

Table 5.1: Confusion matrix obtained through the analysis of the classifier predictions and true conditions of the windows' states, during the one hundred hour test.

		Predicted Condition		
		Present	Missing	Obstructed
True Condition	Present	4530	62	116
	Missing	13	1917	14
	Obstructed	4	1	70

Table 5.2: Confusion matrix obtained through the analysis of the classifier predictions and true conditions of the fire extinguishers' states, during the one hundred hours test.

		Predicted Condition				
		Closed	Open 22.5	Open 45	Open 67.5	Open 90
True Condition	Closed	1723	0	0	0	0
	Open 22.5	0	9	0	0	0
	Open 45	0	0	95	2	0
	Open 67.5	0	0	1	61	0
	Open 90	0	0	0	10	1076

Table 5.3: Confusion matrix obtained through the analysis of the classifier predictions and true conditions of the doors' states, during the one hundred hour test.

The following tables present the results according to Accuracy, Precision, Recall and F1-score metrics, per class, of the classifier predictions for all the three models, during the one hundred hours test.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Open	81,38	80,46	76,9	78,64
	Closed	81,38	82,06	84,98	83,49

Table 5.4: Results according to Accuracy, Precision, Recall and F1-score metrics, per class, of the classifier predictions for the window's model, during the one hundred hours test.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Present	97,1	99,62	96,22	97,89
	Missing	98,66	96,81	98,61	97,7
	Obstructed	97,99	35	93,33	50,9

Table 5.5: Results according to Accuracy, Precision, Recall and F1-score metrics, per class, of the classifier predictions for the extinguisher's model, during the one hundred hours test.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Closed	100,0	100,0	100,0	100,0
	Open 22.5	100,0	100,0	100,0	100,0
	Open 45	99,89	98,96	97,94	98,45
	Open 67.5	99,56	83,56	98,39	90,37
	Open 90	99,66	100,0	99,08	99,53

Table 5.6: Results according to Accuracy, Precision, Recall and F1-score metrics, per class, of the classifier predictions for the door model, during the one hundred hour test.

As referred before, there were some template modifications during the test. Due to high accuracy prediction rates related to doors during the first hours of the test, the template modifications occurred only on templates related to fire extinguishers and windows. Therefore, the analysis of the last twelve hours of the test provides a more accurate evaluation of the system's performance. The confusion matrices are presented below.

		Predicted Condition	
		Open	Closed
True Condition	Open	464	59
	Closed	60	289

Table 5.7: Confusion matrix obtained through the analysis of the classifier predictions and true conditions of the windows' states, during the last twelve hours of test.

		Predicted Condition		
		Present	Missing	Obstructed
True Condition	Present	1438	10	4
	Missing	6	1156	3
	Obstructed	0	0	10

Table 5.8: Confusion matrix obtained through the analysis of the classifier predictions and true conditions of the extinguishers' states, during the last twelve hours of test.

The following tables present the results according to Accuracy, Precision, Recall and F1-score metrics, per class, of the classifier predictions for all the three models, for the last twelve hours of the test.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Open	86,35	88,55	88,72	88,63
	Closed	86,35	83,05	82,81	82,92

Table 5.9: Results according to Accuracy, Precision, Recall and F1-score metrics, per class, of the classifier predictions for the window's model, during the last twelve hours test.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Present	99,24	99,58	99,04	99,31
	Missing	99,16	98,89	99,23	99,06
	Obstructed	99,62	58,82	76,92	66,67

Table 5.10: Results according to Accuracy, Precision, Recall and F1-score metrics, per class, of the classifier predictions for the extinguisher's model, during the twelve hours test.

The following table presents the global results, per model, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics for the window, extinguisher and door models.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1- Score (%)
Model	Window	86.35	85.79	85.77	85.80
	Extinguisher	99.12	85.85	99.42	92.13
	Door	99.5	96.52	99.08	97.78

Table 5.11: Global results, per model, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics for the window, extinguisher and door models.

Tabel 5.11 summarises the performance of the classifier during the one hundred hour test.

5.1.3 Discussion

Analysing the results presented in tables 5.4 and 5.5, which contain the results according to the Accuracy, Precision, Recall and F1-score metrics, per class, of the classifier predictions for the window and extinguisher models, during the one hundred hours test, and comparing them against the results presented in tables 5.9 and 5.10, it is possible to verify that the modifications made on templates during the test contributed to the slight improvement of the systems results. Thus, it is concluded that capturing the templates with each of the robots increases the classifier performance.

Although the robots are very similar, there are minor differences between them, which means that the transformation between the RGB-D camera sensor and the robot base link is different for each one of the three patrolling robots. An adjustment on the robots' transforms would avoid the necessity to record templates for each of the robots. However, contrary to the modifications made on templates, robots' TFs adjustments cannot be executed while the system is running.

Table 5.11 illustrates the global results, per model, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics for the window, extinguisher and door models, during the last twelve hours of test. As it is possible to observe, the system's performance varies according to the different models. In the case of the window model, the system is capable of inferring the state of the window with high accuracy, recall, precision, and f1-score. The graph also demonstrates that the classifier has a better performance for the extinguisher model when compared to the window model. In this case, evaluation results prove that the system is capable of inferring the state of fire extinguishers with very high accuracy and recall. The evaluations results, according to precision and F1-score, are lower but still very respectful. In the case of the door model, the classifier is able to infer, with a high level across all four evaluation metrics used, the doors' state, being able to assess the orientation in which the door is. The results shown in the bar chart of figure 6.2 can be consulted in detail in table 5.11.

Analysing the table 5.10, it becomes clear that the class with the worst results is the obstructed class. The inferring of this class is executed by an elimination process. It is known, at the outset, that when a robot is inspecting the position where a fire extinguisher is supposed to be, there are only three possible outcomes: the fire extinguisher is detected by the classifier and it is considered to be present; the classifier detects a wall and considers that the extinguisher is missing; the classifier does not detect a fire extinguisher or a wall and, therefore, considers that the fire extinguisher is obstructed. The elimination process used to infer if the fire extinguisher is obstructed is far from being perfect, however, this technique ensures that the system is able to infer if the fire extinguisher is being clogged without knowing exactly which object is obstructing it.

Analysing the state of a window using only depth information is a complex challenge due to the fact that it is impossible to extract the depth information for the glass of the window. The developed classifier infers the state of the windows based only on the depth information of the window's structure. The results obtained by the classifier for the model of the windows, which can be consulted in the table 5.9, although acceptable, are not perfect. These results can be improved with a different approach to the problem. Instead of inspecting both windows at the same time, the system can parse one small window at a time. In this way it is possible to increase the useful information from each window and to avoid capturing irrelevant points such as the points that are in the middle of the two small windows.

Given the results obtained during the long and exhaustive test, in which the object classification system correctly operated for one hundred hours on each of the patrolling robots, totalling three hundred hours of operation without any critical failure and with high performance on the four evaluation metrics used, I believe that the expectations and objectives have been successfully met.

5.2 Benchmarking of the Developed Technique

A benchmark based on real patrolling situations occurred in the test environment was planned to assess the performance of the developed technique, regarding the latest state-of-the-art methods for object classification.

5.2.1 Methodology

Due to similarities between the heuristic technique developed throughout this dissertation and the PointNet Classification method, a state-of-the-art method for object classification based on deep learning, was the chosen technique to perform the benchmarking against the developed heuristic method. Both are based on point clouds, having the same type of input - a point without colour, and they also share the same type of output - a label per point cloud.

In oposition to the one hundred hour test, in which the robot freely patrolled through the environment and randomly inspected the objects of interest, the benchmarking test was carried out in a controlled environment. In order to attain a fair benchmarking, the techniques shared both the training and testing dataset. The datasets were composed of three models: window, extinguisher and door. Due to limitations of the *Pointnet Classification* system, the obstructed class was not evaluated in the fire extinguisher model. The training dataset contained ten distinct point clouds per class while the testing dataset contained twenty distinct point clouds per class. Having exactly the same number of point clouds per class makes the test dataset more homogeneous, allowing for an accurate system benchmark.

The datasets used in the benchmarking were captured for four hours, at the CTCV test environment, through the RGB-D camera of one of the patrolling robots. In training and testing, PointNet Classification settings were at default. The developed technique was also in its default parameters, with a 7x5x10 voxel grid.

The evaluation of the performance of the two techniques was performed using the four evaluation metrics, accuracy, precision, recall and F1-Score, mentioned in section 5.1.1.

5.2.2 Results

In order for the results to be easily understood, evaluated and explained, these are presented in the form of confusion matrices, tables of results and bar charts. Due to space limitations, bar charts can be found in the appendix. The following tables present confusion matrix obtained through the analysis of the *CboTM* classifier predictions and true conditions of the objects' states for the benchmarking test dataset.

		Predicted Condition	
		Open	Closed
True Condition	Open	17	3
	Closed	1	19

(a) Confusion matrix obtained through the analysis of the CboTM classifier predictions and true conditions of the windows' states for the benchmarking test dataset.

		Predicted Condition	
		Present	Missing
True Condition	Present	20	0
	Missing	0	20

(b) Confusion matrix obtained through the analysis of the CboTM classifier predictions and true conditions of the extinguishers' states for the benchmarking test dataset.

		Predicted Condition				
		Closed	Open 22.5	Open 45	Open 67.5	Open 90
True Condition	Closed	20	0	0	0	0
	Open 22.5	0	19	1	0	0
	Open 45	0	0	20	0	0
	Open 67.5	0	0	0	20	0
	Open 90	0	0	0	0	20

(c) Confusion matrix obtained through the analysis of the CboTM classifier predictions and true conditions of the doors' states for the benchmarking test dataset.

Table 5.12: Confusion matrices obtained through the analysis of the CboTM classifier predictions and true conditions of objects states for the benchmarking test dataset.

The following tables present the results, per label, according to Accuracy, Precision, Recall and, F1-score metrics for *CboTM* classifier predictions for benchmarking test dataset.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Open	90	94,4	85,00	89,45
	Closed	90	86,36	95,0	90,47

(a) Results, per label, according to Accuracy, Precision, Recall and F1-score metrics of the CboTM classifier predictions for the window model of the benchmarking test dataset.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Present	100,0	100,0	100,0	100,0
	Missing	100,0	100,0	100,0	100,0

(b) Results, per label, according to Accuracy, Precision, Recall and, F1-score metrics of the CboTM classifier predictions for the extinguisher model of the benchmarking test dataset.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Closed	100,00	100,00	100,0	100,0
	Open 22.5	99,00	100,00	95,00	97,4
	Open 45	99,00	95,24	100,00	97,6
	Open 67.5	100,00	100,00	100,00	100,0
	Open 90	100,00	100,00	100,00	100,0

(c) Results, per label, according to Accuracy, Precision, Recall and, F1-score metrics of the CboTM classifier predictions for the door model of the benchmarking test dataset.

Table 5.13: Results, per label, according to Accuracy, Precision, Recall and, F1-score evaluation metrics of the CboTM classifier predictions for the benchmarking test dataset.

The following tables present confusion matrix obtained through the analysis of the *PointNet Classification* predictions and true conditions of the objects' states for the benchmarking test dataset.

		Predicted Condition	
		Open	Closed
True Condition	Open	4	16
	Closed	0	20

(a) Confusion matrix obtained through the analysis of the PointNet classifier predictions and true conditions of the windows states for the benchmarking test dataset.

		Predicted Condition	
		Present	Missing
True Condition	Present	20	0
	Missing	0	20

(b) Confusion matrix obtained through the analysis of the PointNet classifier predictions and true conditions of the extinguishers states for the benchmarking test dataset.

		Predicted Condition				
		Closed	Open 22.5	Open 45	Open 67.5	Open 90
True Condition	Closed	20	0	0	0	0
	Open 22.5	0	2	0	10	8
	Open 45	0	3	0	10	7
	Open 67.5	0	10	0	7	3
	Open 90	16	0	0	2	2

(c) Confusion matrix obtained through the analysis of the PointNet classifier predictions and true conditions of the doors states for the benchmarking test dataset.

Table 5.14: Confusion matrices obtained through the analysis of the PointNet classifier predictions and true conditions of objects states for the benchmarking test dataset.

The following tables present the results, per label, according to Accuracy, Precision, Recall and, F1-score metrics for *PointNet Classification* predictions for benchmarking test dataset.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Open	60,00	100,0	20,00	33,33
	Closed	60,00	55,56	100,0	71,42

(a) Results, per label, according to Accuracy, Precision, Recall and F1-score metrics of the PointNet classifier predictions for the window model of the benchmarking test dataset.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Present	100,0	100,0	100,0	100,0
	Missing	100,0	100,0	100,0	100,0

(b) Results, per label, according to Accuracy, Precision, Recall and F1-score metrics of the PointNet classifier predictions for the extinguisher model of the benchmarking test dataset.

		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Label	Closed	84,00	55,56	100,0	71,4
	Open 22.5	69,00	13,33	10,00	11,4
	Open 45	80,00	0,00	0,00	0,00
	Open 67.5	65,00	24,14	35,00	28,57
	Open 90	64,00	10,00	10,00	10,00

(c) Results, per label, according to Accuracy, Precision, Recall and F1-score metrics of the PointNet classifier predictions for the door model of the benchmarking test dataset.

Table 5.15: Results, per label, according to Accuracy, Precision, Recall and F1-score evaluation metrics for PointNet classifier predictions for the benchmarking test dataset.

The following tables present the results, per model, according to Accuracy, Precision, Recall and F1-score evaluation metrics of the *CboTM* and *PointNet Classification* predictions for the benchmarking test dataset.

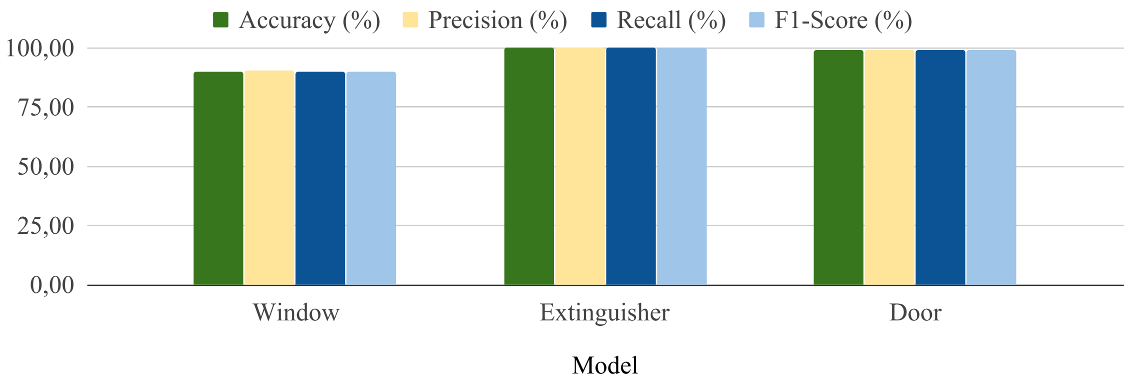
		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Model	Window	90,00	90,40	90,00	89,96
	Extinguisher	100,0	100,0	100,0	100,0
	Door	99,00	99,05	99,00	99,00

(a) Results, per model, according to Accuracy, Precision, Recall and F1-score evaluation metrics of the CboTM classifier predictions for the benchmarking test dataset.

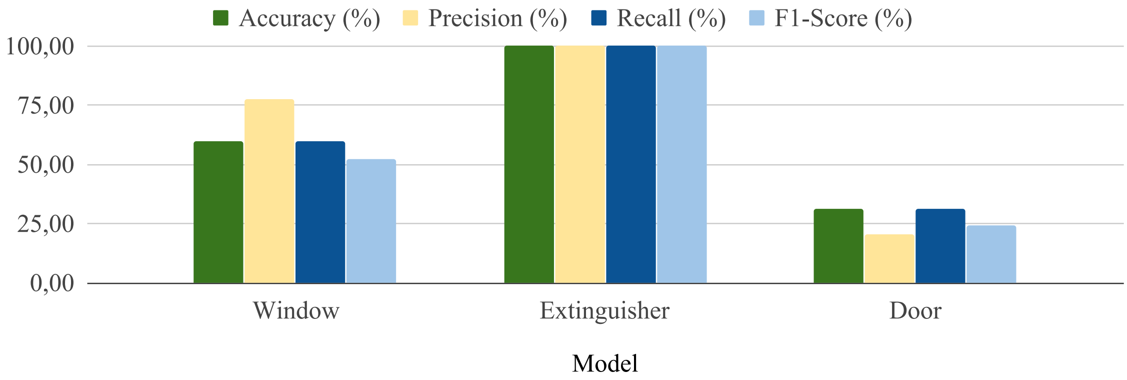
		Evaluation Metric			
		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Model	Window	60,00	77,78	60,00	52,38
	Extinguisher	100,0	100,0	100,0	100,0
	Door	31,00	20,61	31,00	24,29

(b) Results, per model, according to Accuracy, Precision, Recall and F1-score evaluation metrics of the PointNet classifier predictions for the benchmarking test dataset.

Table 5.16: Results, per model, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the CboTM and PointNet classifier predictions for the benchmarking test dataset.



(a) Representative bar chart of the results, per model, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the CboTM classifier predictions for the benchmarking test dataset.



(b) Representative bar chart of the results, per model, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the PointNet classifier predictions for the benchmarking test dataset.

Figure 5.6: Representative bar charts of the results, per model, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the CboTM and PointNet classifier predictions for the benchmarking test dataset.

5.2.3 Discussion

Analysing tables 5.2a, 5.2b and 5.2c, which present the results, per label, according to Accuracy, Precision, Recall and, F1-score evaluation metrics of the *CboSTM* classifier predictions for the window, extinguisher and door model, respectively, of the benchmarking test dataset, it is possible to infer that the *CboSTM* kept, approximately, the same level of results presented in the one hundred hours, as expected.

Tables 5.4a, 5.4b and 5.4c, present the results, per label, according to Accuracy, Precision, Recall and, F1-score evaluation metrics of the *PointNet Classification* predictions for the window, extinguisher and door model, respectively, of the benchmarking test dataset. It is possible to infer that the *PointNet Classification* for the fire extinguisher model are very good, however, the results for the window and door model are less impressive.

Regarding the comparison between *CboSTM* e o *PointNet Classification*, table 5.5a and 5.5b present the results per model, according to the Accuracy, Precision, Recall and F1-score evaluation metrics of the *CboSTM* and the *PointNet Classification* predictions, respectively, for the benchmarking test dataset. According to the results presented, it is possible to infer that CboSTM clearly outperforms PointNet Classification in the window and door model. In the fire extinguisher model, the classification results of CboSTM and PointNet Classification are good. Both techniques present the same result, 100% for accuracy, precision, recall and F1-Score.

Figures 5.6a and 5.6 present, respectively, bar charts of the results, per model, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the *CboSTM* and *PointNet Classification* predictions for the benchmarking test dataset.

In the overall picture, in this real environment comparison, *CboSTM* shows better results.

5.3 System validation for the estimation of objects' complete pose

In order to validate the functionality of the system for the estimation of the six degrees of freedom, a test with two distinct objects was planned. The objects chosen were a rectangular box and a chair.

It was found that the system correctly works for the rectangular box. It is capable of accurately estimating, with a resolution of 20°, the three degrees of freedom for the orientation of the box. The three degrees of freedom for the position of both objects are estimated by averaging the coordinates of the points that constitute the object.

For the estimation of the three degrees of freedom of orientation of the chair the results are less satisfactory. It was found that the point clouds obtained through the RGB-D sensor for the legs of the chair are of very poor quality, which impairs the functioning of the system. Even so, the system was able to estimate, with a resolution of 20°, the chair's orientation around the *roll*, *pitch* and *yaw* axis.

Although the results for the estimation of chair's complete pose are not very impressive, it is possible to confirm that the system is able to estimate the complete pose of objects.

6

Conclusion

This document starts by introducing the framework of this dissertation, presenting and explaining the main goals, the available resources, and requirements.

To accomplish the defined objectives, a centralised architecture for the estimation of objects' 6 DoF based on point clouds was developed. The development of this architecture consisted of the integration of several techniques, having in its core a method for semantic segmentation of point clouds, based on deep learning. In the preliminary tests, the system was found to be unreliable due to the very inaccurate semantic segmentation results.

In order to have a more reliable system, a distributed architecture based on point clouds was developed. The system based on a distributed architecture takes advantage of previous knowledge of the location of objects of interest and applies a classification method based on the analysis of three-dimensional point clouds using template matching. This system proved to be very accurate, providing very impressive object classification results without resorting to any kind of deep learning technique.

Chapter 5 presents the results for the experimental validation of the system, which consisted of three different tests. The first test was a long-term one, in which the system ran for one hundred hours without any critical flaw and with impressive performance. The second test consisted of a benchmarking between the developed classifier, *Classification based on Spatial Template Matching*, and a state-of-the-art object classification system based on deep learning techniques, *PointNet Classification*. The results of the benchmarking are intriguing as the *Classification based on Spatial Template Matching* method outperformed *PointNet Classification* in two of the three models and, in the third model, the performances were equal. The last test aimed to validate the functioning of the system for the estimation of the objects complete pose, which was successfully achieved.

6.1 Future Work

This section presents some improvements that could be introduced in both developed systems that, due to calendar-related constraints, were not implemented.

The centralised system was unable to achieve satisfactory results. It would be interesting to improve the results of it by integrating a more promising object detection and classification method, capable to operate in real environments and with poor light conditions.

The distributed system presented in this dissertation constitutes a solid basis, however, it has the potential to be upgraded. The implementation of an accurate and efficient point clouds clustering technique, such as the ones presented by Klasing [5] and Nakagawa [7], would be a major improvement as it has the potential to make the system more resilient and able to operate without any information of the environment where the robot is patrolling acquired at the start.

One of the skills with which the system could also be endowed is the ability to do a reconstruction in three dimensions of the recognised objects, integrating, for example, the work presented by Schreiberhuber [15].

It also would be interesting to evaluate the performance of the technique using a RGB-D sensor such as the Kinect V2, which produces point clouds with a much better quality than the Astra sensor used throughout this work.

Bibliography

[1] M. Ankerst, M. Breunig, H. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure. *SIGMOD Rec.*, 28(2):49–60, June 1999.

[2] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[3] R. B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015.

[4] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[5] K. Klasing, D. Wollherr, and M. Buss. A clustering method for efficient segmentation of 3d laser data. In *2008 IEEE International Conference on Robotics and Automation*, pages 4043–4048, May 2008.

[6] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. Technical Report arXiv:1109.2378, Sep 2011. Comments: 29 pages.

[7] Masafumi Nakagawa. *Point Cloud Clustering Using Panoramic Layered Range Image.* 08 2018.

[8] A. Ng, M. I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems,*, 14:849–856, 2001.

[9] Kiru Park, Timothy Patten, Johann Prankl, and Markus Vincze. Multi-task template matching for object detection, segmentation and pose estimation using depth images. 05 2019.

[10] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CoRR*, abs/1612.00593, 2016.

[11] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[12] Zheng Qin, Zeming Li, Zhaoning Zhang, Yiping Bao, Gang Yu, Yuxing Peng, and Jian Sun. Thundernet: Towards real-time generic object detection. *CoRR*, abs/1903.11752, 2019.

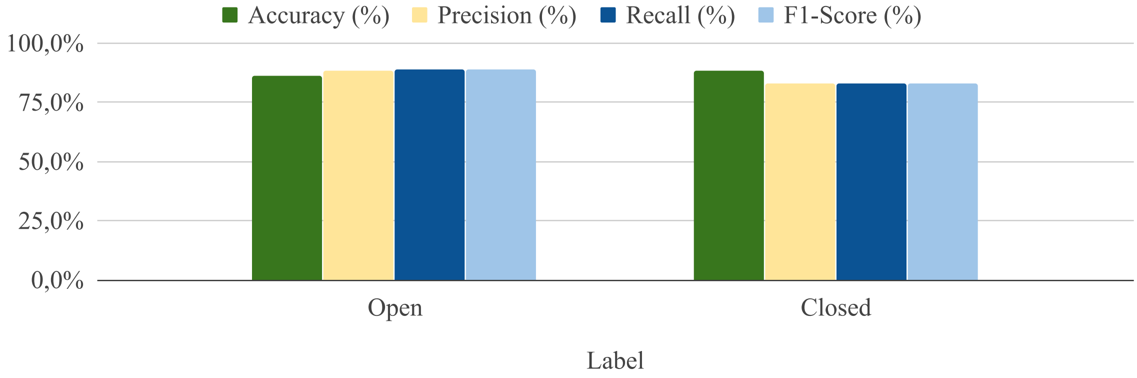
[13] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[14] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

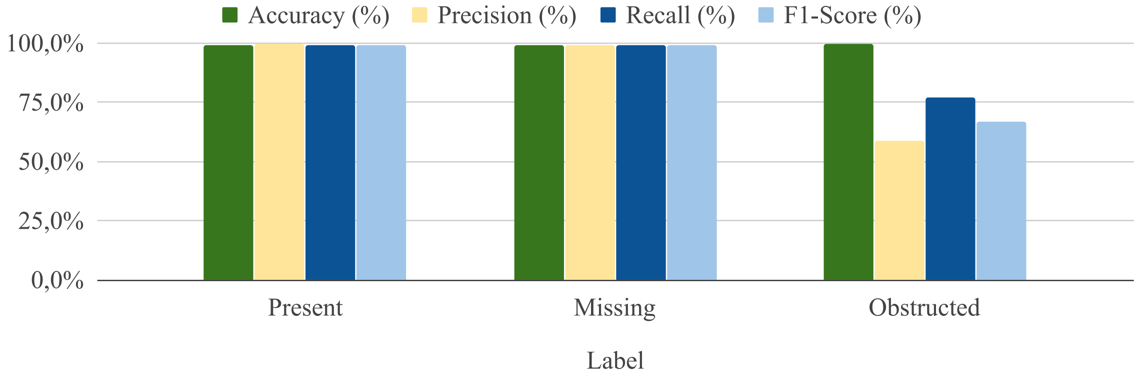
[15] S. Schreiberhuber, J. Prankl, T. Patten, and M. Vincze. Scalablefusion: High-resolution mesh-based real-time 3d reconstruction. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 140–146, May 2019.

Appendix

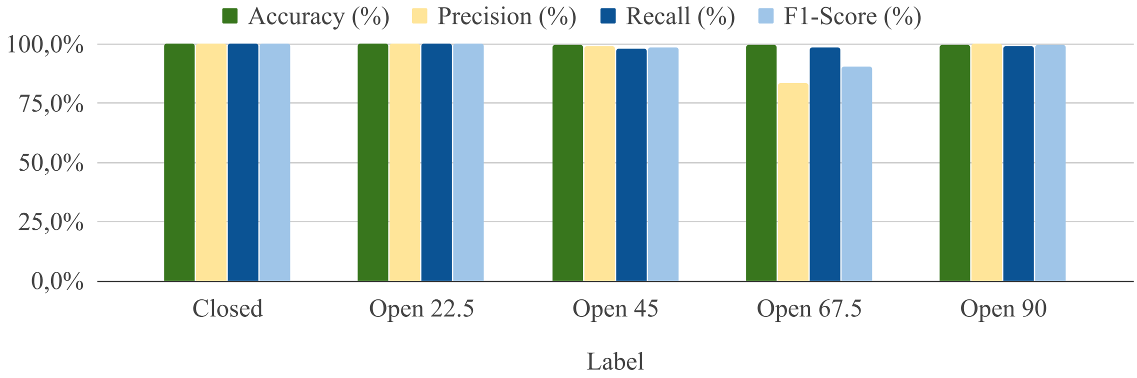
The following bar charts illustrate the classifier prediction results for the accuracy, precision, recall and F1-Score metrics, per class, throughout the last twelve hours of the one hundred hours test. In this period, the system worked in its perfect conditions, which improved the overall results obtained.



(a) Bar chart representative of the results, per class, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics for the window model.



(b) Bar chart representative of the results, per class, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics for the extinguisher model.



(c) Bar chart representative of the results, per class, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics for the door model.

Figure 6.1: Bar charts representative of the results, per label, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics for the window, extinguisher and door models, during the last twelve hours of test.

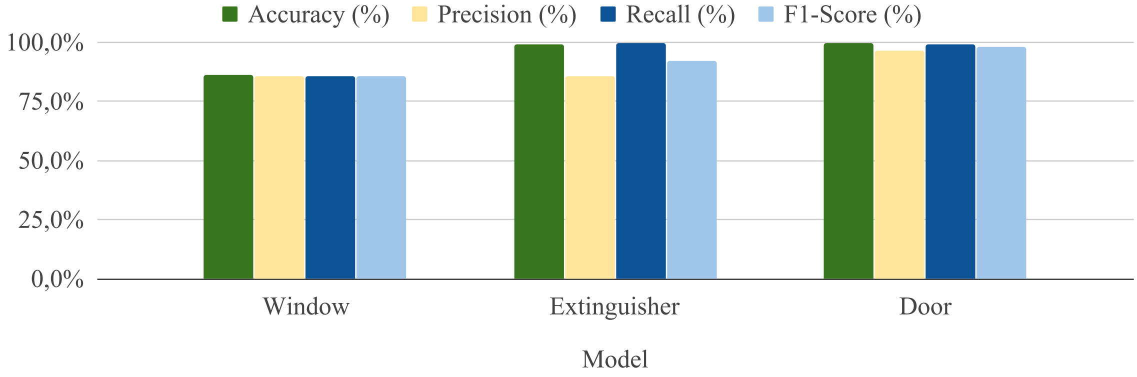
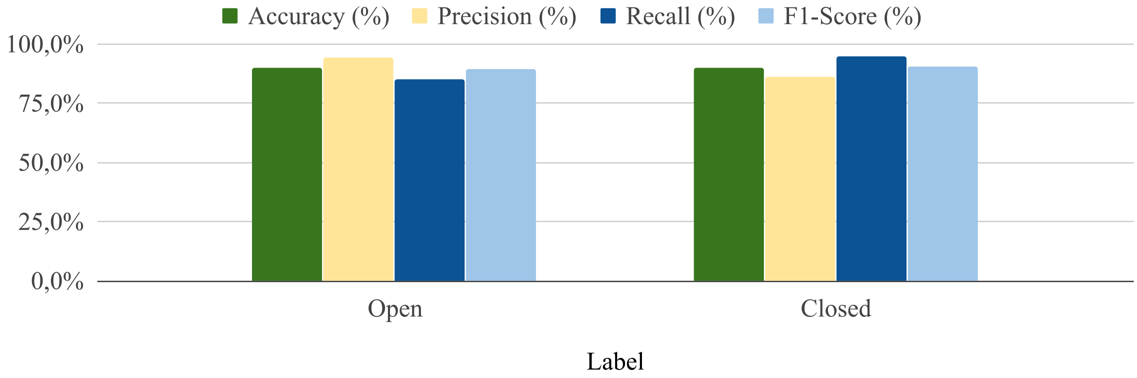
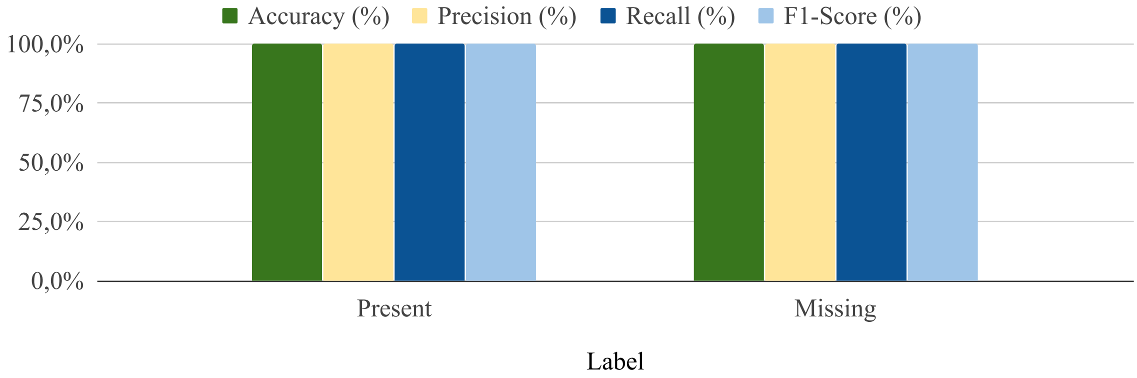


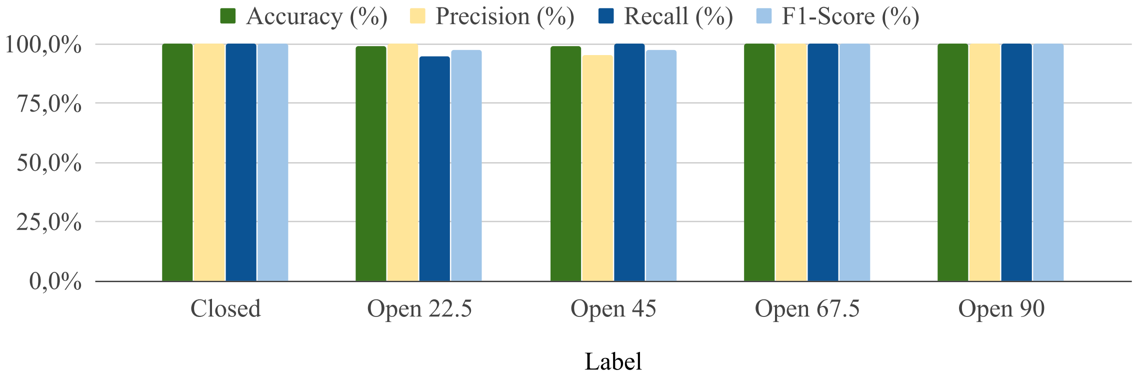
Figure 6.2: Bar charts representative of the global results, per model, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics for the window, extinguisher and door models, during the last twelve hours of test.



(a) Representative bar chart of the results, per label, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the CboTM classifier predictions for the window model of the benchmarking test dataset.

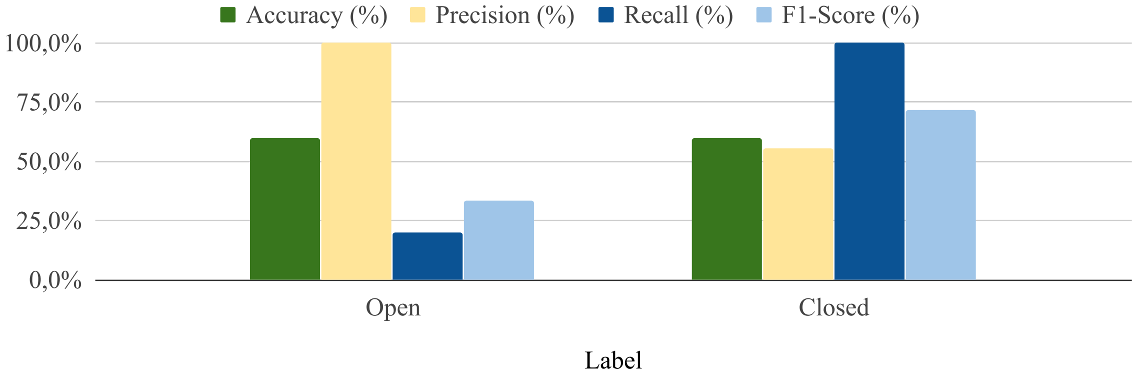


(b) Representative bar chart of the results, per label, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the CboTM classifier predictions for the extinguisher model of the benchmarking test dataset.

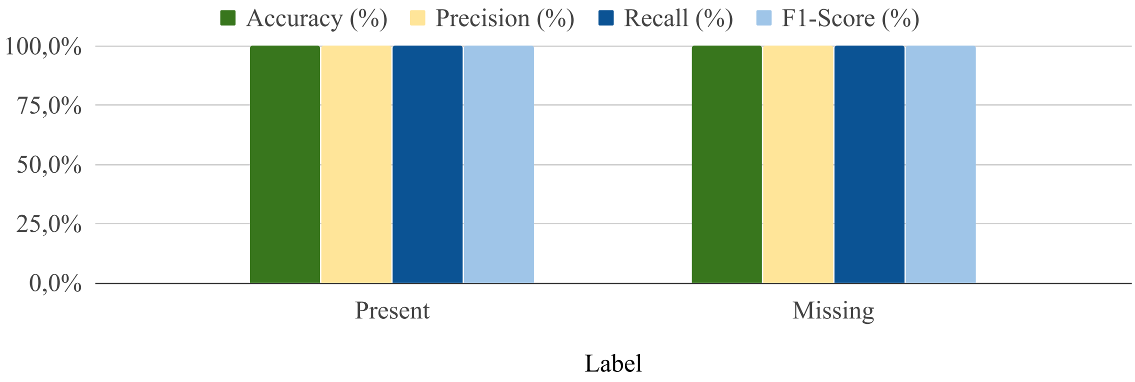


(c) Representative bar chart of the results, per label, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the CboTM classifier predictions for the door model of the benchmarking test dataset.

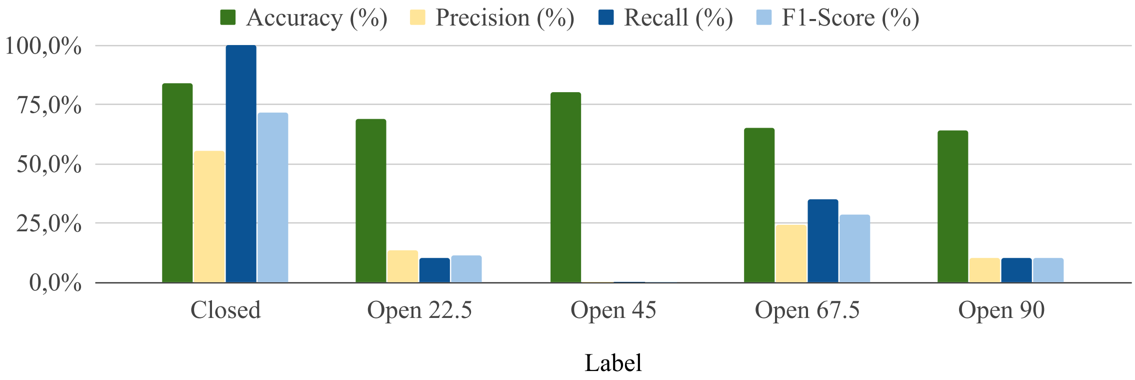
Figure 6.3: Representative bar charts of the results, per label, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the CboTM classifier predictions for the window, extinguisher and door models of the benchmarking test dataset.



(a) Representative bar chart of the results, per label, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the PointNet classifier predictions for the window model of the benchmarking test dataset.



(b) Representative bar chart of the results, per label, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the PointNet classifier predictions for the extinguisher model of the benchmarking test dataset.



(c) Representative bar chart of the results, per label, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the PointNet classifier predictions for the door model of the benchmarking test dataset.

Figure 6.4: Representative bar charts of the results, per label, according to the Accuracy, Precision, Recall and F1-Score evaluation metrics of the PointNet classifier predictions for the window, extinguisher and door models of the benchmarking test dataset.