

Sistema Inteligente de Prevenção de Colisões para Robô Móvel Autónomo em Estufas

JOÃO PEDRO PIRES JACOB CORREIA MARTINS
novembro de 2022

POLITÉCNICO DO PORTO
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

Smart Collision Avoidance System for Greenhouses Autonomous Mobile Robot

João Pedro Pires Jacob Correia Martins

Master in Electrical and Computer Engineering
Specialization Area of Autonomous Systems



DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA
Instituto Superior de Engenharia do Porto

November, 2022

*This dissertation partially satisfies the requirements of the
Thesis/Dissertation course of the program Master in Electrical and Computer
Engineering, Specialization Area of Autonomous Systems.*

Candidate: João Pedro Pires Jacob Correia Martins, No. 1171167,
1171167@isep.ipp.pt

Scientific Guidance: Manuel Fernando dos Santos Silva, mss@isep.ipp.pt

Scientific Co-Guidance: Filipe Neves dos Santos,
filipe.n.santos@inesctec.pt

Company: INESC TEC—Institute for Systems and Computer Engineering,
Technology and Science.



DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA
Instituto Superior de Engenharia do Porto
Rua Dr. António Bernardino de Almeida, 431, 4200-072 Porto

November, 2022

"Keep it simple... Dream, Fight, Conquer..."
– *Unknown*

Acknowledgements

I would first like to thank my supervisor and co-supervisor for guiding me through the entire process of developing this thesis.

I would also like to thank my colleague Humberto Rocha who conceptualize, design and developed the robotic platform together with me and, without which, this thesis would not be possible to develop.

My thanks to Velodyne for providing the prototype LiDAR Velarray M1600 B2.

And last but not least, to my family and girlfriend for providing me a good environment and supporting me throughout this study journey.

Abstract

Mobile robots are increasingly being used in protected and open field agriculture. However, it is necessary to maintain safety for the cooperating agents of the environment, such as persons.

Systems used in industry are prepared for a controlled environment with little dynamics. Agricultural environments are very dynamic with constant changes of the static structures and intervention of the agents of the environment, such as people and living beings. Thus, an intelligent system capable of predicting dangerous situations between the robot and its surroundings is needed.

This thesis addresses the development of a smart collision avoidance system, for robots operating in greenhouses, based on the 3D representation of the environment by fusion of RGB image with 3D point cloud, and object detection through deep learning.

Since agriculture is a dynamic environment, approaches that require prior knowledge of the robot's surroundings become unfeasible. This system allows the creation, in real-time, of a 3D map where objects known by the robot are detected. This way its operation is not conditioned when changes occur in the scene. The system acts in the robot's navigation according to its spacial sense, to static or moving objects, respecting the International Organization for Standardization (ISO) standards stipulated for Autonomous Mobile Robots (AMR).

To evaluate the system, is simulated the scenario with real data from the greenhouses and then rated collision avoidance outputs commands to the navigation system.

Keywords: Collision Avoidance, RGB Image, 3D Point Cloud, 3D Map, Deep Learning, Object Detection.

Resumo

Os robôs móveis estão a ser cada vez mais utilizados na agricultura protegida e de campo aberto. No entanto, é necessário manter a segurança dos agentes que co-operam no meio, tais como as pessoas.

Sistemas utilizados na indústria estão preparados para um ambiente controlado e com pouca dinâmica. Ambientes agrícolas revelam-se muito dinâmicos com constantes alterações das estruturas estáticas e intervenção dos agentes do meio, como pessoas e seres vivos. Assim, é necessário um sistema inteligente capaz de prever situações de perigo entre o robot e o meio que o rodeia.

Esta tese aborda o desenvolvimento de um sistema inteligente de prevenção de colisões, para robôs que operam em estufas, baseado na representação 3D do ambiente por fusão de imagem RGB com nuvem de pontos 3D, e consequente deteção de objetos através de aprendizagem profunda.

Uma vez que a agricultura apresenta um ambiente dinâmico, as abordagens que requerem um conhecimento prévio do ambiente do robô tornam-se impraticáveis. Este sistema permite criar em tempo real um mapa 3D onde objetos conhecidos pelo robô são detetados. Desta forma, o seu funcionamento não é condicionado quando ocorrem alterações no ambiente. O sistema atua na navegação do robô de acordo com o seu sentido espacial, tanto de objetos estáticos ou em movimento, respeitando as normas da organização internacional para a normalização (ISO) estipuladas para veículos móveis autónomos (AMR).

Para avaliar o sistema, simula-se o cenário com dados reais das estufas e depois classificam-se os comandos de saída, que previnem colisões, para o sistema de navegação.

Palavras-Chave: Prevenção de Colisões, Imagem RGB, Nuvem de Pontos 3D, Mapa 3D, Aprendizagem Profunda, Deteção de Objectos.

Contents

List of Figures	ix
List of Tables	xi
List of Symbols	xiii
1 Introduction	1
1.1 Background	1
1.2 Motivation	2
1.3 Goals	3
1.4 Work Scheduling	3
1.5 Structure	4
2 State of the Art	7
2.1 Human-robot Interaction in Agriculture Environments	7
2.1.1 Interaction	8
2.1.2 Obstacle avoidance	8
LiDAR and Camera Fusion	10
Deep learning	11
2.2 ISO's Standards	12
2.2.1 Industrial Robotics Standards	12
2.2.2 Migration to Agricultural Environment	16
2.3 Comparison between Collision Avoidance Methods and Standards . .	18
2.4 Remarks	19
3 Smart Collision Avoidance Requirements	21
3.1 Working Environment	21
3.2 System Requirements	23
4 Smart Collision Avoidance Architecture	27
4.1 Collision Avoidance Architecture	27
4.2 Sensor Fusion	29
4.2.1 Extrinsic and Intrinsic	29
4.2.2 RGB image to Point Cloud	32

4.3	Environment Projection to Collision MAP	33
4.4	Object Detection	34
4.5	Collision Avoidance	35
4.5.1	Collision Evaluation	35
4.5.2	State Machine	36
4.6	Remarks	37
5	Smart Collision Avoidance Module - Hardware and Implementa-	
	tion	39
5.1	Robot Platform	39
5.2	Perception Sensors	41
5.2.1	LiDARs	41
5.2.2	OAK-D Camera	41
5.3	Software Implementation	42
5.3.1	Camera Calibration	43
5.3.2	Transformations	45
5.4	LiDAR Point Cloud and RGB Camera Image Fusion	46
5.4.1	Sensors Data	46
	OAK-D RGB Camera	47
	Livox	47
	Velarray	47
5.4.2	Sensors Fusion	49
5.5	Deep Learning	54
5.6	Robot 3D Modeling	54
5.6.1	Model Definition	54
5.6.2	Model Kinematics	55
5.7	Collision Avoidance	56
5.7.1	3D Environment	57
5.7.2	Collision Map	58
5.7.3	Collision Detection	59
5.7.4	Collision Evaluation	62
5.8	Collision State Machine	64
5.9	Remarks	65
6	Results	67
6.1	Warehouse Navigation	67
6.2	Plant Aisle Navigation	68
6.2.1	Plants and Fruits	69
6.2.2	Other Obstacles	70
6.3	Persons Detection	72

7	Conclusions	75
7.1	Future Work	76
7.1.1	Sensor Fusion	76
7.1.2	Transformations	76
7.1.3	Deep Learning	76
7.1.4	Collision Map	76
	References	77
	Appendix A Specifications Tables	85
A.1	Livox Mid-70 specifications table	85

List of Figures

2.1	GOTO and FOLLOW for master slave robot systems [22].	8
2.2	Color class map [24].	9
2.3	Detection based on color classification [24].	9
2.4	YOLO obstacle detection and obstacle avoidance path generation [28].	10
2.5	Re-planning control strategy introduced by Liu [29].	10
2.6	Object Detection vs Semantic Segmentation vs Instance Segmentation [34].	12
2.7	Industrial ISO standards relation to Agriculture environment and fu- ture withdraws.	13
3.1	Storage warehouse; (a); Phytosanitary treatments storage (b).	22
3.2	Greenhouses interior with middle corridors in concrete (larger corri- dor) and plants corridors on canvas over sand.	23
3.3	Open field operation with the middle corridor on dirt and gravel, and tunnel format greenhouses.	23
4.1	Software Architecture.	28
4.2	Transformation from the LiDAR frame to the camera frame.	29
4.3	Used coordinate system in Camera Vision (CV).	30
4.4	Outliers and Inliers illustration accordingly to HFOV.	32
4.5	Illustration of assigned points in the image to RGB point cloud.	33
4.6	Illustration of RGB point cloud points conversion to the collision map.	34
4.7	Illustration of the detected object projected in the RGB point cloud and, consequently, on the collision map.	35
4.8	Illustration of collision map projection in a collision detection operation.	36
5.1	Robot with sensors used to collision avoidance highlighted.	40
5.2	OAK-D specifications [53].	42
5.3	3D modeled support: (a); Support and lens attached to the camera (b).	43
5.4	OpenCV camera calibration corner detection - 1920x1080 image.	44
5.5	Matlab Camera Calibration corner detection - 1920x1080 image.	44
5.6	Visualization of transformation frames in Rviz [58].	45
5.7	1920x1080 resolution RGB image (a); 416x416 resolution RGB image (b).	47

5.8	Projected point cloud on “livox_link” visualized in Rviz.	48
5.9	Projected point cloud on “velarray_link” visualized in Rviz.	48
5.10	Projected accumulated point cloud on “livox_link” visualized in Rviz.	49
5.11	Illustration of the relation between focal length and AFOV [60].	50
5.12	Computed range image (a); Fused image from both RGB and range image (b).	51
5.13	Projected XYZRGB cloud, Fused Image, and RGB Image (416x416) visualized in Rviz.	52
5.14	Projected XYZRGB cloud and RGB image (1920x1080) visualized in Rviz.	53
5.15	Green tomatoes in the middle of the foliage visible in the point cloud (marked in red).	53
5.16	Fruits detection (a); Forklift detection (b).	54
5.17	Robot model (a); Defined joints and links (b).	55
5.18	Ackermann kinematics [52].	56
5.19	Relative transformation obtained by the robot positions relative to the map.	57
5.20	3D mapping from Mid-70 LiDAR and OAK-D kit RGB camera image along a greenhouse main corridor.	58
5.21	Obstacle Map projection (a); Obstacle Map overlaid by RGB point cloud (b).	59
5.22	Collision map composed by the 3 layers projection.	60
5.23	Updated collision map with advanced collision projection in a plants aisle.	61
5.24	Flow chart of the decision algorithm for collision evaluation.	63
5.25	Collision state machine upper level.	64
6.1	Robot starting position (a); Robot final position (b).	68
6.2	Collision map projection in the warehouse corridor.	69
6.3	Collision map projection in one plant aisle.	70
6.4	State machine transition to Avoidance state.	70
6.5	Collision map projection of a front obstacle creating a block situation.	71
6.6	State machine transition to Stop Operation state.	72
6.7	Detection of a person in plant aisles.	72
6.8	Collision map after person detection.	73
6.9	Hazard Zone state transition.	73
A.1	Livox Specifications [69].	86

List of Tables

1.1	Gantt Diagram listing the scheduled tasks.	5
2.1	Specifications addressed in ISO and IEC standards for the automation of agricultural machines and robotic systems for agriculture.	17
5.1	Comparison of Livox Mid-70 and Velarray M1600 specifications. . .	42

List of Symbols

Symbol	Description	Units
ρ	pixel size	m
f	focal length	m
f_x	focal length in horizontal axis	m
f_y	focal length in vertical axis	m
H	camera sensor size	m
n	number of pixels	pixel
R	rotation matrix in axis	rad
T	translation matrix	m
t	translation in axis	m
u	horizontal coordinate in pixel coordinate system	pixel
v	vertical coordinate in pixel coordinate system	pixel
X	position in coordinate axis x	m
Y	position in coordinate axis y	m
Z	position in coordinate axis z	m

Chapter 1

Introduction

1.1 Background

Today's agriculture is suffering from a lack of labor and thousands of euros are wasted every season for lack of workers at the right time. National Farmers' Union (NFU) [1], in the UK, reported a 33% lack of seasoning works in the harvesting/picking tasks in the season of 2021 – 2022. Riviera Produce Ltd also reported a loss of £500,000 [2]. Therefore, producers are forced to adapt and take advantage of the emerging technologies of Agriculture 4.0 [3]. Thus, the use of robots to help cultivate crops has been accepted in Europe in the last few years [4]. The use of these autonomous vehicles simplifies tasks such as transporting or spraying, but presents safety constraints for the coexistence of human workers and robots in the same space. Looking for accidents considering machinery operating in agriculture, human-machine collisions are the first problem, followed by mobile parts colliding with the operators [5].

Despite this fact, is not possible to find specific legislation for autonomous vehicles in agriculture. The most relevant International Organization for Standardization (ISO) standard that defines the design standards for agricultural machines and the requirements to be met by the operator (ISO 25119-2 [6]) is based on ISO 9001 [7], which refers to quality assurance. However, autonomous guided vehicle's and autonomous mobile vehicle's safety assurance and risk assessment rely on ISO 3691-4 [8], which can be associated with autonomous mobile vehicles in agriculture. Similarly, when it comes to human-robot interaction, ISO 10218-2 (2011) [9], only

addresses machine evaluation methods in the industrial environment. However, ISO 13482 [10], concerning safety requirements for personal care robots, is being revised and should address the human-robot interaction that can be applied in agriculture.

The methods to avoid collision in a cooperative working environment between humans and robots focus on two approaches: mechanical compliance, and strategies to minimize or avoid a collision, as specified in ISO/TS 15066 [11].

Safety Systems were introduced in industrial machinery, such as bumpers, airbags, and elastic absorption systems, as evaluated in [12] and [13]. These systems have been extended to industrial robots, such as Autonomous Guided Vehicles (AGV) in a factory environment. However, these methods lack the redundancy needed in Autonomous Mobile Robots (AMR) and collision avoidance approaches have become standard in the industrial environment.

Project ROBOCARE (norte-01-0247-feder-045289) [14] has the goal to develop robotic platforms for greenhouses environment. Since 2020, this project has been developing a platform for horticulture transportation in indoor and outdoor greenhouses. The AMR is based on a predefined trajectory following algorithm and only has collision detection when some obstacle or living being crosses the collision boundary.

To act in an environment like agriculture, the robot must be equipped with a system that allows it to have a 3D perception of its surroundings to avoid possible collisions and take precautionary measures. Despite the gaps in standards, the system must comply with existing standards for agriculture and adapt some of the industrial standards to prepare the robot for safe operation in the agricultural environment.

1.2 Motivation

To achieve this goal it is imperative for the robot to know the surrounding environment and also identify and classify some obstacles, such as people, objects, and animals, to act accordingly. Using computer vision to build a 3D map of the scene in real-time is then a necessary step to achieve for the robot to understand its environment and take action. This project uses Light Detection and Ranging (LiDAR) and Red, Green, and Blue (RGB) camera images to acquire 3D data from the scene. The LiDAR allows better accuracy regarding the distance measures of the objects in the scene, and the RGB camera allows the assignment of color information to LiDAR measures, creating an RGB 3D scene. Machine learning algorithms allow the detection and classification of objects or living beings in the scene and estimate their locations in the environment.

The main motivation of this work is to be able to evaluate the 3D scene with the help of object detection, to determine robot action according to ISO standards.

1.3 Goals

The main goal of this thesis is to study and explore techniques for the Development of a Smart Collision Avoidance System for Greenhouses Autonomous Mobile Robot. To accomplish this main goal, the secondary goals are:

The first goal is to merge a XYZ point cloud and a RGB image into a XYZRGB point cloud containing all the information in the scene. To achieve this goal there are already well-known techniques to calibrate the sensors and fuse their information [15].

The second goal is to accumulate the XYZRGB point clouds to create a 3D map in real-time, according to robot navigation.

The third goal is to identify and classify the objects in the environment and estimate their location in the 3D scene. To achieve this there are already machine learning techniques that allow object detection in the RGB image [16].

And finally, the main goal is to project the obstacle's location in a collision map and define the robot's actions according to its location in the scene and the arrangement of possible obstacles.

1.4 Work Scheduling

The first step is to install the sensors in the robot platform and prepare them to acquire data. After that, acquire data from the greenhouses, not only plant's aisle but also the main corridors where boxes and machinery are usually parked. Additional sensor information from the greenhouse warehouses is important since the robot has a different view of the scene in that environment. The acquired data then needs to be processed and fused to obtain the RGB point cloud, from the real environment, in real-time. Thus, it has to be accumulated over time, using robot position locations between acquisitions to determine the 3D map position. At the same time, the machine learning model for object detection is classifying the obstacles. To create the collision map, all the obstacles detected and the rest of the 3D map (accumulated point cloud) are projected in a 2D grid map to simplify the calculation of the following parameters.

1. Possible collision
2. Velocity reduction
3. Operating zone

The robot must act according to the parameters determined above, and the parameters must be set according to the standards of velocity, operating zones, and distance requirements, among others.

The Gantt diagram (1.1) presents the scheduling for research, design, implementation, and test of the system, as well as, the accepted review article in the ROBOT 2022 Conference [17].

1.5 Structure

This thesis is divided into the following main chapters: Requirements, Architecture, Implementation, Results, and Conclusions and Future Work.

The Requirements, Chapter 3, defines the requirements for the smart system based on the researched information and working environment.

Architecture, Chapter 4, establishes the software implementation methods to achieve the system purpose, as well as illustrates the results that are intended to be obtained with the implementation.

The Implementation, Chapter 5, presents all the design, hardware, computer algorithms, and implementation on the platform, from the sensor data acquisition, passing through some platform installations, to system implementation.

The results of the system implementation are evaluated in the Results, Chapter 6, where the sensors fusion, computed algorithms, map projection, and collision operation management are rated.

Finally, the Conclusions and Future Work Chapter 7 reflects on the developed work and system functionality and points to future work to enhance the implementation techniques at the level of sensor fusion and object detection.

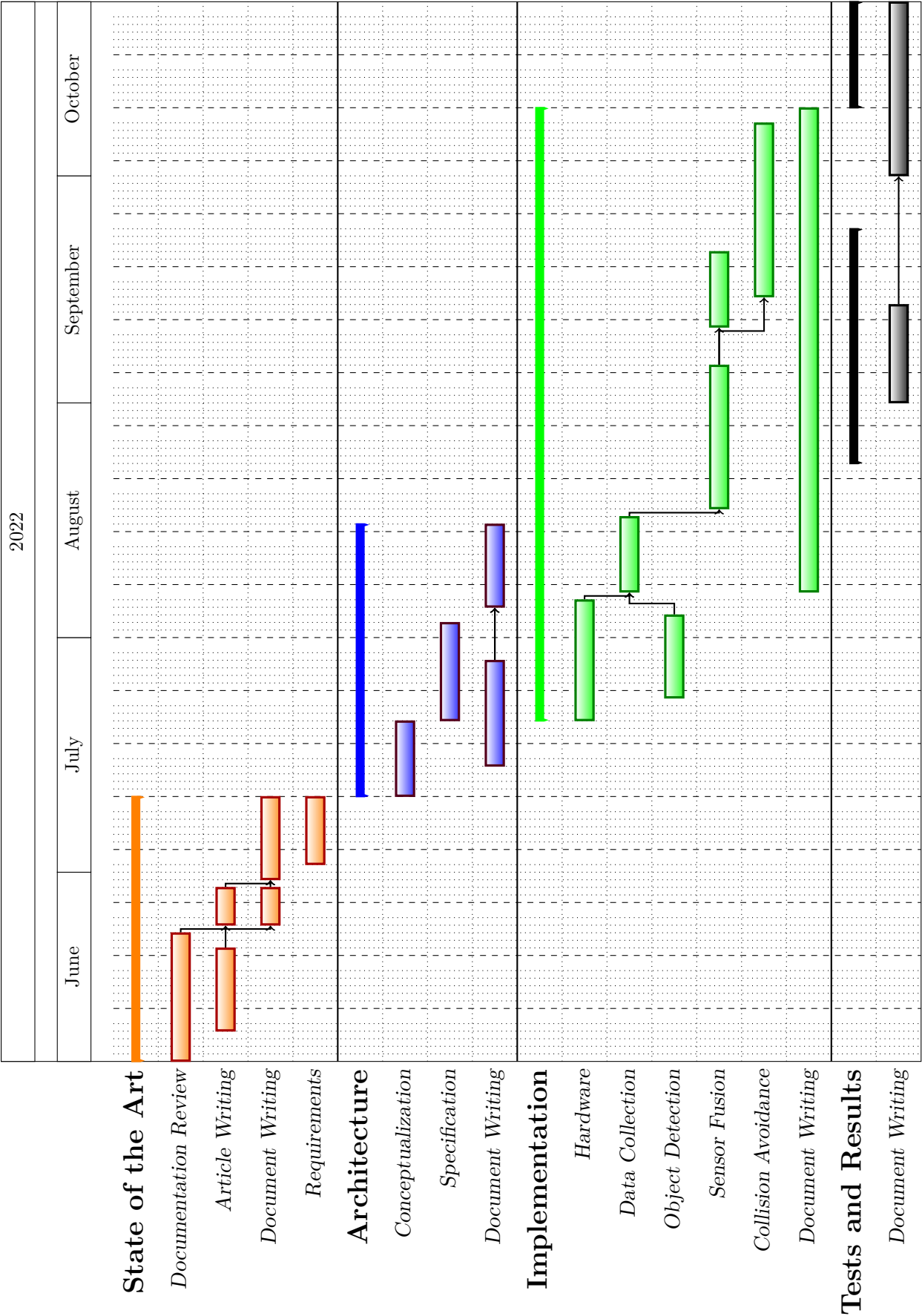


Table 1.1: Gantt Diagram listing the scheduled tasks.

Chapter 2

State of the Art

This chapter presents an overview of the human-robot interaction, methods for collision avoidance specifically built for agriculture environments, and the actual state of ISO standards for agriculture compared to industrial standards. Fusion sensor techniques and object detection techniques for the agricultural scene are also addressed.

2.1 Human-robot Interaction in Agriculture Environments

The dynamics of the agricultural environment imply a constant adaptation of the robots to the environment and to the actors in it. As described by Kruse *et al.* [18], the presence of humans requires approaches that take into account the constraints of human comfort as well as social rules. The adaptation to the actors of the environment implies the association of common social rules known by the robots, to which is associated the name social navigation [19]. This social navigation should adopt a behaviour similar to human behaviour. Ratsamee *et al.* [20] describe social navigation as a system that integrates human priority, is sociably acceptable, has safe motion, and reacts to human approach.

Despite the approach of conscious navigation from the environment agents, the fact that the environment is constantly changing forces the robot to learn the new dynamics with each change of the environment and its agents behaviour [21].

2.1.1 Interaction

In the past two decades, there were studied numerous ways of Human-Robot Interaction (HRI), from experiences with tractors and, more recently, with the new robot products and prototype implementations.

Noguchi *et al.* [22] developed a concept of interaction between human and robot (operator in tractor and robot) or master robot and slave robot that prevents collisions in advance and is divided into two algorithms, GOTO and FOLLOW, presented in Figure 2.1. The GOTO algorithm only indicates different navigation waypoints for the agents in the environment with predefined locations. On the other hand, FOLLOW allows the slave to monitor the master's position and admits changes in the system. It allows velocity and trajectory correction through a robot or robot-human interaction according to the collision risk assessment.

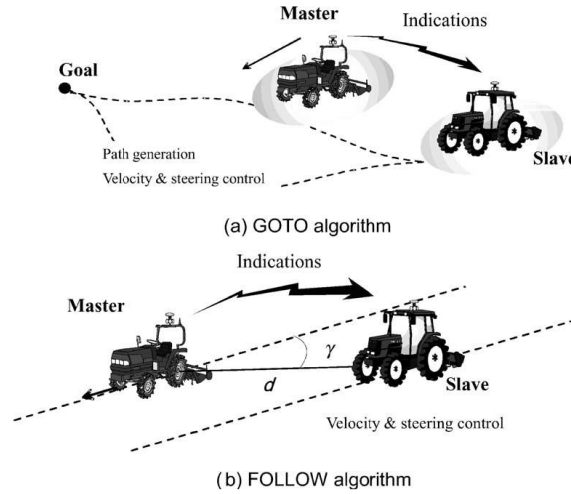


Figure 2.1: GOTO and FOLLOW for master slave robot systems [22].

Approaches such as the one by Vougioukas [23], use the Nonlinear Model Predictive Control (NMPC) method to control the vehicle position and deviation from fixed obstacles by measuring distances to them, a conventional approach.

2.1.2 Obstacle avoidance

Generally, obstacle avoidance methods can be divided into two categories:

- a) determining empty collision trajectories and,
- b) allowing real-time collision avoidance.

In environments such as greenhouses, where there are non-static agents in the environment (animals, persons, or other machines), object recognition and consequent avoidance approaches are required. Thus, it is possible to divide the problem

into two sub-problems, recognizing the obstacle and overcoming it. The obstacle determination can be done with the help of distance sensors or cameras, being the Light Detection and Ranging (LiDAR) technology the one that presents the most information, along with stereo RGB imaging.

Nissimov *et al.* [24] detect obstacles in greenhouses from the image of a Kinect [25] sensor by applying slope computation and analysing the pixel slope, its intensity and surrounding neighbour pixels. Figures 2.2 and 2.3 show the obstacle detection results from Nissimov computation.

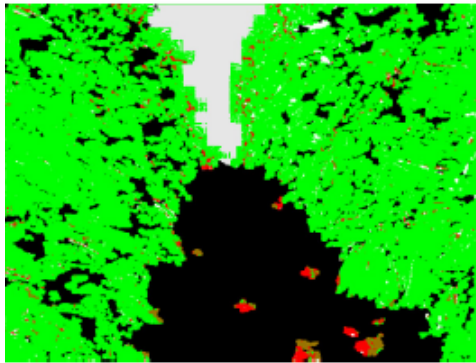


Figure 2.2: Color class map [24].



Figure 2.3: Detection based on color classification [24].

RADAR Detection And Raging (RADAR) technologies are capable of operating in harsh environmental conditions (dust, rain, fog, snow, etc.). Giulio *et al.* [26] present an approach for 3D localization of threatening obstacles by combining RADAR with stereovision, which allows to obtain precision range from the RADAR and good resolution from the stereovision, to find the boundaries of objects.

Artificial intelligence algorithms, such as You Only Look Once (YOLO) v3 based on a Convolution Neural Network (CNN) [27], allow to locate and classify the obstacle in the images obtained in real-time from camera sensors. Using the depth image is possible to estimate the relative distance to the obstacle and the direction [28]. The trajectory is corrected according to the update of the environment with the surrounding obstacles. Thus, it is re-evaluated the compliance of the safety distance to the obstacles in real-time, validating the trajectory path [28]. Figure 2.4 shows YOLO v3 detection and further avoidance path, respectively.

Liu *et al.* [29] present a path re-planning path method based on the distance to obstacle and monitoring of that distance in real-time by localization through LiDAR (see Figure 2.5). The re-planning algorithm relies on an objective function which increases in closer distances to objects [29] “The obstacle avoidance function describes the influence of the distance between the agricultural vehicle and the obstacle on the path re-planning”.

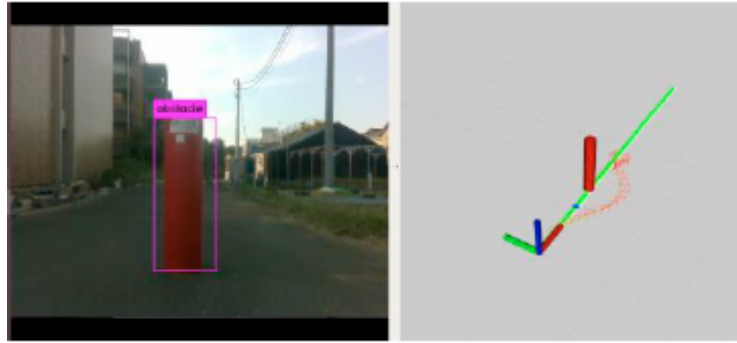


Figure 2.4: YOLO obstacle detection and obstacle avoidance path generation [28].

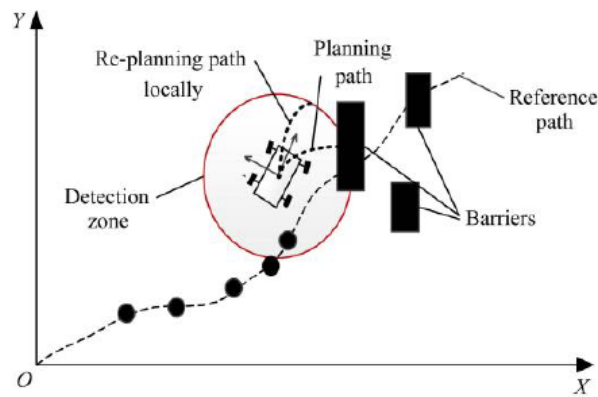


Figure 2.5: Re-planning control strategy introduced by Liu [29].

Although these methods are based on only one sensor, it is foreseeable that the combination of camera and LiDAR can provide more information about the robot's working environment and bring redundancy to security systems.

LiDAR and Camera Fusion

The use of two sensors is presented mainly in automotive approaches, such as, self driving cars and several of these approaches rely on camera and LiDAR fusion. The LiDAR is able to represent the environment depth in relevant detail and the camera allows to identify visual features in the environment.

Limin *et al.* [30] report exactly the issue of low sensing information with only one sensor in complex and dynamic traffic environments. Therefore, is presented an approach based on a combination of LiDAR and camera, with 2D depth map from LiDAR and object detection from the camera image and the 2D depth image. The information from both detection's is then combined to estimate the position of the object in the environment [30].

Even though sensor fusion is widely used in automotive environment new approaches for agriculture environment have been introduced in last years. Benet *et*

al. [31] presents a method for row tracking of various crops in various vegetation levels. This application uses a laser sensor, a color camera and an inertial measurement unit in real-time. The projection of laser scan in the color image allows to determine crop color pixels and noise or points outside the desired place. This characterization is computed to determine the 3D position of crop pixels and consequently of the crops.

Besides detecting crops, these fusion methods also allow to detect fruits. As presented by Kang et al. [32], the fusion of solid-state LiDAR (SSL) with RGB imagery allows to detect apples in orchards. This approach tested various extrinsic calibration methods performed in two steps: feature extraction and extrinsic computation [15] [33]. The vision system recognition and segmentation of the fruits is performed through an one-stage instance segmentation network architecture.

Deep learning

Deep learning is widely use nowadays in all type of implementations, such as, advanced driver assistance systems in automated cars on object detection to detect pedestrians or signals, industrial surveillance to detect pipe leaks or even simple IoT applications. In agriculture, as well, there are approaches which rely on deep learning to detect certain objects in the environment; one of most used is the fruit detection [16] and counting using object detection. The most used techniques in computer vision are object detection and instance segmentation. However instance segmentation imply semantic segmentation.

Object detection, is basically a classification and localization of an object in the image, however able to process it to multiple objects and uses various techniques. For object detection through deep learning, there are two key approaches:

1. Pre-trained models -> this approach uses an already trained model and re-trains it to suit a new application. This gives faster results, since the model is trained from millions of images.
2. Custom model -> training a custom object detector model involves developing a network architecture that learns the features of objects and compiling a large dataset of labeled images for training a CNN. The obtained results are really feasible, however the implementation time is considerable higher.

Semantic Segmentation creates pixel-level categories for object classes. Instead of locating and classifying objects individually, it groups them by classes.

Instance Segmentation combines object detection and semantic segmentation, whereas object detection spots objects in images by drawing boundary boxes and semantic segmentation create pixel categories for object classes. Instance segmentation generates a segment map for each map localizing each object from all possible classes.

Figure 2.6 illustrates the difference between Object Detection, Semantic Segmentation and Instance Segmentation.

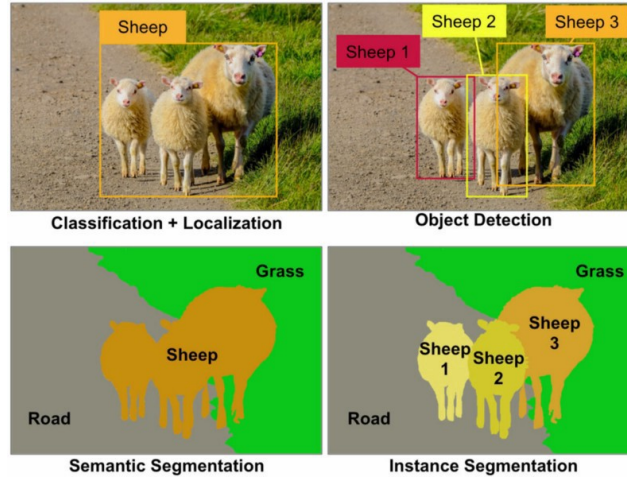


Figure 2.6: Object Detection vs Semantic Segmentation vs Instance Segmentation [34].

2.2 ISO's Standards

Although there are no ISO standards for autonomous vehicles in agricultural environments, the association of standards such as ISO 10218 [35] can help set standard approaches for agriculture robots systems.

Figure 2.7 shows the relationships between standards and the given specification level of each one, as explained in subsections 3.1 and 3.2. ISO 10218 refers to safety instructions for industrial robots. Relevant aspects of this standard are based on mechanical collision detection, force-torque sensors with the respective control algorithm, tactile and pressure sensors, the safe maximum allowed velocity for the robot, proximity sensors, area detectors and cameras to detect humans nearby [36].

2.2.1 Industrial Robotics Standards

ISO TS 15066 was published in 2016 to help in the safety shared workspace and risk assessment, which specifies the previous requirements of ISO 10218 for general safety with robots.

No less important documents are the EU mandatory Machinery Directive [37], ISO 12100 [38] for risk assessment, ISO 13849 [39] and ISO 11783 [40], specifically for communication on tractors and machines, detailed in subsection 3.2. Also, the electrical and electromagnetic compliance standards for safe operation as well as IEC EN 62061 [41] and the influence of detection on safety mode entry in the event of a single or multiple faults, IEC 61496 [42]. Based on the information presented,

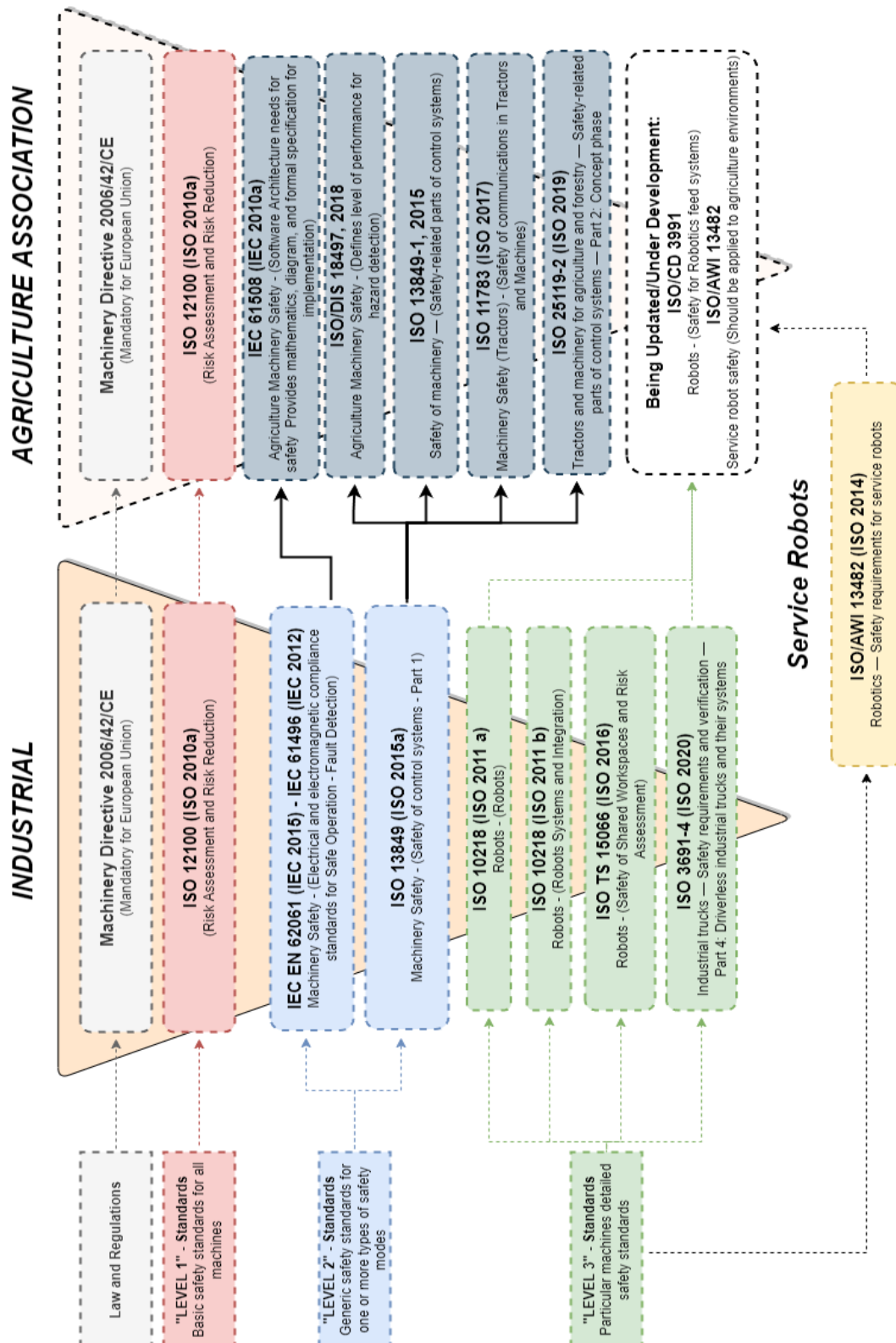


Figure 2.7: Industrial ISO standards relation to Agriculture environment and future withdraws.

it is possible to determine the level of detail and consequent influence of standards on the design of a product and its safety systems.

ISO 15066 is the most detailed specification for safety requirements in HRI for collaborative robots, mainly robotic arms. Depending on the specificity of the application, the robot should be integrated with different safety devices. The main goal of the safe collaboration is to minimize the mechanical risk which could arise from unexpected contacts in terms of severity and/or probability.

ISO TS 15066 presents four methods for human-robot contact:

- a) Safety-rated monitored stop – When an operator enters the collaborative workspace the robots stop.
- b) Hand guiding – In a hand guided operation the robot navigation must remain stopped.
- c) Speed and separation monitoring – The robots should be monitoring speed and distance between itself and the operator, and adjust it to maintain a separation over the set protective distance. When the separation is below a defined distance it must stop.
- d) Power and force limiting – Reduce occurrence of biomechanical contact risk through safety-related control systems. If pressure or other type of force limits remain under determined values during risk assessment, unforeseen collisions are allowed.

However, the standard that rules the interaction between humans and robots for autonomous forklifts, AGV, AMR, etc, and determines the specifications, implementation methods, for Eletro and Pressure-Sensitive Protection Equipment, and risk assessment is ISO 3691-4. According to this standard, in terms of detecting people in the path of the vehicle or robot, it must be equipped with detection means applying the following methods:

- a) Pressure sensors, physical, e.g. bumper, or virtual (ESPE) ¹ for detecting people.
- b) The detection means must operate at a minimum distance from the maximum width of the machine and its load in the direction of motion.
- c) The detection means must operate in such a way that the vehicle stops before there is contact between its rigid parts and a stationary person (not a person stepping into the front of the machine) according to the stipulated parameters, or in case of contact, it must not exceed the predefined force values.

¹Eletro-sensitive protective equipment.

- d) In case of stop, a visual or audible indication must be given as soon as the detection range is free and the machine is restarted. In case of using sensitive pressure protection (PSPE) ², the system should delay for 2 seconds before restarting.
- e) If the specifications can't be met the vehicle should not surpass the 0.3 m/s and have installed a laser scan or proximity switch that actuates 600 mm from the hazardous point.

ISO 3691-4 standard defines three access zones with different requirements, that can be totally or partially applied to a system [43].

1. **Operating Hazard Zone** – Space where a person can be exposed to a hazard →Velocity limited to 0.3 m/s or 0.7-1.2 m/s (depending on the clearance distance and conditions).
2. **Restricted Zone** – Space where only authorized persons have permission to enter →Velocity limited to 0.3 m/s (without detection means), 1.2 m/s (with detection means).
3. **Confined Zone** – Space where only authorized persons have permission to enter when the vehicles are stopped →Vehicles stopped.

In addition, the ISO 13482 standard for personal care robots specifies HRI regarding contact with human body parts, visual and audible signals in silent mode, as well as, stop operation when obstacle step into robot and distance measures, as also stated in ISO 3691-4.

Regarding collision avoidance and non-contact sensing these standard points:

1. ESPE should be used to detect persons.
2. The installation position should be according to risk estimation and should show reliability if used as primary sensing device.
3. In the case of detecting obstacles other than persons other ESPE can be used, and their reliability should be measured according to the risk assessment.

The parameters of minimum distance of detection of persons or separation distance between obstacles other than persons must follow the specifications of ISO 13855, regarding the positioning of safeguards in speed approach to human body parts [44].

²Pressure-sensitive protective equipment.

2.2.2 Migration to Agricultural Environment

Future farms, whether open fields or greenhouses, will have the imperative coexistence of robots and humans. The separation principle enunciated in numerous ISO standards, e.g., ISO 15066 for industrial environments, does not present direct solutions to the human-robot safety problem. However, starting from standards that include safety mechanisms and risk assessment methods, IEC 61508-1 [45] provides software architecture needs for the safety of agriculture machines, such as diagrams, computer-oriented tools, and formal specifications deriving from implementation [46]. ISO 25119-2 specifies the development of Safety-Related Parts of Control Systems (SRP/CS) on tractors for agriculture and forestry, and on self-propelled ride-on machines. It applies to electrical, electronic, and programmable electronic systems.

In 2017 the standards for serial control and communication data network for tractors and machinery for agriculture were upgraded in ISO 11783-1 [47]. Moreover ISO/DIS 18497 [48] appears with the rise of automation in tractors and heavy open field machinery. This standard specifies the design principles for highly automated machines, for example, tractors equipped with Global Navigation Satellite System (GNSS) and Real-Time Kinematic positioning (RTK)³ systems, in agricultural field operations. In addition, it lists information on safe work practices. Concerning the coexisting environment between humans and robots for agriculture, there are not yet specific standards. Table 2.1 presents a summary of the specifications addressed in each ISO standard for the automation of agricultural machines and robotic systems for agriculture, similar to AMR, such as tractors and self-propelled machinery.

However, a Committee Draft Registered, ISO/CD 3991 [49], concerning agriculture machinery, specifically robotic feed systems, is under development. On the other hand, there is no evidence of withdrawn standards that include autonomous vehicles in agriculture, so the standards for industry and driverless trucks, AGV and AMR, should be followed in the integration of robots in the agricultural environment to prevent and prepare robots for the upcoming standards. As ISO 13482, for personal care robots, will have appliances for the agriculture environment as soon as released.

Moreover, there is a group formed by ISO to come up with solutions for Smart Farming [50], however, it only addresses best practices, useful data from currently used machines and highlights the development of ISO/DIS 21384-1 [51] to standardize unmanned aircraft systems. From this overview is concluded that the standards for agriculture machines and safety control mechanisms should be implemented in an autonomous vehicle or robot for agriculture task, however collision avoidance presents a gap in standards and its implementations should rely in best practice and correlation with industry standards for AGV and AMR.

³Real-time Kinematic positioning (RTK) is a satellite navigation technique used to increase the accuracy of position data obtained by the global navigation satellite system (GNSS).

Table 2.1: Specifications addressed in ISO and IEC standards for the automation of agricultural machines and robotic systems for agriculture.

	IEC 61508-1	ISO 25119-2	ISO 11783-1	ISO 18497
Architecture Safety	X	X	X	
Implementation Specifications	X			X
SRP/CS		X		
Communications Protocol			X	X
Design Safety	X			X
Safe Control				X
Data Management			X	X
Risk Assessment				X
Collision Avoidance				
Code Quality Assurance	X			

2.3 Comparison between Collision Avoidance Methods and Standards

Based on the overview presented in the previous section, regarding standards for the agricultural environment, is preminent to use the ISO 3691 and ISO 13482, since the collision avoidance systems discussed in this paper apply to AMR, despite the HRI specifications not applicable to agriculture. As described in section 3.1 these two standards present the implementation methods and risk assessment for autonomous vehicles and robots.

Analysing ISO 3691-4, the distance to the human is evidenced, whether in relation to truck size, method **b)**, or in monitoring of distance to stationary persons, method **c)**, which is associated to a person in a stationary position in an orchard row or greenhouse corridor. The velocity is dependent on the access zone and must remain at a maximum of 0.3 m/s when safety requirements can not be met. This standard also covers post avoidance and the signaling that should be used to return to normal operation, method **d)**, or even in the distance after contact, methods **a)** and **c)**.

As far as collision avoidance systems are concerned, the distance after contact with ESPE is neglected since these systems are intended to avoid contact. Therefore, approaches like FOLLOW on tractors or agricultural robots, NMPC and all LiDAR obstacle determination respect the spacing between man and machine. In case of failure, the mechanical systems are enabled. On the other hand, the importance of monitoring the speed of dynamic objects and the robot itself is also highlighted. Once again, the FOLLOW method contemplates velocity corrections that in turn determine the instruction of deviation trajectories, performing it in pre-collision situations.

From a different perspective, velocity may not be the main factor for collisions, as obstacles are identifiable in more recent approaches using imaging and avoidance systems can reduce velocity as they have knowledge of obstacles in the environment or, for collaborative tasks, maintain a certain velocity while following the operator on field. Either detection by computing slope and image segmentation or by obstacle detection with deep learning and subsequent estimation of relative distance, allow locating obstacles or important features of the environment at a useful time, so that location prediction based on velocity estimation becomes inefficient.

The collision avoidance systems transmit confidence, although mechanical systems should be present and applied to agricultural robots, as presented in ISO 10218, which states that safety principles established in ISO 10218 can be utilized for other robots. These mechanical systems are also described in detail in method **e)**, of ISO 3691-4. However, the limits specifications of mechanical collisions, robot speed and certified sensors must be able to provide the robot with collision detection systems

that allow the redundancy of the robot safety in case of failure of the control system and must be executed in parallel. ISO 25119-2 also presents preventive specifications in the malfunction of agriculture machines that should be implemented in robotic systems to keep equal reliability to tractors. Concerning the robot decision that is part of the integration of a collision avoidance system, ISO 13482 states that the risk of harm occurring as an incorrect decision from the robot can be minimized either by increasing the reliability of the decision or limiting the effect of that decision in the global operation, which can be related to mission decisions on robot navigation that do not harm interaction with operators in the same workspace (collision avoidance).

2.4 Remarks

According to the state of the art it is preeminent that the fusion of data between lidar and camera is already studied with much evidence, however with few implementations for agricultural environments. On the other hand, object detection is commonly used in applications that involve detecting fruits or vegetables on plants and determine their position. Kang *et al.*, as mentioned above, present exactly one implementation with fruit detection, through a model trained for point cloud detection, on a 3D map merged through RGB image and point cloud.

When it comes to standards and as compared in section 2.3 there are still gaps that need to be filled. Thus the industry standards that appear to be equivalent for AMR's should be adopted.

Chapter 3

Smart Collision Avoidance Requirements

Accordingly to the state-of-art, collision avoidance systems have to be reliable and process environmental conditions fast to avoid hazard scenarios. Therefore, it is imperative for the software implementation that the code is as simple as possible and the algorithms used have a fast processing time. However, reliability must remain the top priority.

Before developing any type of software in such a dynamic environment is essential to evaluate and obtain a description of the scene. The organized and programmed tasks that already take place in this environment, and the agents and objects will influence the implementation. For that reason, is important to specify its characteristics for the development of the system.

3.1 Working Environment

The environment in vegetable or fruit production greenhouses can be divided into three distinct groups:

1. **Warehouses.**
2. **Greenhouses.**
3. **Open field.**

The environment inside warehouses, Figure 3.1 (a), despite the similar name, does not resemble the industrial environment. The warehouses that support greenhouse production are generally used to store plants and maintenance material, and to store machinery that operates in the greenhouses. They also performed tasks, such as spraying preparation for phytosanitary treatments and control of irrigation systems, among others, as presented in Figure 3.1 (b). Inside these warehouses, there are always people performing production support tasks, such as control of quantities produced and loading and unloading. For this reason, some areas are restricted to the use of machinery or require special attention to speed and maneuverability.



(a)



(b)

Figure 3.1: Storage warehouse; (a); Phytosanitary treatments storage (b).

The greenhouses, predominantly in a metal structure with plastic or glass covering, have a dynamic environment created by the plants in it and the living beings that interact with them, such as bees. The terrain can be flat or sandy with a canvas covering, or the whole terrain with some terraces. The greenhouses are, the majority of the time, organized in equally spaced corridors and there may be larger corridors, called main corridors, where forklifts or tractors circulate for transportation. Figure 3.2 shows the greenhouse in a real environment.

The open-field greenhouses depict productions where the main corridors are dirt roads. The greenhouses, in tunnel format, are arranged on the sides of these roads, and all machinery locomotion between greenhouses and warehouses takes place in the open field. Figure 3.3 shows the production environment in open field tunnel-type greenhouses.



Figure 3.2: Greenhouses interior with middle corridors in concrete (larger corridor) and plants corridors on canvas over sand.



Figure 3.3: Open field operation with the middle corridor on dirt and gravel, and tunnel format greenhouses.

Whether in the greenhouses of Figure 3.2, or the greenhouses of Figure 3.3, there is always the cohabitation between agents of the environment, mainly, people, plants, animals, and machinery. In both cases, some objects that are not always in the same position, they are dynamic, such as pallets, empty or full boxes of fruits or vegetables, carts to help with harvesting, and tools and maintenance material.

3.2 System Requirements

Based on the information researched and the analysis of the work environment it is then possible to determine the requirements for the system.

From the description of the environment, it can be seen that the aisles of plants are always moving due to the circulation of air. Therefore, the robot may collide laterally with some of the leaves, fruits, and vegetables. The collision with leaves can

be disregarded since it will not affect the production or the health of the plant. On the other hand, collision with fruits or vegetables can damage them, so the system must be able to identify them and avoid possible collisions.

Many systems used in industrial environments rely on collision detection at the hardware level, such as certified LiDAR for collision detection. However, in this dynamic environment is required a smarter avoidance system, since objects can't be assumed as obstacles just because, the distance to the robot is inferior to a defined minimum distance parameter, as stated in ISO 3691-4.

Therefore, the **first** requirement is a smart perception approach to the operating environment. The system should be based on a multi-sensor fusion to improve the visualization of the 3D environment. LiDAR and RGB camera fusion is a suitable combination since it allows the creation of a 3D map of the scene by fusing point cloud point's position and color given by the sensors. In the 3D map the system should be able to detect the defined agents in the work environment:

1. Persons;
2. Fruits or vegetables;
3. Other objects.

These must be classified and located so that the robot can be aware of possible dangers to its operation and the safety of the environment.

As presented in Chapter 2, the recommended approach to detect objects is the use of deep learning. Therefore, the system should have an object detection model to determine the object's position in the RGB image from the camera. This is the **second** requirement, ensuring a detailed characterization of the objects in the environment.

The **third** requirement is the software framework implementation. The robot platform used in project ROBOCARE, as most of the AMR, uses Robot Operating System (ROS). ROS is reliable, allows easy communication between different processors in the robot, and simplifies the debugging and monitoring of the system, which makes it perfect for new and complex implementations. Therefore, the smart collision avoidance system should be implemented in the ROS software framework. The platform's specifications are detailed in Section 5.1

The **fourth**, and last requirement, is the management of the robot operation according to the environment perception. The system must be able to evaluate possible collision and, when detected a collision, manage the robot's operation in one of these two ways:

1. Reduce velocity (ISO 3691-4); and,
2. Stop operation (ISO 3691-4).

And also, introduce a new security feature, intelligent collision avoidance, suitable for dynamic environments implementations. Then, is required a simple and robust system, usually state machines, to ensure correct operation management.

Chapter 4

Smart Collision Avoidance Architecture

This chapter illustrates the proposed software architecture for the complete smart system and defines specific aspects for implementation.

4.1 Collision Avoidance Architecture

To accomplish smart collision avoidance is proposed a system with 4 blocks:

1. Detection
2. Sensor Fusion
3. Environment Projection
4. Collision Avoidance

Figure 4.1 illustrates the system architecture.

The Detection block receives RGB images and detects objects in the image through a TinyYoloV4 object detection model which localizes and classifies the objects in the image.

Sensor Fusion is the block responsible for interpreting the acquired data and giving it meaning. For that, the fusion should be as accurate as possible, which

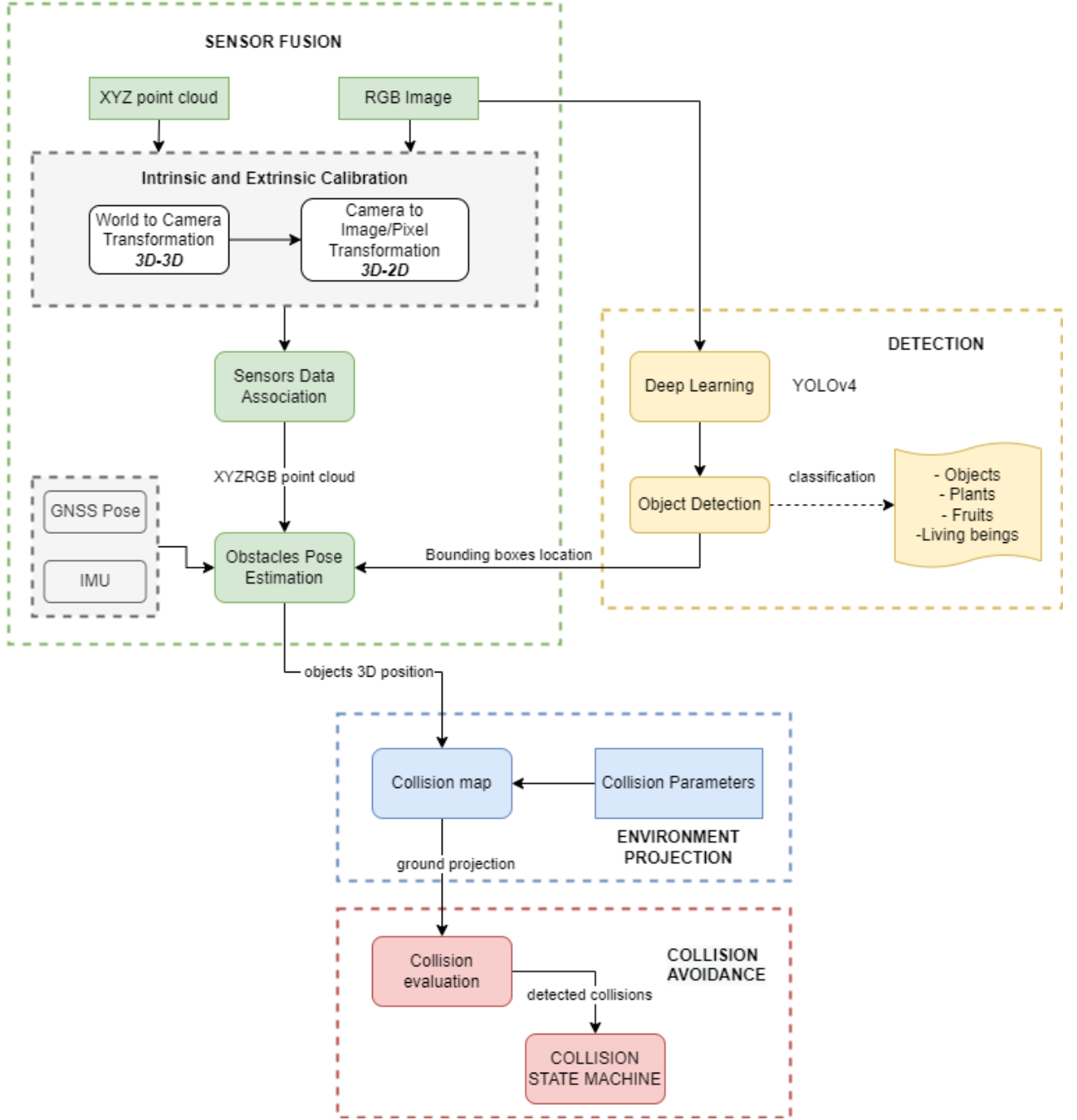


Figure 4.1: Software Architecture.

supposes a precise extrinsic calibration between sensors. Also, reliable object localization in the images and robot's pose synchronization, to consequently obtain enhanced data with better characterizing information.

The Environment Projection block projects the surrounding objects in the environments and the detected obstacles from the object detection to create the collision map.

Finally, the Collision Avoidance block evaluates the collision map to detect possible collisions and define the type and the predicted position of the obstacle. Therefore, the system enters the collision state machine, explained in Section 4.5.2.

4.2 Sensor Fusion

An RGB camera is used to take the points from the world and project them onto a 2D plane seen as images. This transformation is generally divided in **Extrinsic** and **Intrinsic**. They are both distinct and the first depends on the location and orientation of the camera which does not intervene in the intrinsic parameter such as focal length. However, in this implementation, the points from the projected point cloud have to be transformed from the LiDAR frame to the camera frame, as illustrated in Figure 4.2. According to the camera, Horizontal Field Of View (HFOV) and Vertical Field Of View (VFOV) the projected points are defined as inliers or outliers on the camera image projection.

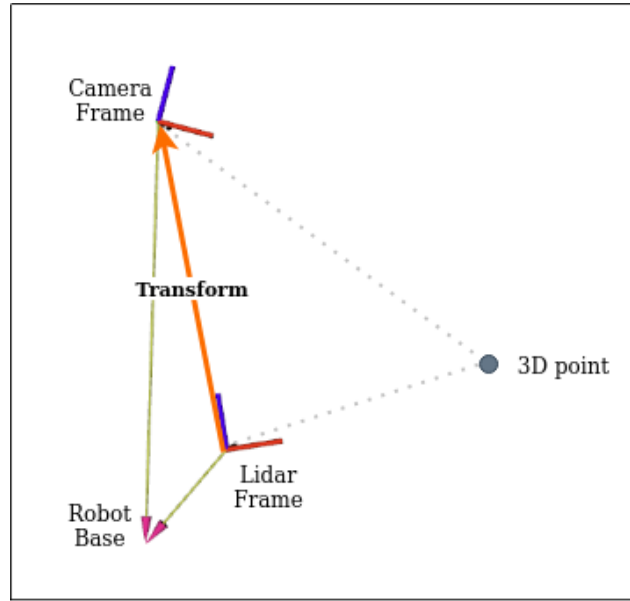


Figure 4.2: Transformation from the LiDAR frame to the camera frame.

4.2.1 Extrinsic and Intrinsic

Extrinsic and Intrinsic matrices allow the transformation of the points, defined by $Pxyz$. By obtaining the inliers it is possible to transform the point cloud into an image. Figure 4.3 illustrates the conversion from the world coordinate system (3D) to the Pixel coordinate system (2D).

The extrinsic matrix $\mathbf{E}$, equation 4.6, is composed of the rotation in the three axes x, y, z and the translation in the same axis. This matrix is the transformation from

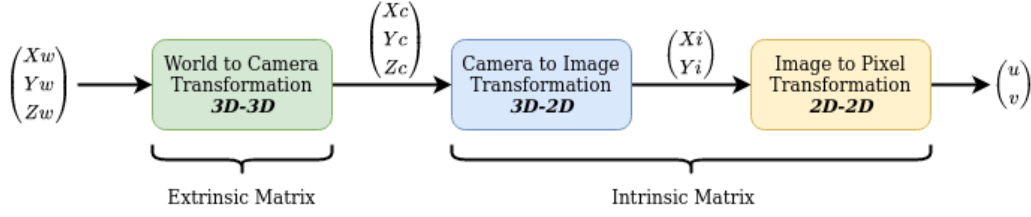


Figure 4.3: Used coordinate system in Camera Vision (CV).

the world coordinate system to the camera coordinate system $Pxyz$, decomposed in the equations 4.1, 4.2, 4.3, 4.4, and 4.5. Applying the intrinsic matrix, $Pxyz$ is transformed from the camera coordinate system to the pixel coordinate system Pu, v .

3D-3D

Rotation matrix in x axis where ψ represents roll angle.

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \psi & -\sin \psi & 0 \\ 0 & \sin \psi & \cos \psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.1)$$

Rotation matrix in y axis where ϕ represents pitch angle.

$$R_y = \begin{bmatrix} \cos \phi & 0 & \sin \phi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \phi & 0 & \cos \phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.2)$$

Rotation matrix in z axis where θ represents yaw angle.

$$R_z = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.3)$$

Translation matrix where t_x , t_y and t_z represent the translation in x, y, z axis.

$$T_{xyz} = \begin{bmatrix} 0 & 0 & 0 & t_x \\ 0 & 0 & 0 & t_y \\ 0 & 0 & 0 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.4)$$

$$R_{xyz} = R_x \times R_y \times R_z \quad (4.5)$$

$$E_{xyz} = R_{xyz} \times T_{xyz} = \begin{pmatrix} R_{3x3} & t_{3x1} \\ 0_{1x3} & 1_{1x1} \end{pmatrix}_{4x4} \quad (4.6)$$

The 3D-3D point transformation from world to camera is represented by equation 4.7.

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} = \begin{pmatrix} R_{3x3} & t_{3x1} \\ 0_{1x3} & 1_{1x1} \end{pmatrix}_{4x4} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \quad (4.7)$$

The application of the intrinsic matrix proposes two conversions, from the camera coordinate system (3D) to the image coordinate system (2D) and then to Pixel coordinate system (2D). This conversions are represented by the 3D-2D transformation (equation 4.8) and the 2D-2D transformation (equation 4.10).

3D-2D

$$\begin{pmatrix} X_i \\ Y_i \\ Z_c \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} \quad (4.8)$$

To obtain the point coordinates in the camera 2D, Pinhole Model, X_i and Y_i , is necessary to multiply the coordinates x and y in the camera optical axis z by the focal length f , as represented in equation 4.9.

$$X_i = f \frac{X_c}{Z_c}; Y_i = f \frac{Y_c}{Z_c} \quad (4.9)$$

2D-2D

From the previously known pixel size, it is possible to determine the pixel position from the camera coordinate system to the pixel coordinate system through the equation 4.10.

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} \frac{1}{\rho_u} & 0 & c_x \\ 0 & \frac{1}{\rho_v} & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} \quad (4.10)$$

Equation 4.11, simplifies obtaining the u and v pixel coordinates in the image, being u the horizontal coordinates (starting from the left) and v the vertical coordinates (starting from the top). ρ represents horizontal and vertical pixel size.

$$u = \frac{1}{\rho_u} X_i; v = \frac{1}{\rho_v} Y_i \quad (4.11)$$

4.2.2 RGB image to Point Cloud

The extrinsic and intrinsic parameters allow to remove the range in Z_c from the points and project them in the 2D plane. However, depending on the camera HFOV and VFOV, each point should be defined as an inlier or outlier, as presented in Figure 4.4. The red point represents the inliers and the blue points are the outliers (this Figure only illustrates HFOV).

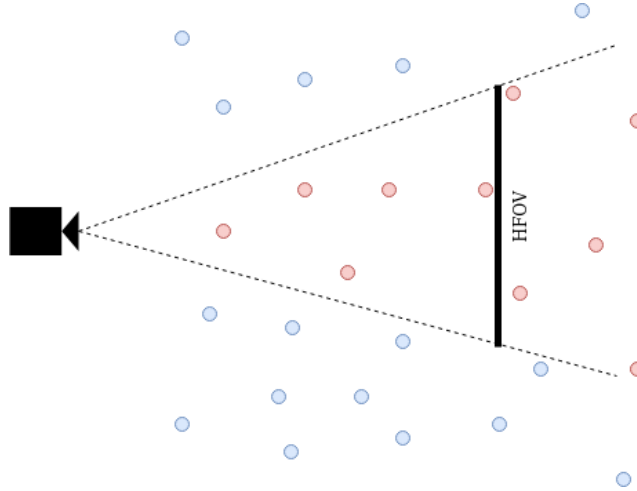


Figure 4.4: Outliers and Inliers illustration accordingly to HFOV.

The inliers from the LiDAR are projected in the RGB image and from that projection, it is possible to obtain two important features:

1. Pixel location projection; and,
2. Point cloud point RGB color.

The range is a piece of important information associated with each pixel image. The association is generally made accordingly to the minor location point inside the

pixel coordinates. This information allows for locating the obstacles and features of the image in the real environment.

From the RGB information of the camera image, it is possible to assign an RGB color to the point cloud points from the pixel. Therefore, the composition of all points with RGB creates an RGB Point Cloud which represents the real environment in a 3D scene with precision. Figure 4.5 shows the assignment between pixels and points in the RGB point cloud.

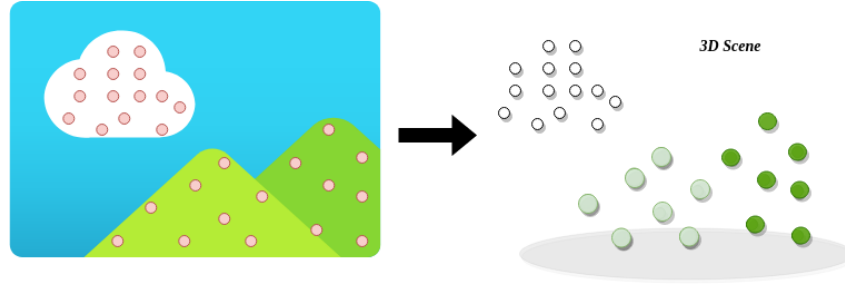


Figure 4.5: Illustration of assigned points in the image to RGB point cloud.

4.3 Environment Projection to Collision MAP

The collision map is where all obstacles, people, and vegetation in the robot's surrounding environment are projected. This map is composed of three layers.

1. Complete projection of the environment and detected obstacles;
2. Location of the robot on the collision map, including the excess width caused by the rotation of the wheel (robot's footprint);
3. Trajectory that the robot will follow in the next few meters, a parameter that can be defined according to the operation;

To determine the position of objects and obstacles, as well as persons, the XYZ information of each point is projected on the collision map with an associated color. Figure 4.6 illustrates an environment projection from the RGB point cloud to the collision map.

In the illustration, only the points corresponding to the mountains are projected since the robot could never collide with the clouds. The trajectory, in red, is projected on the map from the robot navigation computation. The robot footprint is projected in blue and the environment is projected accordingly to point colors, which allows to distinguish them.

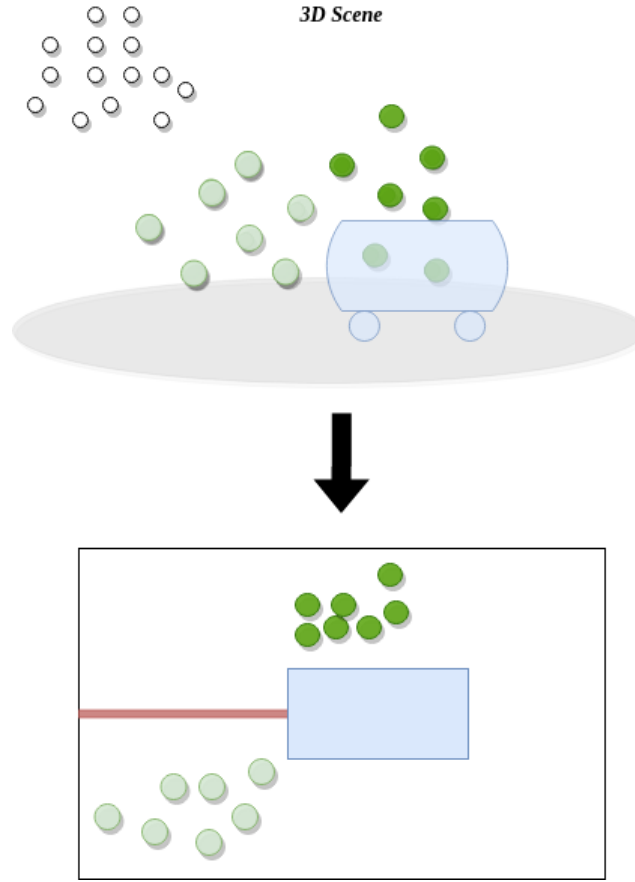


Figure 4.6: Illustration of RGB point cloud points conversion to the collision map.

4.4 Object Detection

The need to project obstacles onto the collision map makes it necessary to identify them and determine their location. Hence, to detect the desired objects in the scene a deep learning model for object detection is used. The model used is an object detection model TinyYoloV4 processed by the RGB camera. The onboard processing enhance the performance of the model, instead of overloading the robot's main computer to process the model detections.

As the image is merged with the point cloud, it is possible to determine the location of the objects by evaluating which points lie within the circle (inliers). Figure 4.7 illustrates the detected points with a different color (purple) projected in the RGB point cloud and, consequently, on the collision map.

The points belonging to each detected and classified object are projected onto the collision map to enable collision detection.

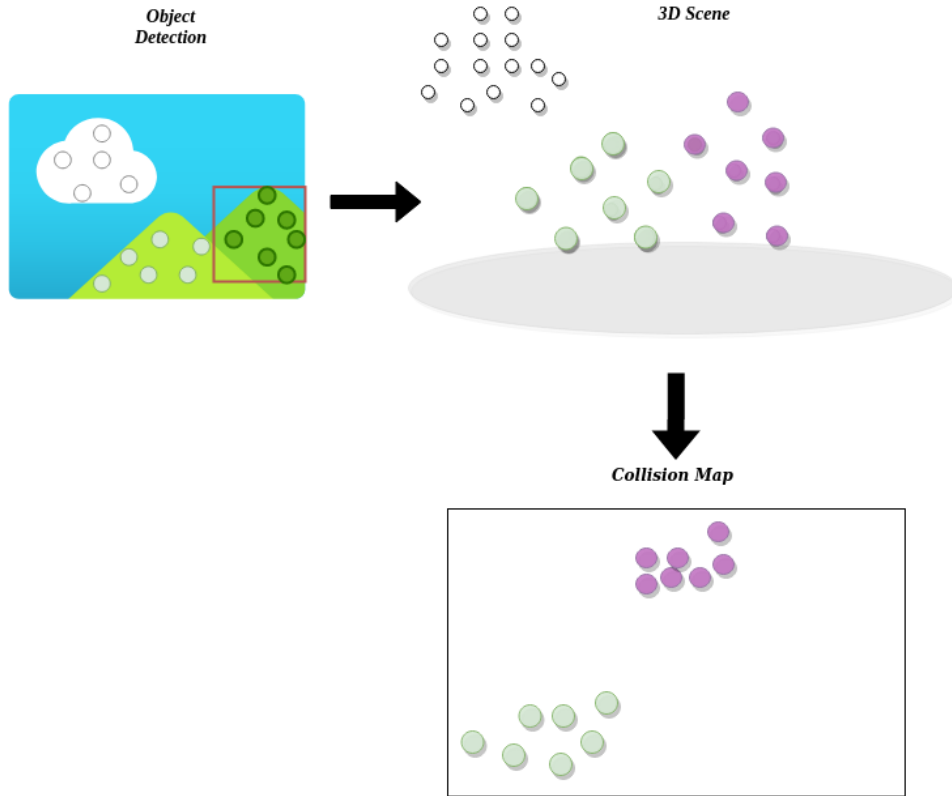


Figure 4.7: Illustration of the detected object projected in the RGB point cloud and, consequently, on the collision map.

4.5 Collision Avoidance

The evaluation of the collision map and determination of the actions to be performed by the robot, after evaluating its surroundings, is performed based on a state machine. However, collisions must be predicted according to a series of parameters, predefined for the type of operation and environment in which the robot acts.

According to the proposed approach, the parameters that must be defined before the evaluation of possible collisions are the following:

1. Collision areas;
2. Objects present in the scene;
3. Position of the robot's footprint;
4. Trajectory.

4.5.1 Collision Evaluation

The determination of collisions needs the interpretation of the defined parameters before evaluating the collision. First, it is necessary to determine the collision areas of the robot and classify them.

Second, it is necessary to determine the type of objects, present in the scene, with which it is possible to collide. Once the objects are determined, the evaluation of each collision is performed. This is done by predicting the collision of the robot footprint with the projection of the objects (identified as obstacles) on the collision map.

Thirdly, it is needed to validate the collisions by projecting the advancing robot footprint onto the collision map and confirming the collision points. Figure 4.8 illustrates the constitution of the collision map after projecting all the parameters.

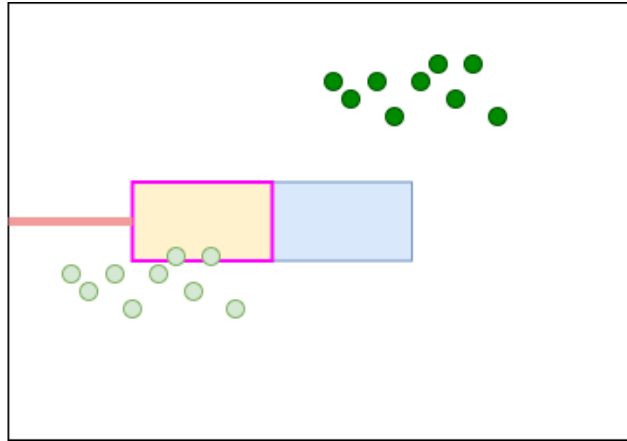


Figure 4.8: Illustration of collision map projection in a collision detection operation.

In Figure 4.8 the collision areas are perceptible by the rectangular footprint of the robot (in blue), which reveals a front, left, right and back possible collision areas (in pink). The illustration also shows the projection of the advanced footprint (in yellow) for collision prediction with the projected environment (light green dots).

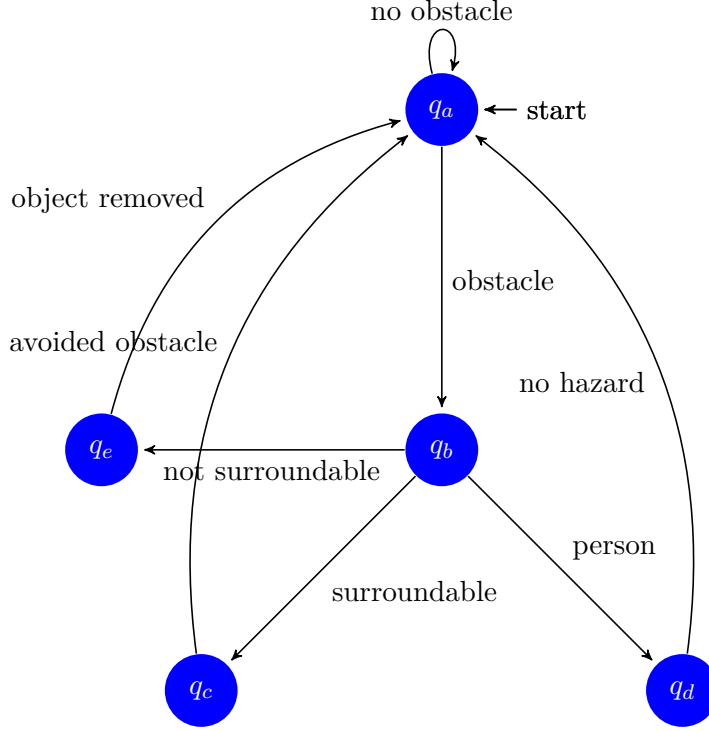
The type of object or living being (person, plants, ...) that the robot is likely to collide with, validated by the forward projection, triggers an action in the collision state machine. Thus, the collision state that the state machine is in limits the robot's actions during its operation.

4.5.2 State Machine

The state machine is divided into five states:

- a) Normal Operation;
- b) Collision Determination;
- c) Avoidance;
- d) Hazard Zone;
- e) Stop Operation.

Being $\mathbf{qx}$ each state of the state machine, with $\mathbf{x}$ corresponding to each state mentioned above, the state machine is represented as follows:



The Normal Operation states ($\mathbf{qa}$) is the default node for robot operation in the collision state machine. The detection of a possible collision with an object in the collision map causes the transition from $\mathbf{qa}$ state to Collision Determination ($\mathbf{qb}$). This state represents the robot evaluation operation to determine which type of previously defined obstacle is preset in the collision map. Therefore, depending if the obstacle is **surroundable**, **not surroundable** or a **person**, the machine transits to states, Avoidance ($\mathbf{qc}$), Stop Operation ($\mathbf{qe}$) and Hazard Zone, respectively.

Since these three states represent collision management, they must have a redundancy loop to guarantee the end of the operation and the state transition back to Normal Operation if:

1. the obstacle is avoided (Avoidance);
2. the person is not present in the environment (Hazard Operation);
3. the obstacle was removed or the robot entered manual mode (Stop Operation).

4.6 Remarks

The four implementation blocks were defined and the methods to be used in the implementation were specified. The 3D map of the environment relies on the point cloud, RGB image, and object detection. Therefore, data has to be acquired to supply the inputs of the Sensor Fusion and Detection blocks. The collision map

projects the 3D map, XYZRGB point cloud, onto the ground which enables collision detection, hence, the Collision Avoidance block, relies on the quality of data provided by the previous blocks.

The Architecture specification and the detailed block interconnection simplify the implementation of the system.

Chapter 5

Smart Collision Avoidance Module - Hardware and Implementation

5.1 Robot Platform

Primarily to the development of the collision avoidance system, it was necessary to obtain data from the environment inside the greenhouses. Moreover, characterize some common features and obstacles during the operation of the robot in standalone and collaborative tasks. For this purpose, the sensors define in the system requirements chapter 3 were installed in the Acrobat robotics platform developed within the ROBOCARE project (as previously mentioned).

This platform features an Ackermann-type traction system with a drive train on the front axle and Ackermann steering on the rear axle [52]. A traction system that is similar to that of forklifts and street cleaning trucks.

Localization is performed by Global Navigation Satellite System (GNSS) and Real-Time Kinematic positioning (RTK)¹ with an accuracy of approximately 3 to 4 centimeters, which allows precise navigation within the greenhouse aisles. The platform also has two LiDAR sensors, Livox Mid 70 and Velodyne Velarray M1600

¹Real-time kinematic positioning (RTK) is a satellite navigation technique used to increase the accuracy of position data obtained by the global navigation satellite system (GNSS).

B2, in the front panel, just above the traction system, and a stereo and RGB camera in the upper arch of the platform.



Figure 5.1: Robot with sensors used to collision avoidance highlighted.

Figure 5.1 presents the robot with an OAK-D kit camera marked in red, Livox Mid-70 in blue and Velarray M1600 marked in green, a prototype for automotive vehicles. The sensors are positioned in strategic positions. Therefore, the LiDARS in the bottom enhance plant detection from the bottom stem to the top, and the camera in the top arm of the robot allows for visualizing the ground. The camera is tilted 10° at a height of 1.65 meters, which, apart from a ground vision from 1 meter of the robot front, also allows for visualizing high obstacles or agents in the environment.

The robot is also equipped with four black boxes making a rectangle in the top arm. The boxes have an Ultra Wideband (UWB) radio technology to detect the

operator's relative position to a tag in its pocket. However, only visual perception sensors were used in the smart collision avoidance system implemented.

5.2 Perception Sensors

As the robotic platform was in the prototype development phase, it was possible to use not only the Livox Mid 70, already installed on the platform but also the Velarray M1600. The Velarray is also in the prototype phase but might be implemented in a future version of the platform due to its characteristics.

5.2.1 LiDARs

The Velarray has a higher vertical resolution than the Livox and a higher density of points per sample, as tested and confirmed in Section 5.4.1. Livox has a field of view (FOV) better suited for the application since it reproduces a 70° diameter cone allowing it to obtain distances from the top of the greenhouse. This characteristic facilitates the implementation of SLAM with those same static features. However, for the collision avoidance system specifications, the following features were taken into account:

1. Resolution;
2. Acquisition Volume (Point Cloud Density);
3. Acquisition Speed.

Appendix A.1 attaches the specification table of the Livox Mid-70 and Section 5.4.1 details the visual data possible to acquire with both LiDARs.

Velarray M1600 is, yet, a prototype, implying full concealment of some of its specifications. Therefore, at the date of this thesis, no sensor information or acquired data that has not been duly authorized by Velodyne will be revealed.

Consequently, the comparison between both sensors can be done by presenting, which Velarray specifications are higher or lower than the Mid-70. Table 5.1 presents the relation between both sensors specifications.

5.2.2 OAK-D Camera

The camera used is an OAK-D which includes two cameras for stereo and an RGB camera. Figure 5.2 shows both cameras specifications.

The RGB camera has an IMX378 from Sony [54] sensor and aperture of 2.0 and the stereo camera has an OV9282 from OmniVision [55] sensor and aperture of 2.2. The RGB camera has a maximum frame rate of 60 fps, which is enough for the use and implementation of the collision avoidance system, considering the

Table 5.1: Comparison of Livox Mid-70 and Velarray M1600 specifications.

	Mid-70	Velarray M1600
Max Range (10% reflectivity)	90 m	Lower
HFOV	70.4°	120°
VFOV	70.4°	32°
Horizontal Resolution	0.03°	Lower
Vertical Resolution	0.28°	Higher
Range Precision (0.1 - 1 m)	3 cm	Lower
Data Acquisition	100000 points/s	330000 points/s
Data Port	100 Mbps	Higher
Data Transmission Rate	10 Hz	Higher
Enclosure Rate	IP67	Equal
Operating Temperature	-10°C – 60°C	Higher Range

Camera Specs	Color camera	Stereo pair
Sensor	IMX378	OV9282
DFOV / HFOV / VFOV	81° / 69° / 55°	82° / 72° / 50°
Resolution	12MP (4032x3040)	1MP (1280x800)
Focus	Auto-Focus: 8cm - ∞	Fixed-Focus: 19.6cm - ∞
Max Framerate	60 FPS	120 FPS
F-number	2.0	2.2
Lens size	1/2.3 inch	1/4 inch
Pixel size	1.55μm x 1.55μm	3μm x 3μm

Figure 5.2: OAK-D specifications [53].

robot's maximum velocity of 1.2 m/s . Although the FOV of the camera is enough to view the greenhouse's aisles, it was added a 120° wide-angle lens to enlarge the robot's FOV when navigating in more open areas, such as main aisles or greenhouse warehouses.

To implement the lens in the camera and ensure that it remains immovable, a support was designed and 3D printed in PolyEthylene Terephthalate Glycol (PETG). PETG has been used in other parts of the robot and validated to withstand high greenhouse temperatures and possible impacts. For easy removal and robust attachment, there were used 3 mm screws to fix the support on the camera and the lens. Figure 5.3 shows the 120° lens modeled support to attach the lens.

5.3 Software Implementation

The software implementation relies on the software architecture presented in Chapter 3. However, previously procedures needed to be done to obtain relevant data for sensor implementation, namely:



Figure 5.3: 3D modeled support: (a); Support and lens attached to the camera (b).

1. Camera Calibration (Intrinsic);
2. Transformations between LiDARs and Camera (Extrinsic).

5.3.1 Camera Calibration

To obtain the camera intrinsic parameters two different approaches were used to confirm the reliability of the calibration and apply them correctly. The first tool used was the OpenCV toolbox for camera calibration [56]. For calibration was used a chessboard of 9 per 7 squares, each square with 28 millimeters of side length, in image resolutions of 416x416 and 1920x1080 pixels.

The intrinsic matrix $\mathbf{I}$ (equation 5.1) gives the horizontal and vertical focal length f_u and f_v , and image origin c_u e c_v in pixel, as presented in Section 4.2.1. Therefore, were obtained the intrinsic parameters presented in equations 5.2 and 5.3.

$$\mathbf{I} = \begin{pmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix} \quad (5.1)$$

Figure 5.4 presents the camera calibration corner detection in the 1920x1080 resolution image.

$$\begin{pmatrix} 460.0157 & 0 & 203.3042 \\ 0 & 463.4635 & 165.9310 \\ 0 & 0 & 1 \end{pmatrix} \quad (5.2)$$

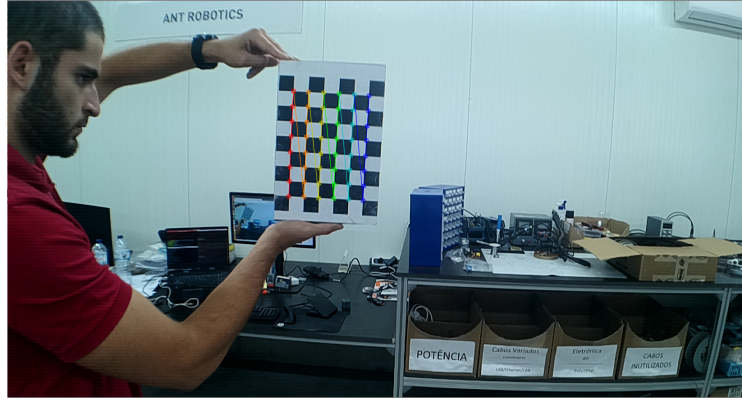


Figure 5.4: OpenCV camera calibration corner detection - 1920x1080 image.

$$\begin{pmatrix} 1143.5677 & 0 & 950.1413 \\ 0 & 1140.0413 & 559.9947 \\ 0 & 0 & 1 \end{pmatrix} \quad (5.3)$$

The second tool used for calibration was the MATLAB application Camera Calibration application [57]. Figure 5.5 presents the corner detection on the 9 by 7 chessboard of each valid image, since the API determines which images are suitable for calibration. These intrinsic parameter obtained from the calibration process are presented in equations 5.4 and 5.5 respectively.



Figure 5.5: Matlab Camera Calibration corner detection - 1920x1080 image.

$$\begin{pmatrix} 429.7195 & 0 & 192.0786 \\ 0 & 424.9374 & 237.5174 \\ 0 & 0 & 1 \end{pmatrix} \quad (5.4)$$

$$\begin{pmatrix} 1154.4000 & 0 & 926.5726 \\ 0 & 1152.7000 & 543.3537 \\ 0 & 0 & 1 \end{pmatrix} \quad (5.5)$$

The parameters obtained were used for the implementation of the Sensor Fusion block presented in Section 4.2.

5.3.2 Transformations

The transformations from the robot's base to the sensors and actuators are a need in robot implementation. Therefore, to implement the system, the position of the three sensors is determined from the real location in the robot frame. Regarding the locomotion, the “base_link” represents the robot's main link, located in the center of the traction axle, and the “direction_wheel_link” represents the middle angle from the Ackermann system wheels. Other sensors are linked to the “base_link”, thus, the three sensors needed for the collision avoidance system are linked as presented in Figure 5.6.

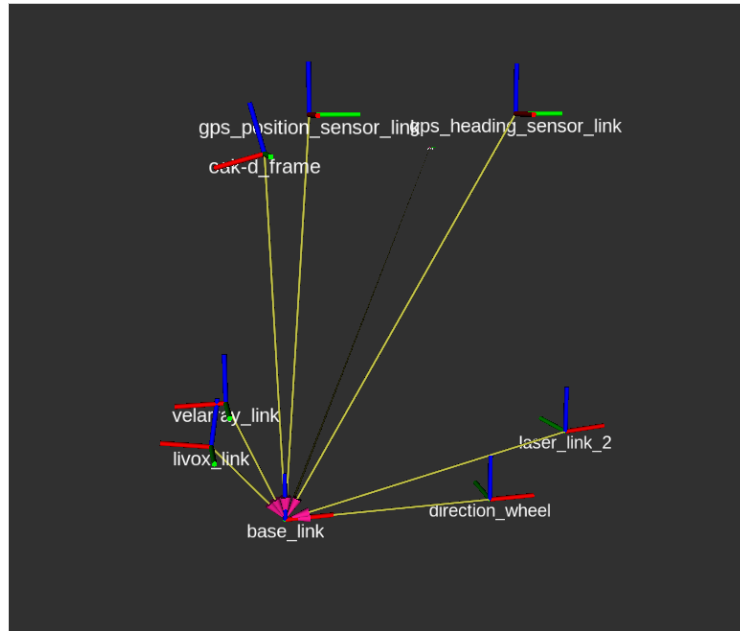


Figure 5.6: Visualization of transformation frames in Rviz [58].

The transformation between “base_link” and “oak-d_frame” is composed by a translation in z and x axis and a rotation in y axis relative to the “base_link”. In turn, the transformation from “base_link” to “livox_link” also represents a translation in z and x axis and a rotation in y axis. On the other hand, the transformation from “base_link” to “velarray_link”, only presents a translation in z axis and a rotation in y axis. Knowing the position of each frame relative to the “base_link”

it is possible to determine the transformation matrix from the camera frame (“oak-d_frame”) to each LiDAR frame (“livox_link” and “velarray_link”). This transformation matrix is the extrinsic matrix between the sensors. Equation 5.6 shows the extrinsic transformation matrix between “livox_link” and “oak-d_frame” and equation 5.7 shows the extrinsic transformation matrix between “velarray_link” and “oak-d_frame”.

Being 19° the pitch angle between the Livox and camera frames, ϕ the rotation angle in y axis and t_x, t_y, t_z the translation in x, y, z axis, the transformation matrix is composed by a rotation in y axis followed by a translation (equation 5.6).

$$\begin{pmatrix} \cos \phi & 0 & \sin \phi & t_x \\ 0 & 1 & 0 & t_y \\ -\sin \phi & 0 & \cos \phi & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos(19) & 0 & -\sin(19) & -0.050 \\ 0 & 1 & 0 & 0.000 \\ \sin(19) & 0 & \cos(19) & 1.175 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (5.6)$$

The transformation from the Velarray to the camera has a similar transformation as the Livox to the camera, with a different rotation in y , 15° (equation 5.7).

$$\begin{pmatrix} \cos \phi & 0 & \sin \phi & t_x \\ 0 & 1 & 0 & t_y \\ -\sin \phi & 0 & \cos \phi & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos(10) & 0 & -\sin(10) & 0.000 \\ 0 & 1 & 0 & 0.000 \\ \sin(10) & 0 & \cos(10) & 0.980 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (5.7)$$

5.4 LiDAR Point Cloud and RGB Camera Image Fusion

As explained in Chapter 4, the system goal is to process data from both sensors (LiDAR and Camera) and project an RGB point cloud from image and point cloud point fusion. To compute these procedures it was imperative to obtain data from the sensors in a real environment.

To make this possible, the Acrobat platform, the robot presented in Figure 5.1, was transported to the greenhouses. The environment presented in Section 3.1, was collected data from the three sensors (OAK-D camera, Livox Mid-70 and Velarray M1600). The data was recorded using *rosvbag*, a tool from 1920x1080 resolution Robot Operating System (ROS) [59], which allows to record in real-time all the parameters and data given by the system.

5.4.1 Sensors Data

The data from all sensors, saved using the *rosvbag* tool, has been recorded along with the transformations concerning the robot and the world, which allows, not

only to fuse the data itself but also representing it in the world frame, obtaining a 3D projection of the real environment.

OAK-D RGB Camera

The image from the RGB camera was recorded in two different resolutions, at 1920x1080 and 416x416, the last one as a cropped and resized image from the higher resolution (1920x1080). The frame rate of image acquisition, at 416x416 resolution, is 9 Hz, which means 9 frames per second. The frame rate of image acquisition, at 1920x1080 resolution, is 2 Hz, which means 2 frames per second. Figure 5.7 shows both images at scale (same pixel size), for visual comparison.

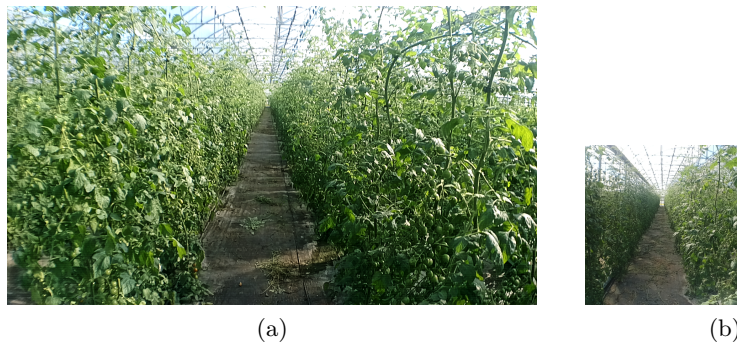


Figure 5.7: 1920x1080 resolution RGB image (a); 416x416 resolution RGB image (b).

This difference justifies the use of the cropped and resized resolution, as well as the use of this image for Object Detection, since the model implemented requires 416x416 resolution, as explained in Section 5.5.

Livox

Point Cloud data from the Livox was acquired at 10 Hz, with a fixed points density of 9984 points per point cloud. The Livox driver fills the empty point spaces with NaN values, thus, it maintains the same point density in the data array. However, this data is different from the real points acquisition. Figure 5.8 presents one point cloud acquired using Livox.

Velarray

Point cloud data from the Velarray, as from the Livox, was acquired at 10 Hz, with an average point density of 30000 points per point cloud. The Velarray driver has a dynamic allocation of the points in the data array, computing its size accordingly. Figure 5.9 presents one point cloud acquired using the Velarray.

From Figure 5.9 it is possible to observe that the Velarray point cloud has way more information than the Livox point cloud. Therefore, since the frame rate is 10

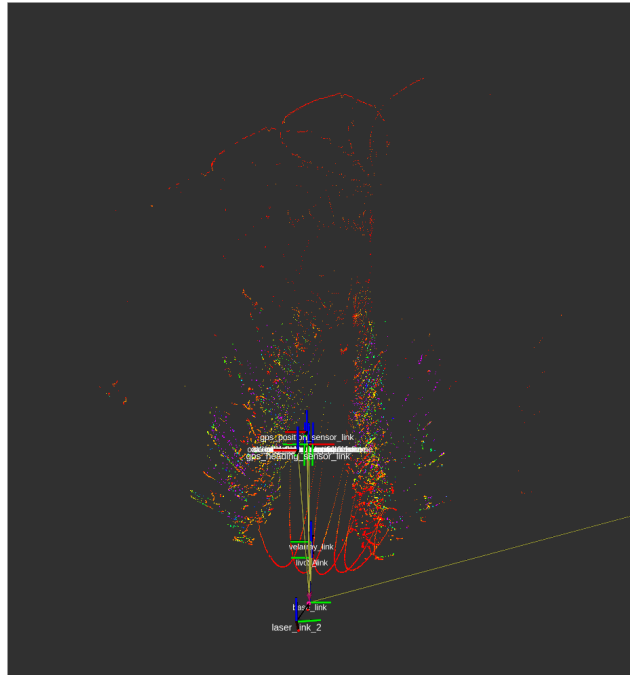


Figure 5.8: Projected point cloud on “livox_link” visualized in Rviz.

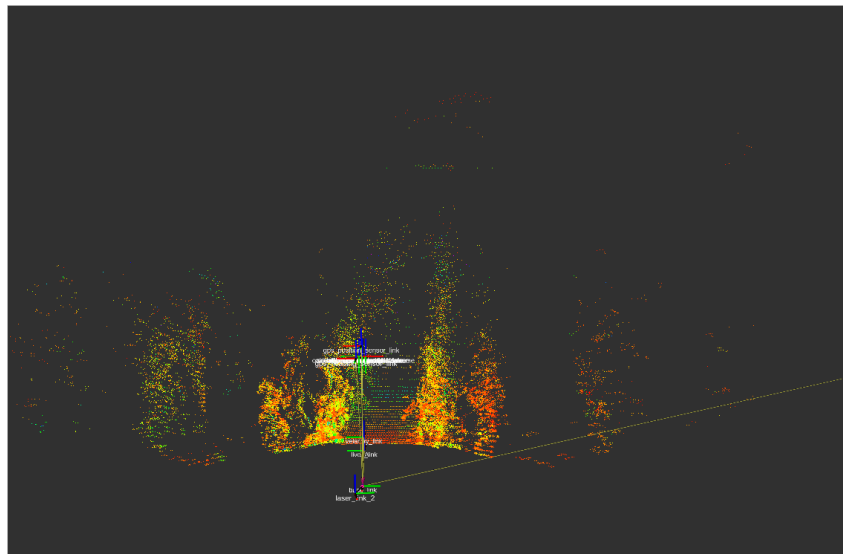


Figure 5.9: Projected point cloud on “velarray_link” visualized in Rviz.

Hz, the Livox point clouds were stacked per second, which means ten point clouds data in one point cloud, with a point density of 99840 points. Figure 5.10 presents the stacked point cloud from Livox. In Figure 5.10 it is possible to visualize the robot transforms frames in the middle of a plant aisle, and the top of the greenhouse (in red) in the top of the point cloud.

Although the stacked point cloud seems to be visually correct, was not accumulated based on the movement of the robot over one second of acquisition time,

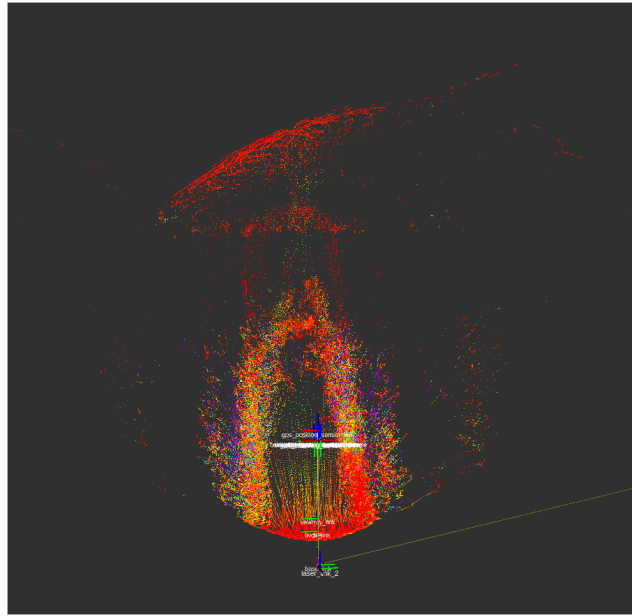


Figure 5.10: Projected accumulated point cloud on “livox_link” visualized in Rviz.

corresponding to the 10 frames. Thus, there is an error in the relative position of the points to the robot’s “base_link” frame. On the other hand, the image acquired every 10 frames is only associated with the last acquisition, so the sensor fusion presented in Section would not be reliable. Thus, it was decided to keep only one frame per second, as shown in Figure 5.8, for input in the sensor fusion module.

5.4.2 Sensors Fusion

The point cloud obtained initially was set for the LiDAR frame. However, to project the points into the RGB image from the camera, the points were transformed. Therefore, by computing the new coordinates for the camera coordinate system (3D) using equation 5.6, for the transformation from Livox frame, and 5.7, for the extrinsic transformation from Velarray frame, it was possible to determine the camera points location.

After determining the camera point’s location, they could be converted to the image coordinate system (2D) and create a range image from the calculated range of each point on the point cloud. This was possible through the point conversion presented by equation 4.9.

The camera angular field of view AFOV was determined from those calculations, in which the inliers would be located. Equation 5.10 represents one way to determine the AFOV from the focal length f and the sensor size H , considering the sensor size as a computation between pixel size ρ and number of pixels in the image n (equation 5.8). However, since the focal length f_x and f_y from the intrinsic matrix

(5.2) is represented in pixel, the real focal length in meters is computed according to equation 5.9.

$$H = \rho * n \quad (5.8)$$

Considering f_x and f_y as f_p :

$$f = f_p * n \quad (5.9)$$

$$AFOV = 2 * \arctan\left(\frac{H}{2f}\right) \quad (5.10)$$

The illustration shown in Figure 5.11 displays the relation between focal length and angular field of view.

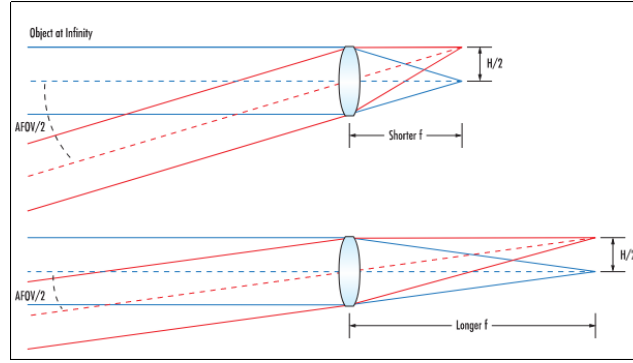


Figure 5.11: Illustration of the relation between focal length and AFOV [60].

Considering the acquired images, the HFOV and VFOV of the camera, coupled with the 120° wide angle lens for the 416×416 and 1920×1080 resolution RGB images, are given by equations 5.11 and 5.12 for the HFOV, and 5.13 and 5.14 to obtain the VFOV value. Although the resolution for the smaller image is 416×416 , the focal length has to be considered before the re-scale from the 1080×1080 cropped image. Therefore, considering the initial pixel size of $1.550 \mu m$, multiplied by the re-scale factor between resolutions, the computed pixel size is $4.024 \mu m$.

$$HFOV_{416} = 2 * \arctan\left(\frac{416 * 4.024 * 10^{-6}}{2 * (460.015656 * 4.024 + 10^{-6})}\right) = 48.7^\circ \quad (5.11)$$

$$HFOV_{1920} = 2 * \arctan\left(\frac{1920 * 1.55 * 10^{-6}}{2 * (1143.5677 * 1.55 + 10^{-6})}\right) = 80.0^\circ \quad (5.12)$$

$$VFOV_{416} = 2 * \arctan\left(\frac{416 * 4.024 * 10^{-6}}{2 * (463.4635 * 4.024 + 10^{-6})}\right) = 48.3^\circ \quad (5.13)$$

$$VFOV_{1080} = 2 * \arctan\left(\frac{1080 * 1.55 * 10^{-6}}{2 * (1140.0413 * 1.55 + 10^{-6})}\right) = 50.7^\circ \quad (5.14)$$

The f_x and f_y values used in the computation of equations 5.11 to 5.14 are from the OpenCV library calibration. The intrinsic values from the MATLAB camera calibration tool give different FOV values, $(51.6^\circ, 79.5^\circ, 52.2^\circ, 50.2^\circ)$, respectively. For the implementation use case, the difference between both results does not influence the accuracy of the system. Therefore, the OpenCV intrinsic parameters were used for reference.

The range image was then overlaid in the RGB image. Figure 5.12 presents the range image (a) and the fused image (b) from both RGB and range image. The RGB image resource is the 416x416 resolution.

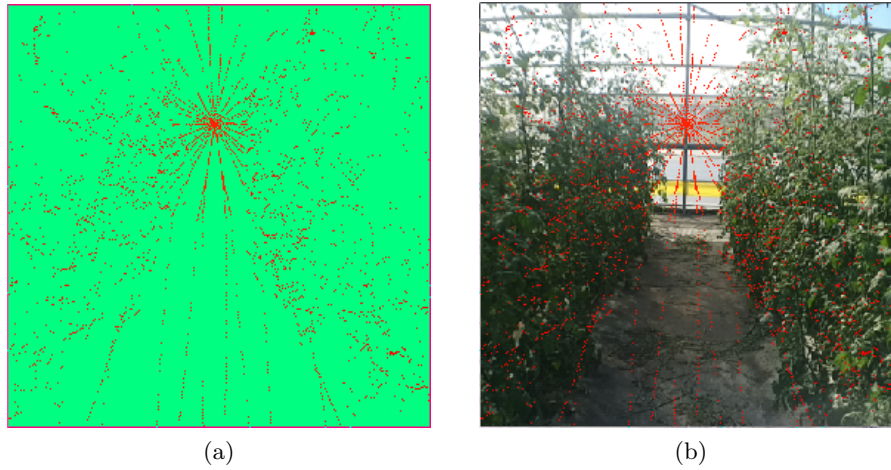


Figure 5.12: Computed range image (a); Fused image from both RGB and range image (b).

After determining the fused image, and assigning the RGB color to each point projected onto the range image, each point presents the following information:

- 1) x position;
- 2) y position;

- 3) **z** position;
- 4) **R** color;
- 5) **G** color;
- 6) **B** color;
- 7) **Intensity**.

This information allows the creation of new data, for example, compute a XYZRGB point cloud, which can give a 3D simulated visualization of the environment. Consequently, the points which were inliers in the camera FOV, as explained in Chapter 4, have been allocated to the new XYZRGB point cloud, creating a simulated view of the robot working environment. The generated RGB point cloud is restricted by the camera HFOV and VFOV, which in this case, for the 416x416 resolution, is 48.7° by 48.3° .

To validate the correct computation of the images fusion and the XYZRGB point cloud, the point's size was increased and both RGB image and point cloud were placed side by side, as depicted in Figure 5.13.

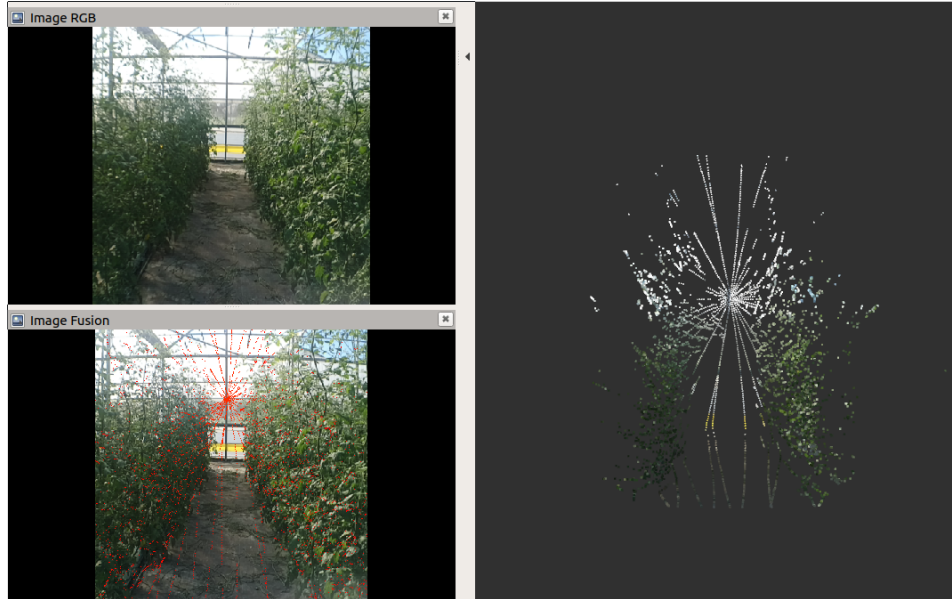


Figure 5.13: Projected XYZRGB cloud, Fused Image, and RGB Image (416x416) visualized in Rviz.

From both representations, it is possible to validate the matching RGB color to the 3D point cloud points.

As compared before, the acquired points from the Velarray point cloud are three times more than the ones from the Mid-70 and the vertical resolution is higher. For that reason, is estimated that the Velarray point cloud presents a better representation of the real environment, especially better detail. A relevant aspect is the

visualization of fruits in the point cloud, which can be useful for the integration of fruits detection on the ground that the robot cannot collide with. Figure 5.14 shows the computed RGB point cloud and Figure 5.15 the detail of the visibility and differentiation of tomatoes, even if they are green, covered by leaves of the plant. The FOV obtained is higher horizontally (80.0°), but lower vertically (32.0°), compared to the Mid-70, due to the Velarray's FHOV.



Figure 5.14: Projected XYZRGB cloud and RGB image (1920x1080) visualized in Rviz.

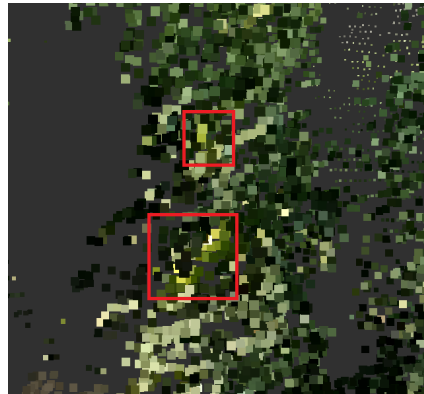


Figure 5.15: Green tomatoes in the middle of the foliage visible in the point cloud (marked in red).

Although the analyses made are qualitative, the only influence of the sensor in the computation of the environment is the FOV, which should be the maximum to represent obstacles, plants, and persons in the scene. For that reason, Mid-70 is the correct choice when the robot is navigating in the aisles, since it gives a higher vertical view of the scene. However, the wider HFOV of the Velarray can be useful in more open areas, like warehouses or main corridors. Overall, the Mid-70 is more suitable for the type of environments found in greenhouses.

5.5 Deep Learning

The deep learning model used was the already trained TinyYoloV4, from Luxonis [61], for 416x416 image resolution. The model was able to perform localization and classification of the objects in the input RGB image. Figure 5.16 shows some of the detected elements present in the scene.



Figure 5.16: Fruits detection (a); Forklift detection (b).

5.6 Robot 3D Modeling

To recreate the real robot operation in the greenhouses, and have a reliable simulation of the environment for system tests and evaluation, the real robot was modeled to be compatible with Gazebo [62] and Rviz. From the SolidWorks [63] model and using the SolidWorks to URDF SW2URDF plugin [64], it is possible to define the links and joints in the 3D CAD model, and associate parts of one assembly to each link and joint.

5.6.1 Model Definition

To model this robot the assembly was divided into five parts: the main body which corresponds to “base_footprint”, the two traction wheels which correspond to two different continuous joints, “traction_left_link and” and “traction_right_link”, and for directional wheel, both direction arms and wheels. The last two, defined by two joints each, “steering_left_link” and “steering_right_link” as revolution joints and limited to rotate from -90° to 90° , and two continuous joints, “rot_steering_left_link” and “rot_steering_right_link”.

The exported model was then uploaded to Gazebo and Rviz and the model matched the existent transforms to the positions of the sensors, as presented in Figure 5.17.

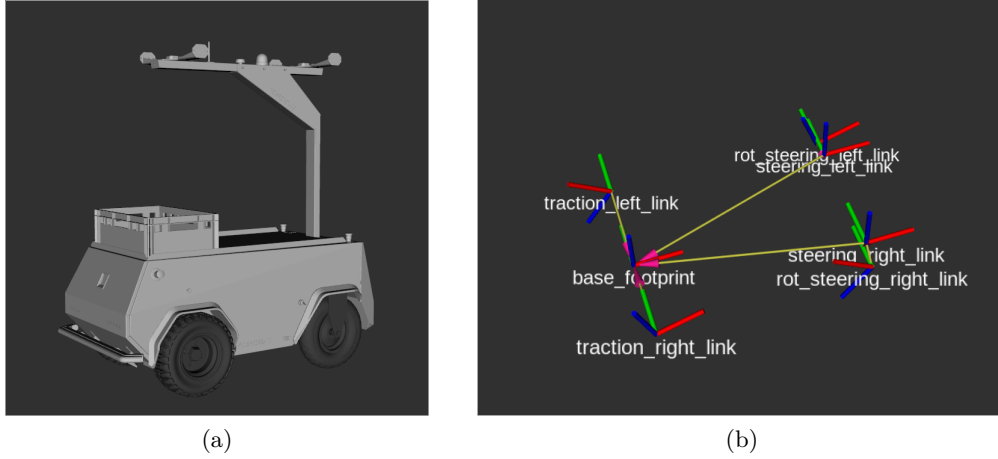


Figure 5.17: Robot model (a); Defined joints and links (b).

Although the defined joints and links are defined in SolidWorks and exported to Gazebo and Rviz the transformation from the “base_link” to the “base_footprint” must be implemented so that the junction of the existing transformations with the transformations of the robot model match.

5.6.2 Model Kinematics

The use of the robot model also implies that it simulates the kinematics of the real robot. Thus a joint controller was implemented to control the joints of the model according to the linear speed of the drive wheels and the angular speed of the steering shaft in the Ackermann system.

The controller receives the reference speeds sent by the speed controller of the real robot and applies the kinematics to each joint, according to the position in a given time space. The robot features an electronic differential controller, as well as an angular relation between wheels representing the Ackermann system. Considering the steering angle as θ , the traction relation r between wheels is described by equation 5.15.

$$r = -0.00035 * |\theta^3| + 0.0026 * |\theta^2| - 0.0773 * |\theta| \quad (5.15)$$

Equation 5.15 was determined based on the robot traction axis length, wheel diameter, and average steering angle of the Ackermann system.

On the other end, to control the directional wheels was determined the relation between the angle of each wheel and the theoretical average angle of the steering system. The average angle of the steering system is estimated by the speed controller on the real robot and sent to the steering system. Figure 5.18 represents the Ackermann kinematics being:

θ_i - Inner wheel angle;

θ_o - Outer wheel angle;

W - Distance between the steering axis of both wheels;

L - Wheelbase of the robot.

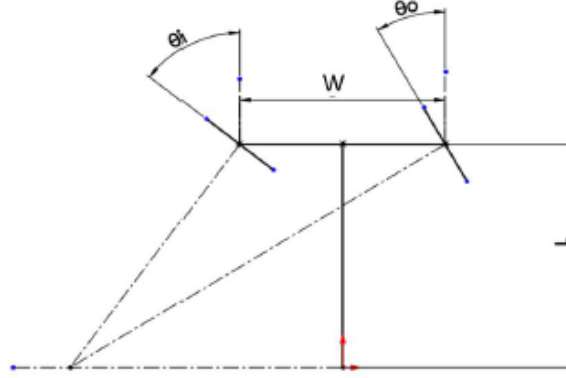


Figure 5.18: Ackermann kinematics [52].

Again, considering θ as the steering angle, equations 5.16 and 5.17, presents the relation between the angle of each wheel, inner θ_i and outer θ_o , and the theoretical average angle θ_m .

$$\theta_i = \operatorname{arccot}[\cot(\theta_m) - \frac{W}{2L}] \quad (5.16)$$

$$\theta_o = \operatorname{arccot}[\cot(\theta_m) + \frac{W}{2L}] \quad (5.17)$$

5.7 Collision Avoidance

Collision avoidance assumes knowledge of the robot's surroundings. Therefore, it is necessary that the information perceived by the robot's sensors is accumulated and grouped according to its importance.

Since this is a dynamic environment, it is not feasible to use a pre-known map to locate the robot and obtain its location according to the distance to known objects in the surrounding environment. The collision avoidance system acts on the trajectory planning, previously implemented in the robot, determining the position of obstacles in a given area.

5.7.1 3D Environment

To obtain a 3D simulation of the environment surrounding the robot, the point cloud obtained by fusing LiDAR with the RGB camera was accumulated over the robot's movement time to make the 3D mapping of the environment. Consequently, to obtain a reliable 3D map of the actual greenhouse scene, the position of the points and the transformation between each point would have to be obtained at each sensor fusion.

To determine the relative position between point clouds and project on the camera fusion frame the accumulated point cloud, the position of the robot on the map at each of the RGB point cloud fusions was determined, and the transformation matrix between positions $\mathbf{T}$ is represented by equation 5.18.

$$T_{xyz} = \begin{pmatrix} R_{3 \times 3} & t_{3 \times 1} \\ 0_{1 \times 3} & 1_{1 \times 1} \end{pmatrix}_{4 \times 4} \quad (5.18)$$

Figure 5.19 illustrates the relation of the relative point cloud transformation obtained by the robot positions relative to the map, given by the RTK system, mentioned in Section 5.1.

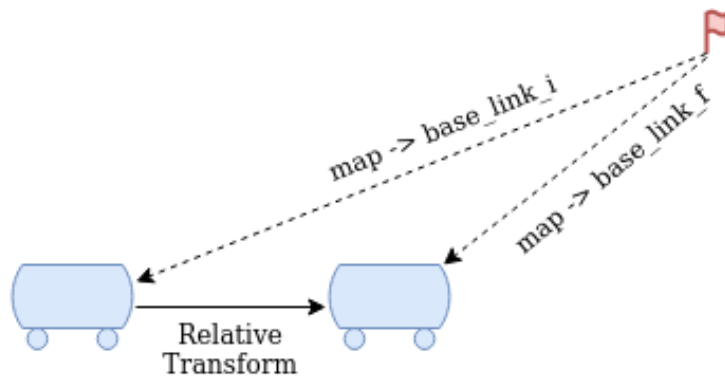


Figure 5.19: Relative transformation obtained by the robot positions relative to the map.

As presented in Figure 5.19, the relative transformation between point clouds is equal to the difference from the robot initial position, “base_link_i”, to the robot second position “base_link_f” from the map (red flag).

From transformation matrix $\mathbf{T}$ and assuming the coordinates of a point from the initial position as X_i, Y_i, Z_i , and the coordinates of a point from the final position as X_f, Y_f, Z_f , equation 5.19 computes the relative transform position.

$$\begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix} = \begin{pmatrix} R_{3 \times 3} & t_{3 \times 1} \\ 0_{1 \times 3} & 1_{1 \times 1} \end{pmatrix}_{4 \times 4} \begin{pmatrix} X_f \\ Y_f \\ Z_f \\ 1 \end{pmatrix} \quad (5.19)$$

The accumulation of each point cloud to the robot frame is computed by applying the relative transformation to the existent stacked cloud at each iteration and accumulating the new RGB point cloud. Consequently, over time, and accordingly to robot locomotion, the 3D map of the environment is formed. Figure 5.20 depicts a point cloud accumulation of the RGB point cloud fused with the Mid-70 LiDAR in which the robot is navigating.

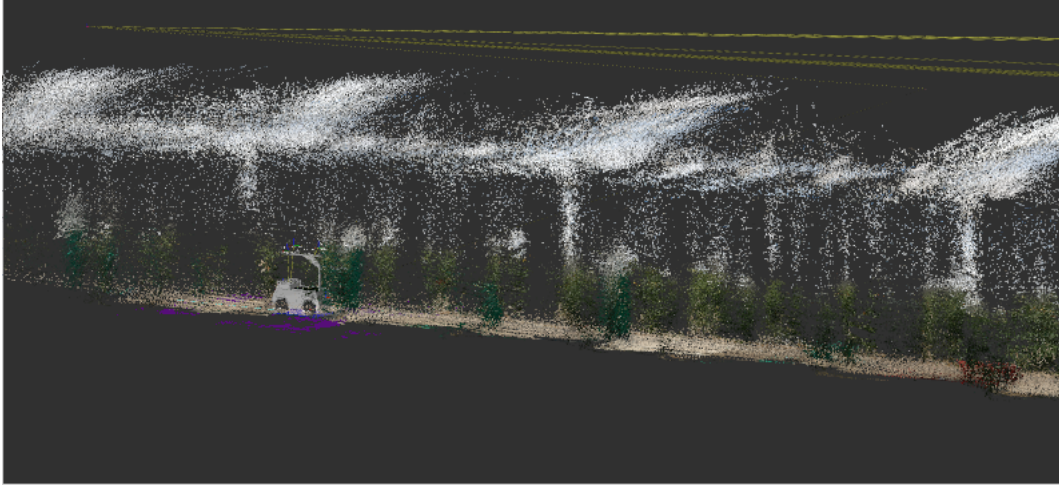


Figure 5.20: 3D mapping from Mid-70 LiDAR and OAK-D kit RGB camera image along a greenhouse main corridor.

5.7.2 Collision Map

The collision map, as explained in Section 4.3, represents the projection of the accumulated RGB point cloud. An example is presented in Figure 5.20, in the ground plane ($z = 0$). The robot has a high of approximately 1.65 meters and a ground clearance of 0.10 meters. Hence, the points with a z coordinate, relative to the map, above 1.65 and below 0.10 were not projected, due to the impossibility of collision in that 3D space. The objects under 0.10 meters are considered surmountable.

The built map is divided into three grid maps, as explained in Section 4.3 and has 8 meters of width and height, which implies 4 meters in the negative and positive

direction of x and y axis, to cover the robot curvature angle and prevent collisions while turning around tomatoes rows. The **obstacle map** is where point cloud points, corresponding to the environment, between 0.10 and 1.65 meters in the z axis, are projected. The map resolution is 5 centimeters per square, which is sufficient for collision detection. The points in the RGB point cloud outside the x and y map limits were removed from the stacked point cloud, for efficiency, since they do not need to be projected into the ($z = 0$) plane.

Figure 5.21 (b) presents an obstacle map projection from the robot navigation in one of the greenhouses main corridor. It is visible the cut point cloud at the bottom of the map.

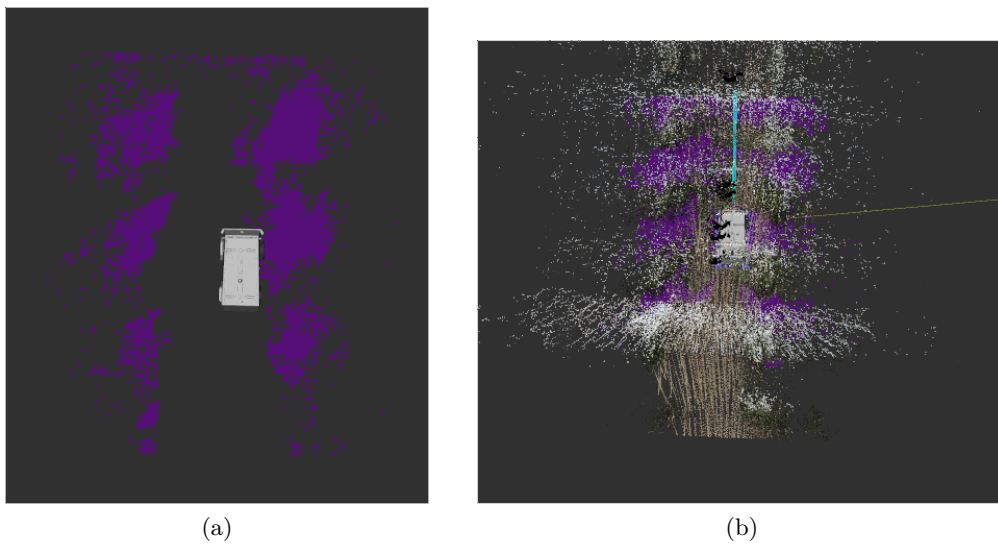


Figure 5.21: Obstacle Map projection (a); Obstacle Map overlaid by RGB point cloud (b).

The footprint map, or robot map, represents the robot footprint in the 8 by 8 meters collision map, considering the wheel curve radius as an extra robot footprint to be able to predict collisions with the robot wheel when cornering. The trajectory map represents the robot trajectory projected on a grid map. Figure 5.22 shows the obstacle map projection in the system after defining the footprint map (under the robot model in blue) and the trajectory map (in light blue).

On the left side of the map are lighter purple or violet squares, which represent a detected object by the object detection model implemented in the system.

5.7.3 Collision Detection

The detection of collisions is integrated by the system as an evaluation of the real-time collision map. As the map is updated, the system detects possible collisions with detected objects and defines the type of objects with which it can collide.

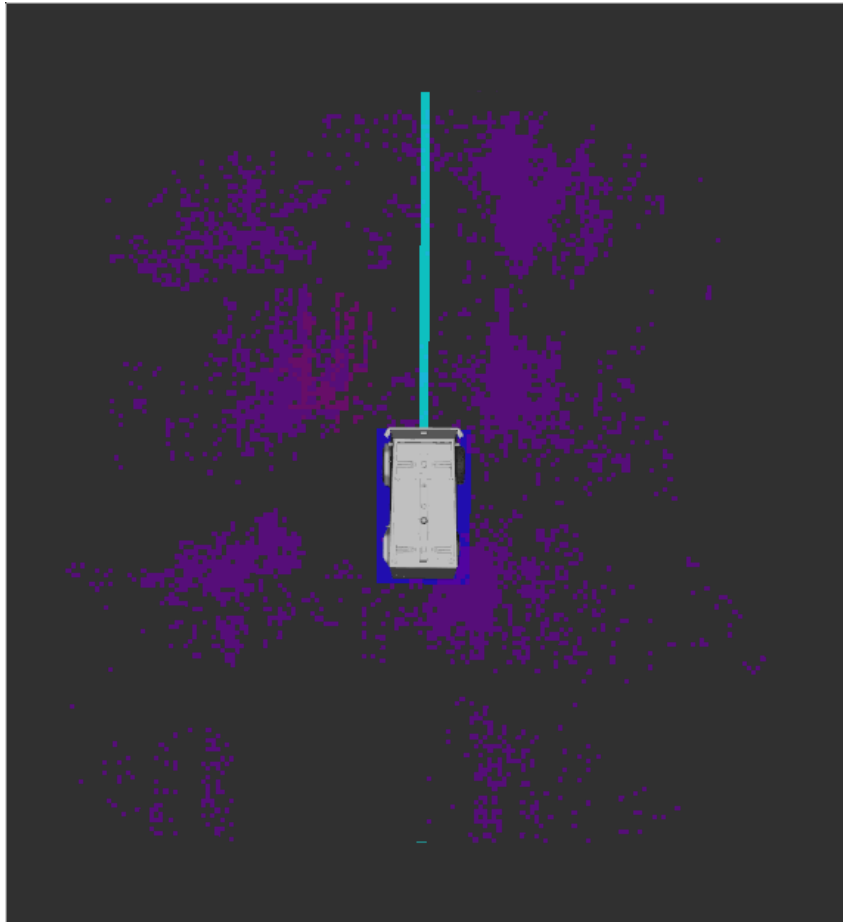


Figure 5.22: Collision map composed by the 3 layers projection.

As explained in Section 4.5 before the collision evaluation some parameters must be defined.

First, is important to define the robot's collision areas which, by its geometry and also the footprint projected before, are four:

1. Front;
2. Left;
3. Right;
4. Back.

The objects that can be interpreted as possible collision obstacles should also be defined. For this implementation there were defined four types of obstacles:

1. Person;
2. Plant;
3. Fruits;

4. Other.

The definition “other”, represents all the other objects that are not intervenients in the robot operation, but can be present in the scene.

Once the possible objects are determined, the evaluation of collision is performed predicting the collision of the robot footprint with the projection of the obstacle in the map. The footprint is projected in advance within its real pose on the map to ensure the validation of the collision. Figure 5.23 presents the updated collision map projecting the advanced footprint to enable collision prediction.

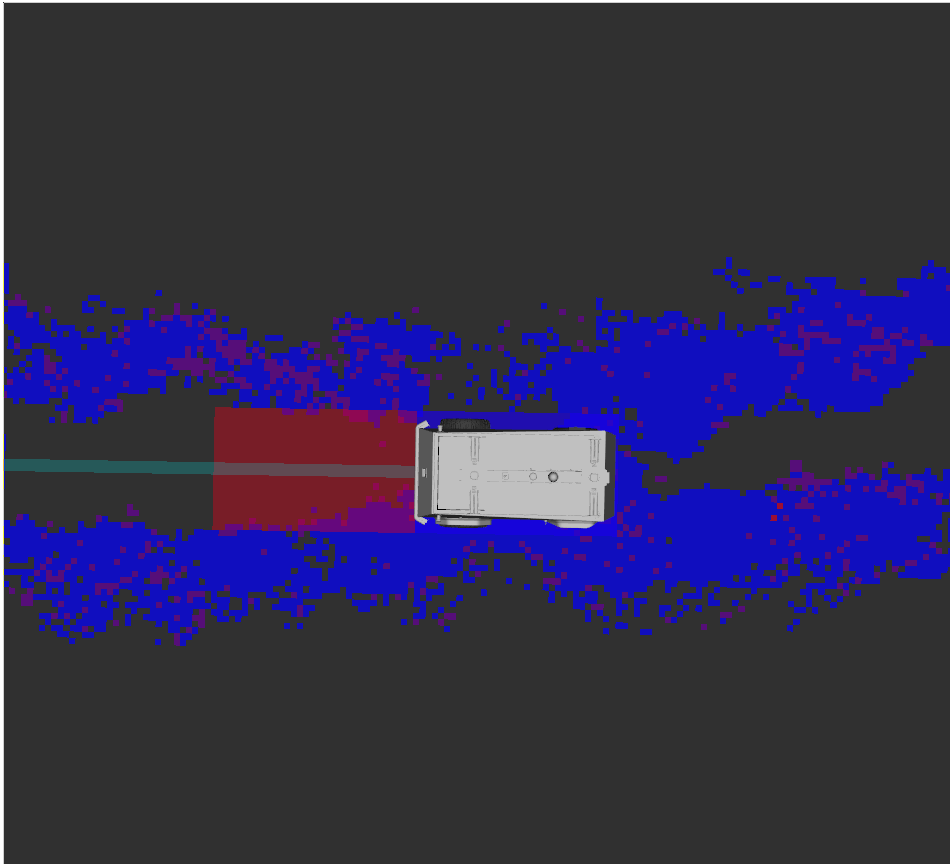


Figure 5.23: Updated collision map with advanced collision projection in a plants aisle.

The map presented in Figure 5.23 shows blue squares projected along with the purple squares. Blue squares represent the projection of the plants. To distinguish the plants, and update their projection in the map, it was defined an RGB window prioritizing the green colors of the plants. The points with the RGB color inside the defined window are projected as blue squares. The fruit’s colors (in red) are also different from the rest of the projection, and its position is determined by the localization in the image through the object detection model (two red squares in the left bottom row of plants).

To determine the presence of people on the scene, the collision map projects red alerts in the four corners of the map, warning that it has entered a hazard zone, as explained in Chapter 6.

Defined the map and all the features to determine the type of object in the scene, it is possible to evaluate the collision.

5.7.4 Collision Evaluation

For collision evaluation, was implemented a detector that overlays all map layers, and evaluates the intersections of the obstacle map with the collision areas of the advanced footprint, in the footprint map.

The number of squares that intercept from the obstacle map with the footprint collision areas is calculated for each area. The amount of squares, of each obstacle, projected in the collision area of the advanced footprint, determines the type of obstacle.

A decision algorithm determines in which area the collision is occurring and the type of obstacle determining the type of collision. The algorithm evaluation sequence is illustrated by the flow chart in Figure 5.24.

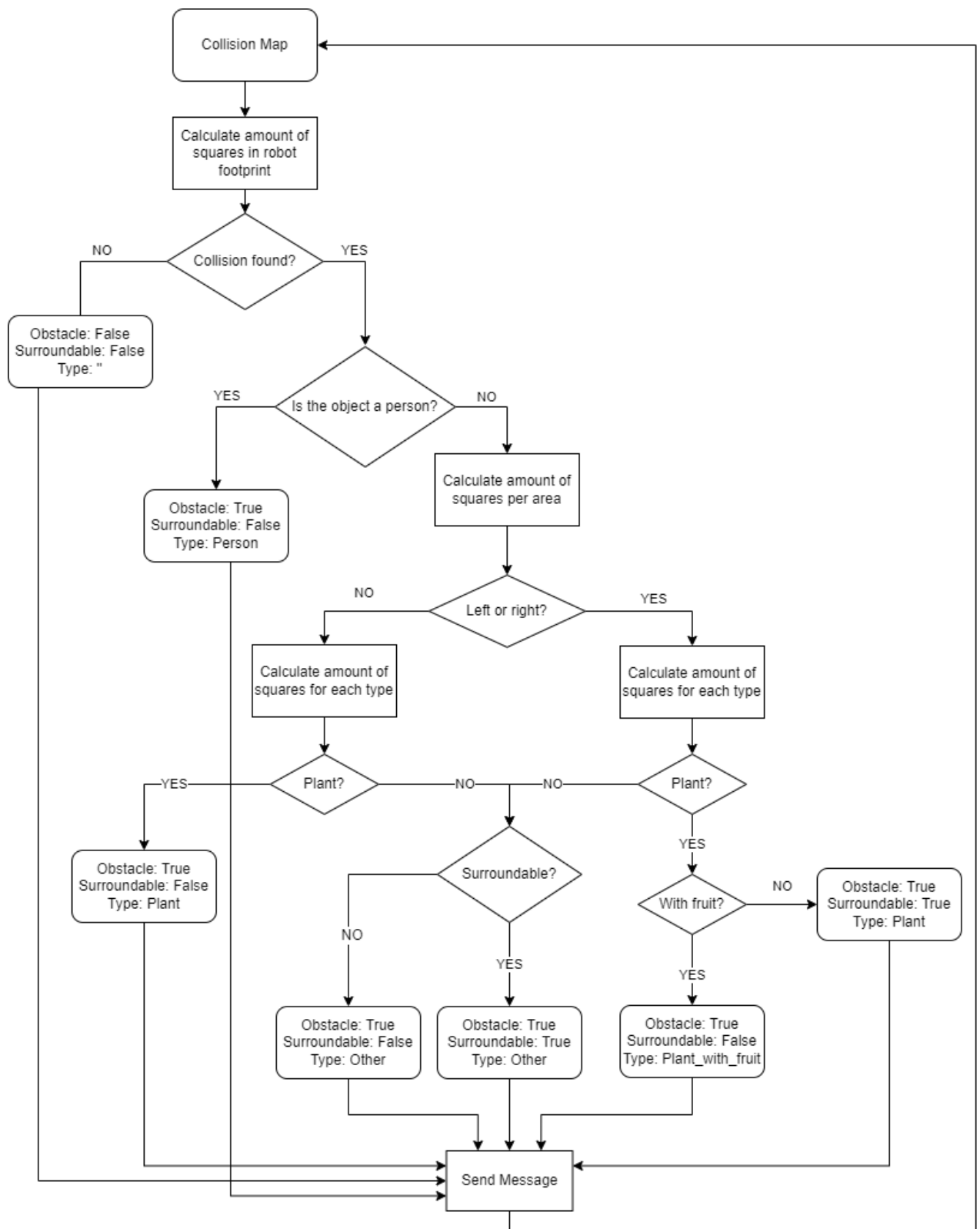


Figure 5.24: Flow chart of the decision algorithm for collision evaluation.

5.8 Collision State Machine

The state machine is composed of five main states, as explained in Chapter 3. To implement the state machine, was used **Smach** [65], a ROS package designed to compose and coordinate complex robotics tasks. Smach allows the integration of different types of state levels, giving introspection about the tasks being performed by the robot. Figure 5.25 shows the upper level of the state machine.

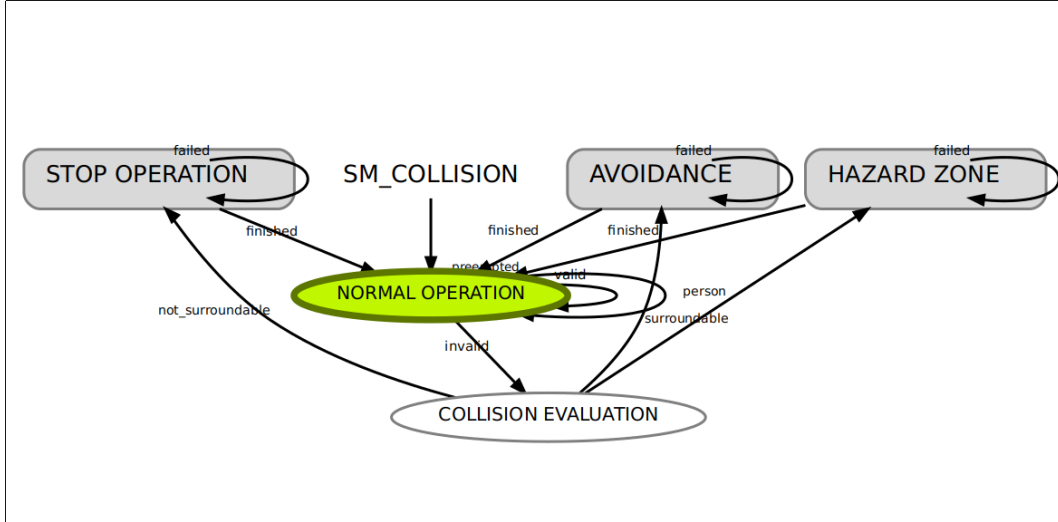


Figure 5.25: Collision state machine upper level.

The Normal Operation state is the start state, defined as a monitor state in the smach architecture, which means that it maintains its state till it is invalid and a transition is triggered. To transit from Normal Operation to Collision Evaluation an object collision, in the collision map, has to be detected. Once the collision is detected the machine stays in the Collision Evaluation state till the obstacle is defined. The obstacle can be defined as a *person*, *surroundable* and *not_surroundable*. If the obstacle is classified as a person, a state transition is triggered and the robot enters a Hazard Zone, the state name indicates.

The Hazard Zone acts on the robot control parameters and defines its “MaxLinearSpeed” equals to 0.3 m/s , which corresponds to the maximum velocity allowed by ISO 3691-4 when an AMR enters a hazard zone. In case the person is no longer present in the collision map, is triggered a transition state to the Normal Operation state.

Next, if the obstacle is classified as surroundable the state machine enters the Avoidance state, which performs two types of behaviours, obstacle avoidance, and fruit avoidance. When the operation is finished a state transition to Normal Operation is triggered.

The Stop Operation state is triggered if an obstacle is not surroundable. Therefore, the robot cannot continue the current operation and the “MaxLinearSpeed”

parameter is set to 0 m/s . Being the obstacle removed, or the robot manually controlled to avoid, it triggers a state transition to Normal Operation.

5.9 Remarks

The hardware was defined and tested to evaluate which LiDAR sensor was best for sensor fusion of the environment with RGB camera image, in the specified conditions. The sensor fusion was achieved, by implementing the 3D to 2D transformations presented in Section 4.1. The detections by the object detection model were projected onto the XYZRGB point cloud and the position of the detected objects was estimated.

The collision map was determined by projecting the XYZRGB point cloud onto the 2D ground plane ($z = 0$) to obtain the most important discretization of the system. This discretization projection of the collision map allows for determining the collisions of the robot with the environment or, known objects.

Moreover, the collision operations and the collision detection are managed by the implemented state machine, which allows a simple real-time inspection of the system.

Chapter 6

Results

This chapter demonstrates the final state of the implemented system, as well as the tests that were carried out with the available data to verify and evaluate its functioning.

To evaluate the system it is necessary to test all of its states. Thus, based on the data previously obtained in a real environment in the greenhouses, three situations were admitted that made it possible to verify the system's reaction:

1. Warehouse navigation;
2. Plant aisle/corridor navigation;
3. Persons in the environment;

6.1 Warehouse Navigation

Navigation in the warehouse always starts from the robot station point and stops at the entrance for the greenhouses corridors. The robot localizes itself in space through RTK, since the top is covered with translucent plastic, as the main greenhouses are not a conventional industrial warehouse. However, the metallic structures in the warehouse are bigger and sometimes the localization is not as accurate as it should be, being forced to show some deviation.

As presented in Figure 6.1 (a) the robot starts its operation in a warehouse corridor with two already known objects in front: a box and a forklift.

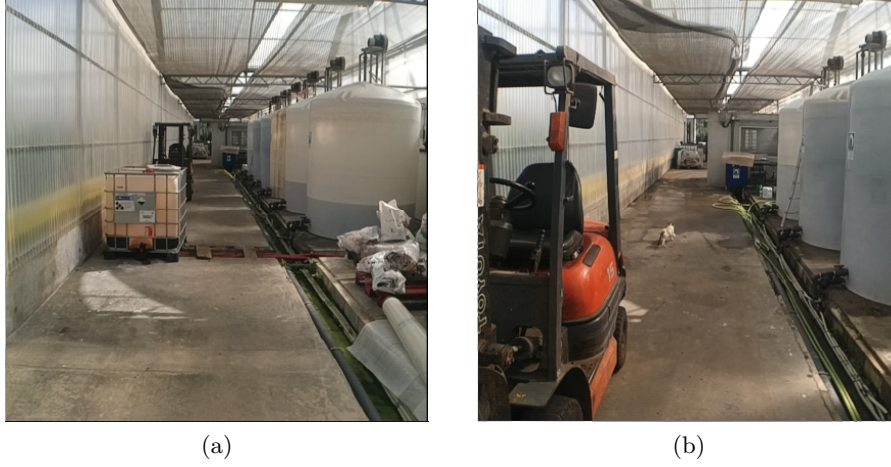


Figure 6.1: Robot starting position (a); Robot final position (b).

After navigating down the corridor and passing the two obstacles (Figure 6.1 (b)), is expected that the object detection model detects these two elements in the scene and the system estimates their position in the 3D space. The object is then projected in the corresponding pose in the collision map. The collision map (Figure 6.2) shows the projected environment after the robot has passed the box and the forklift.

On the right side of the map are visible the phytosanitary silos in purple (unknown obstacles color), and on the left side, the box (bottom) and forklift (top) are projected in two colors, purple and pink. Due to the localization deviation mentioned above, the accumulated point cloud transformation can be affected, since it uses the robot's pose to accumulate the points, which creates outlier points in the forklift and box detection, projecting them in purple instead of pink. To prevent the system from suffering from these localization deviations, and knowing that the dynamic objects are detected by the deep learning model, the unknown obstacles (purple squares) were ignored when at close distance to known obstacles (pink squares). This only applies to warehouse navigation.

Consequently, the collision state machine maintains the Normal Operation state, as long as the advanced robot footprint does not get overlaid by pink squares.

6.2 Plant Aisle Navigation

The Plant aisles surround the robot on both sides, with leaves and fruits. Therefore, it is important to avoid the fruits or, not colliding directly, since the robot was built to not damage the plants if it collides laterally with them.

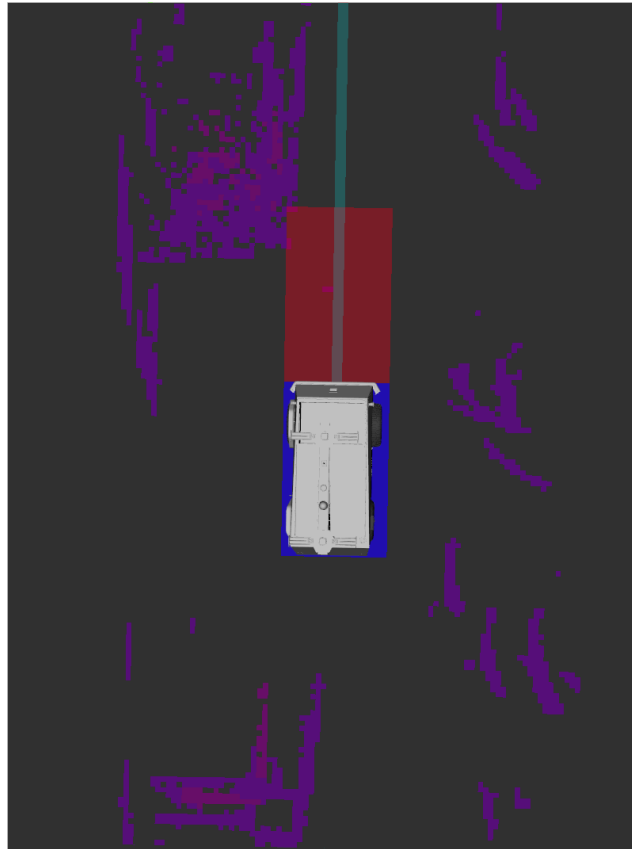


Figure 6.2: Collision map projection in the warehouse corridor.

6.2.1 Plants and Fruits

As mentioned in Section 5.8 the leaves are represented in the collision map in blue and the fruits in red to distinguish them from the other objects in purple. Figure 6.8 shows the collision map with plants from both sides and some red squares in the top left corners. These squares are detections of fruits projected on the map which the robot interprets as obstacles.

These operations are established in parallel by the state machine which triggers a state transition from **Normal Operation** to **Avoidance**. In Figure 6.9 is possible to observe that the state Avoidance has two concurrent states inside, which, in this case, triggers the **Fruit_Avoidance**. The Fruit_Avoidance implemented in the system determines the proximity of the fruits to the advanced footprint and projects the ideal trajectory (in red) in the collision map for visualization of the navigation correction, as shown in Figure 6.8.

The Avoidance state also shows a **check** state inside, the state machine, which remains in this state to check if there are more obstacles of the same “fruits” type and do not transits states unnecessarily.

The system actuates as supposed and returns to the Normal Operation states after the Avoidance state.

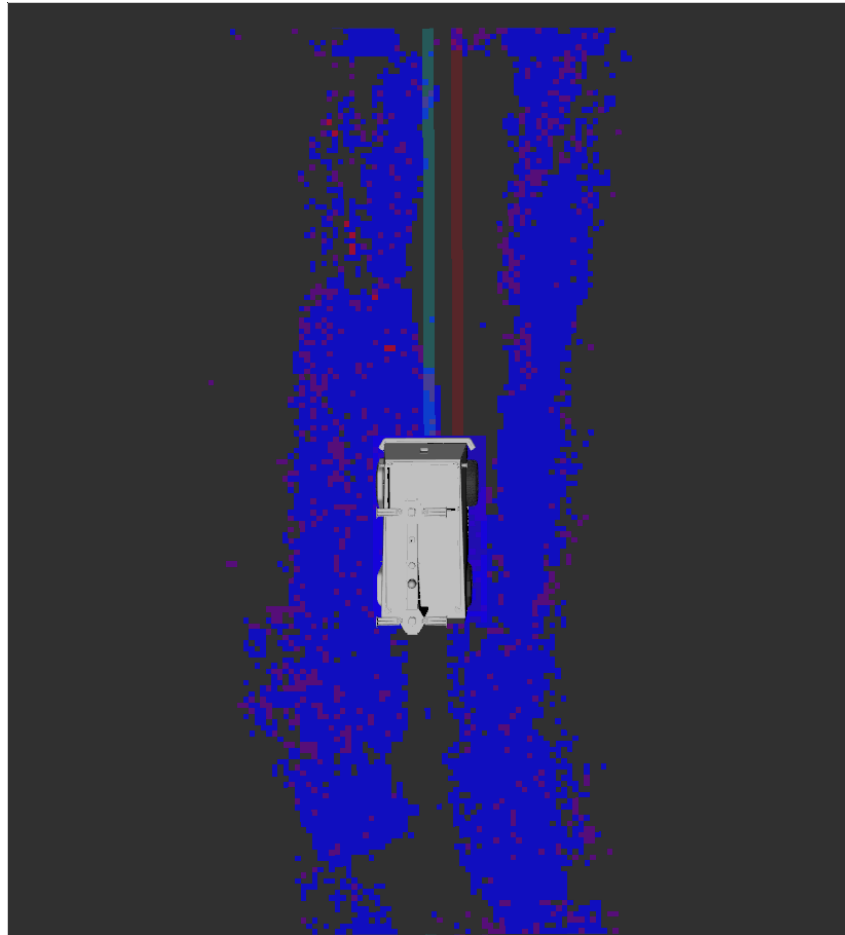


Figure 6.3: Collision map projection in one plant aisle.

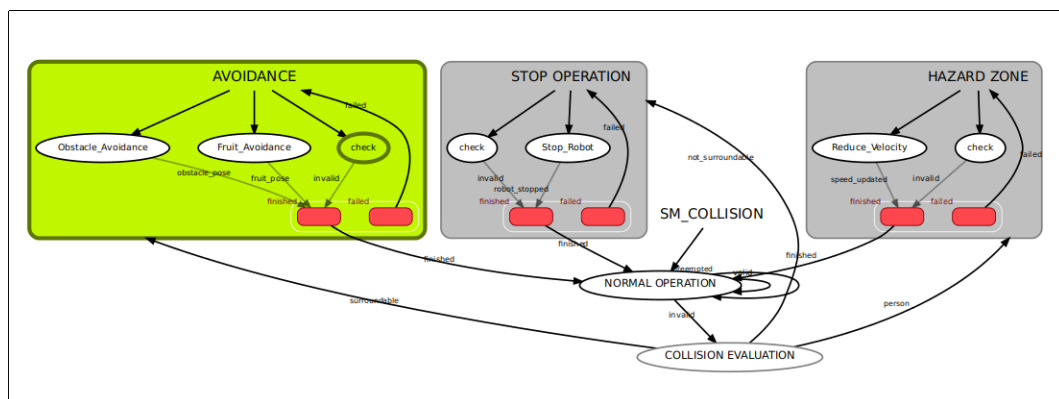


Figure 6.4: State machine transition to Avoidance state.

6.2.2 Other Obstacles

Aligned with the collision avoidance of plants and fruits, also the detection of “other” obstacles in the plant aisles is important and could result in stop situations by the robot.

Figure 6.5 shows one of that situations. The advanced footprint (in red) is

overlaid by obstacle points in purple and blue in its front, right, left, and back. Therefore, the robot navigation cannot estimate the next robot's position, and the robot stops in a blocked situation triggering the state transition to **Stop Operation** in the state machine.

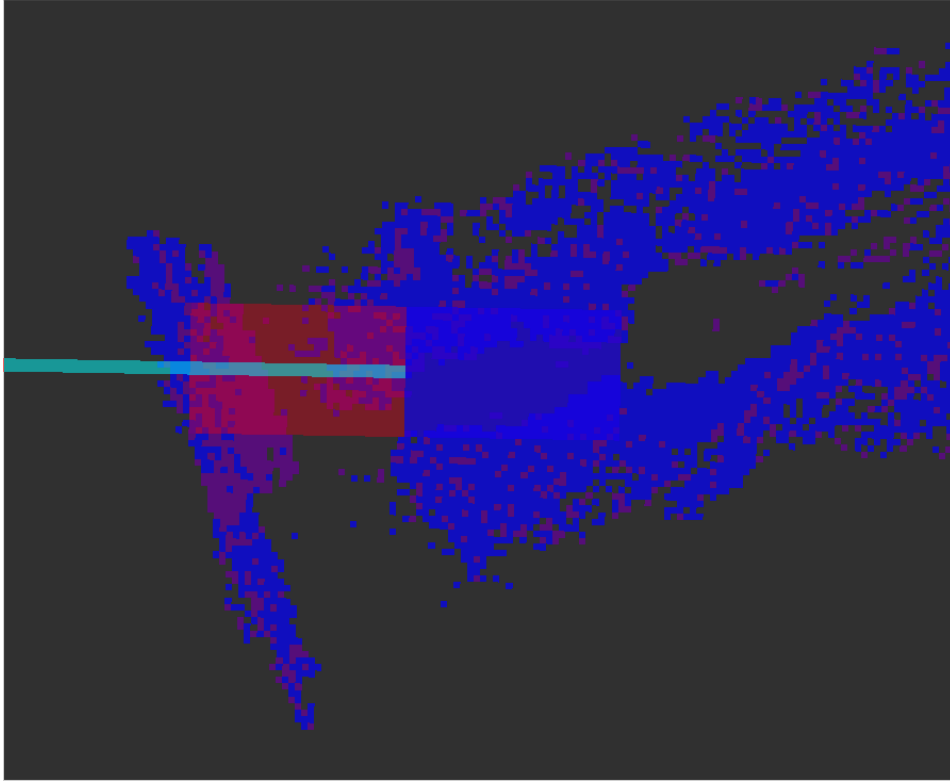


Figure 6.5: Collision map projection of a front obstacle creating a block situation.

The state machine triggers a state transition to **Stop_Robot** and the robot navigation control receives a new value for parameter “MaxLinearSpeed”, 0 m/s . Therefore, it transits inside the state to the **check** state to verify if this condition is still defined in the collision map. The check state also sends a message to the lights control, and they turn red. When out of the Stop Operation the lights turn back to their initial color (usually white). This visual sign must be present in the AMR according to ISO 3691-4 method d). Normal Operation states transition is then triggered after the obstacle is removed or the robot is controlled manually.

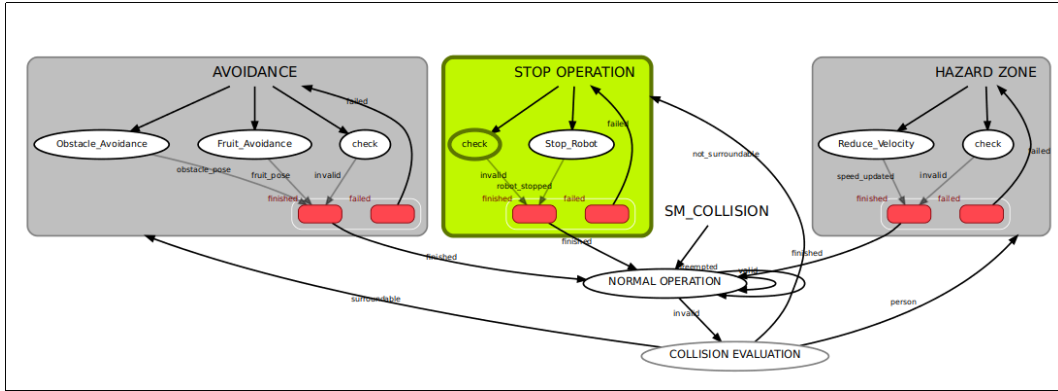


Figure 6.6: State machine transition to Stop Operation state.

6.3 Persons Detection

To test the reaction of the system to the presence of a person in the scene, it was used data from a plant aisle with a person in the middle of the aisle in front of the robot, as shown in Figure 6.7. The person is detected by the object detection model and its localization is stored.



Figure 6.7: Detection of a person in plant aisles.

The person's position is not projected in the collision map since over time it can have different positions and the accumulated point cloud stores the person's position many times. For that reason, the system is prepared to discard a person's position after storing it to ensure a clean and reliable collision map. Figure 6.8 presents the map when detected the same person of Figure 6.7 and no pink squares are projected.

The map projects four groups, of red squares, in the map corners which indicate a hazard zone, meaning the presence of a person near the robot. This projection triggers a state transition to **Hazard Zone** in the state machine. The Hazard Zone

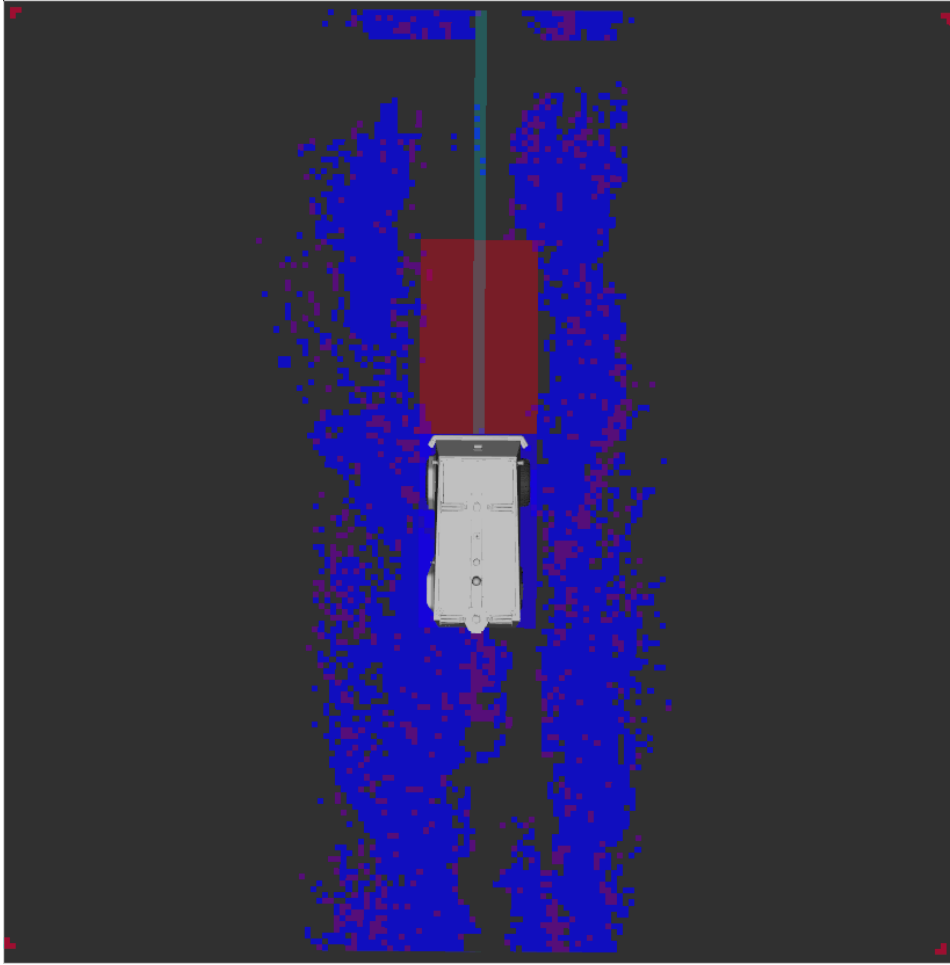


Figure 6.8: Collision map after person detection.

state trigger its internal action **Reduce_Velocity**, which sets a value of 0.3 m/s for parameter “MaxLinearSpeed”. This velocity value is stated by ISO 3691-4 regarding operating zones, which defines this velocity value for operating hazard zones.

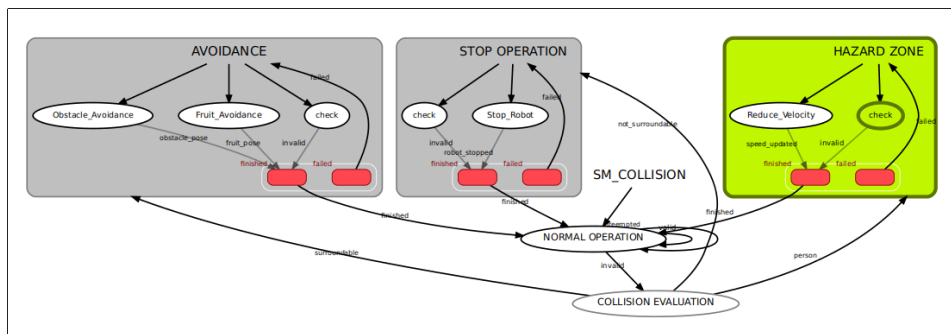


Figure 6.9: Hazard Zone state transition.

The system behaviour corresponds to the expected. Although the person’s position is not projected on the map, its accumulation in the point cloud can cause

false interpretations of the person and extend the Hazard Zone state. In terms of safety, it is not harmful, since it prolongs the safety state. However, it reduces the efficiency of the robot, since it will have to operate at a lower speed while in that state.

Chapter 7

Conclusions

Accordingly, to the results, the proposed approach was achieved, the system was able to recreate the environment surrounding the robot in a 3D scene, by fusing the two sensors LiDAR and RGB camera. The objects in the scene were determined and located in 2D through object detection and associated with the RGB point cloud of the 3D scene. Furthermore, the collision map projection has been created plotting the known and unknown obstacles, as well as people, plants, and fruits, the latter three being the main agents of the environment in the robot's operation.

In terms of determining collisions, the system was able to isolate possible collision situations and eliminate false collisions such as discarding lateral collisions with plants. On the other hand, the operation conditioning, through state transition after collision prediction, revealed that sometimes the robot transitions to verify false collisions. This behavior may reduce the efficiency of the system in very populated environments, however, it increases its safety and demonstrates compliance with the stipulated standards.

Overall the system behaves as supposed, however, it can be enhanced to be more efficient and precise as well as presenting other features based on the information acquired by the sensors.

Besides the smart collision avoidance system developed, this work also allowed the production of two contributions to the robotics community: a review article entitled "Safety standards for collision avoidance systems in agricultural robots - A review" accepted at the ROBOT 2022 conference, and an evaluation form to

rate smart collision avoidance systems for agriculture, CASE - Collision Avoidance System Evaluation for agriculture robots [66].

7.1 Future Work

7.1.1 Sensor Fusion

The extrinsic calibration between the LiDAR and the camera could be improved using auto systems calibration, such as ATOM [67] which is more accurate for the determination of the extrinsic parameters and allows for an easier manual correction. Thus, the quality in terms of localization and color association of the RGB point cloud would increase.

7.1.2 Transformations

The transformations acquired in real-time by the RTK system can suffer large deviations. To improve this aspect, techniques, such as Iterative Closest Point (ICP) [68] or similar should be used to determine the accumulated point clouds and obtain a more reliable 3D map.

7.1.3 Deep Learning

The model used is general so a model trained for the greenhouse environment should be used. In this implementation, it was not possible due to the unavailability of image acquisition in all greenhouse circumstances. A semantic segmentation model would be the most appropriate. Therefore, the definition of inliers and outliers by bounding boxes can identify pixels of the image that do not correspond to the object. This model would also allow the determination of the boundaries of the objects in the 3D scene.

7.1.4 Collision Map

The collision map should be improved to estimate the avoidance trajectory with curvature since at this point it only estimates straight-line trajectories. Also, the system is prepared to add precise and complex trajectory planning algorithms to actuate in case of possible collision with suroundable objects. This could be an important feature for future implementations.

References

- [1] NFU, “Written evidence submitted by nfu (ls0045).” Available at <https://committees.parliament.uk/writtenevidence/40079/html/>, 2021. (Last accessed in 22/05/2022). [Cited on page 1]
- [2] H. of Commons, “House of commons: Labour shortages in the food and farming sector. parliamentary copyright house of commons.” Available at <https://committees.parliament.uk/publications/9580/documents/162177/default>, 2022. (Last accessed in 22/05/2022). [Cited on page 1]
- [3] M. De Clercq, A. Vats, and A. Biel, “Agriculture 4.0: The future of farming technology,” *Proceedings of the World Government Summit, Dubai, UAE*, pp. 11–13, 2018. [Cited on page 1]
- [4] F. Rubcke, F. Clausen, and H. Heise, “Autonomous field robots in agriculture: A quantitative analysis of user acceptance according to different agricultural machinery companies.” Available at <http://ageconsearch.umn.edu/record/305587>, 2010. (Last accessed in 22/05/2022). [Cited on page 1]
- [5] R. Kogler, E. Quendler, and J. Boxberger, “Analysis of occupational accidents with agricultural machinery in the period 2008-2010 in austria,” *Safety Science*, vol. 72, pp. 319–328, 2021. [Cited on page 1]
- [6] I. O. for Standardization, “Iso 25119-2:2019 tractors and machinery for agriculture and forestry — safety-related parts of control systems — part 2: Concept phase,” 2019. (Last accessed in 25/05/2022). [Cited on page 1]
- [7] I. O. for Standardization, “Iso 9001 quality management systems — requirements,” 2015. (Last accessed in 25/05/2022). [Cited on page 1]
- [8] I. O. for Standardization, “Iso 3691-4:2020 industrial trucks — safety requirements and verification — part 4: Driverless industrial trucks and their systems,” 2020. (Last accessed in 03/06/2022). [Cited on page 1]
- [9] I. O. for Standardization, “Iso 10218-1:2011 robots and robotic devices — safety requirements for industrial robots — part 1: Robots,” 2011. (Last accessed in 25/05/2022). [Cited on page 1]

-
- [10] I. O. for Standardization, “Iso 13482:2014, robots and robotic devices — safety requirements for personal care robots,” 2014. (Last accessed in 03/06/2022). [Cited on page 2]
 - [11] I. O. for Standardization, “Iso ts 15066:2016, robots and robotic devices — safety requirements for personal care robots,” 2016. (Last accessed in 28/05/2022). [Cited on page 2]
 - [12] G. Hirzinger, A. Albu-Schaffer, M. Hahnle, I. Schaefer, and N. Sporer, “On a new generation of torque controlled light-weight robots,” *IEEE Int. Conf. on Robotics and Automation*, vol. 4, pp. 3356–3363, 2001. [Cited on page 2]
 - [13] R. Weitschat, J. Vogel, S. Lantermann, and H. Hoppner, “End effector airbags to accelerate human-robot collaboration,” *IEEE Int. Conf. on Robotics and Automation*, pp. 2279–2284, 2017. [Cited on page 2]
 - [14] “Robocare: Plataforma robóticas para culturas de precisão inteligentes.” Available at <http://www.horticoladonorte.pt/projecto-cofinanciado-ue.html>, 2020. (Last accessed in 14/8/2022). [Cited on page 2]
 - [15] C. Yuan, X. Liu, X. Hong, and F. Zhang, “Pixel-level extrinsic self calibration of high resolution lidar and camera in targetless environments,” *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7517–7524, 2021. [Cited on pages 3 and 11]
 - [16] G.-J. Horng, Liu, Min-Xiang, Chen, and Chao-Chun, “The smart image recognition mechanism for crop harvesting system in intelligent agriculture,” *IEEE Sensors Journal*, vol. 20, no. 5, pp. 2766–2781, 2020. [Cited on pages 3 and 11]
 - [17] “Robot 2022.” <https://www.iberianroboticsconf.eu/>, 2022. (Last accessed in 27/6/2022). [Cited on page 4]
 - [18] T. Kruse, A. Pandey, R. Alami, and A. Kirsch, “Human-aware robot navigation: A survey,” *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1726–1743, 2013. [Cited on page 7]
 - [19] B. Lefteris, B. Avital, and B. Dionysis, “Safety and ergonomics in human-robot interactive agricultural operations,” *Science Direct*, vol. 200, pp. 55–72, 2020. [Cited on page 7]
 - [20] P. Ratsamee, Y. Mae, K. Kamiyama, M. Horade, M. Kojima, and T. Arai, “Social interactive robot navigation based on human intention analysis from face orientation and human path prediction,” *ROBOMECH Journal*, vol. 2, no. 1, p. 11, 2015. [Cited on page 7]

- [21] E. J. Van Henten, C. Bac, J. Hemming, and Y. Edan, “Robotics in protected cultivation,” *IFAC Proceedings Volumes*, vol. 46, no. 18, pp. 170–177, 2013. [Cited on page 7]
- [22] N. Noguchi, W. J., J. Reid, and Q. Zhang, “Development of a master-slave robot system for farm operations,” *Computers and Electronics in Agriculture*, vol. 44, pp. 1–19, 2004. [Cited on pages ix and 8]
- [23] S. G. Vougioukas, “Reactive trajectory tracking for mobile robots based on non linear model predictive control,” *IEEE Int. Conf. on Robotics and Automation*, pp. 3074–3079, 2017. [Cited on page 8]
- [24] S. Nissimov, J. Goldberger, and V. Alchanatis, “Obstacle detection in a greenhouse environment using the kinect sensor,” *Computers and Electronics in Agriculture*, vol. 113, pp. 104–105, 2015. [Cited on pages ix and 9]
- [25] Z. Zhang, “Microsoft kinect sensor and its effect,” *IEEE MultiMedia*, vol. 19, no. 2, pp. 4–10, 2012. [Cited on page 9]
- [26] R. Giulio, M. Annalisa, R. Raphael, N. Michael, W. Rainer, and R. B. Morten, “Ambient awareness for agricultural robotic vehicles,” *Biosystems Engineering*, vol. 146, pp. 114–132, 2016. [Cited on page 9]
- [27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection.” Available at doi.org/10.48550/arXiv.1506.02640, 2016. (Last accessed in 04/06/2022). [Cited on page 9]
- [28] K. Inoue, S. Igarashi, and K. Imou, “The development of autonomous navigation and obstacle avoidance for a robotic mower using machine vision technique,” *IFAC-PapersOnLine*, vol. 52, no. 30, pp. 173–177, 2019. [Cited on pages ix, 9, and 10]
- [29] Z. Liu, Z. Lü, W. Zheng, W. Zhang, and X. Cheng, “Design of obstacle avoidance controller for agricultural tractor based on ros,” *Int J Agric & Biol Eng*, vol. 12, no. 6, pp. 58–65, 2019. [Cited on pages ix, 9, and 10]
- [30] L. Guan, Y. Chen, G. Wang, and X. Lei, “Real-time vehicle detection framework based on the fusion of lidar and camera,” *Electronics*, vol. 9, no. 3, 2020. [Cited on page 10]
- [31] B. Benet, R. Lenain, and V. Rousseau, “Development of a sensor fusion method for crop row tracking operations,” in *European Conference on Precision Agriculture (ECPA)*, vol. 8 of *Advances in Animal Biosciences: Precision Agriculture (ECPA)*, (Edinburgh, United Kingdom), pp. 583–589, J.A. Taylor, July 2017. [Cited on page 11]

- [32] H. Kang, X. Wang, and C. Chen, “Accurate fruit localisation for robotic harvesting using high resolution lidar-camera fusion,” 2022. [Cited on page 11]
- [33] J. Cui, J. Niu, Z. Ouyang, Y. Q. He, and D. Liu, “Acsc: Automatic calibration for non-repetitive scanning solid-state lidar and camera systems,” *ArXiv*, vol. abs/2011.08516, 2020. [Cited on page 11]
- [34] N. Murali, “Image classification vs semantic segmentation vs instance segmentation..” Available at shorturl.at/almtx, 2021. (Last accessed in 12/9/2022). [Cited on pages ix and 12]
- [35] I. O. for Standardization, “Iso 10218-2:2011 robots and robotic devices — safety requirements for industrial robots — part 2: Robots systems and integration,” 2011. (Last accessed in 28/05/2022). [Cited on page 12]
- [36] J. P. Vasconez, G. A. Kantor, and F. A. Auat Cheein, “Human-robot interaction in agriculture: A survey and current challenges,” *Biosystems Engineering*, vol. 179, pp. 35–48, 2019. [Cited on page 12]
- [37] I. O. for Standardization, “Iso 12100:2010 general principles for design — risk assessment and risk reduction,” 2010. (Last accessed in 28/05/2022). [Cited on page 12]
- [38] “Directive 2006/42/ec new machinery directive.” Available at <https://osha.europa.eu/en/legislation/directives/directive-2006-42-ec-of-the-european-parliament-and-of-the-council>, 2006. (Last accessed in 28/05/2022). [Cited on page 12]
- [39] I. O. for Standardization, “Iso 13849-1:2015 safety of machinery — safety-related parts of control systems — part 1: General principles for design,” 2015. (Last accessed in 28/05/2022). [Cited on page 12]
- [40] I. O. for Standardization, “Iso 11783-1:2017 tractors and machinery for agriculture and forestry — serial control and communications data network — part 1: General standard for mobile data communication,” 2015. (Last accessed in 29/05/2022). [Cited on page 12]
- [41] I. E. Commission, “Iec en 62061:2021 safety of machinery - functional safety of safety-related control systems,” 2021. (Last accessed in 29/05/2022). [Cited on page 12]
- [42] I. E. Commission, “Iec 61496:2012 safety of machinery - electro-sensitive protective equipment - part 1: General requirements and tests,” 2012. (Last accessed in 29/05/2022). [Cited on page 12]

-
- [43] T. Risto, M. Timo, and R. Ari, “An overview of current safety requirements for autonomous machines – review of standards,” *Open Engineering*, vol. 10, no. 1, pp. 665–673, 2020. [Cited on page 15]
- [44] I. O. for Standardization, “Iso 13855:2010 safety of machinery — positioning of safeguards with respect to the approach speeds of parts of the human body,” 2010. (Last accessed in 29/05/2022). [Cited on page 15]
- [45] I. E. Commission, “Iec 61508-1:2010 functional safety of electrical/electronic/programmable electronic safety-related systems - part 1: General requirements,” 2010. (Last accessed in 30/05/2022). [Cited on page 16]
- [46] M. Ingibergsson, P. Schultz, and D. Kraft, “Towards declarative safety rules for perception specification architectures,” *International Workshop on Domain-Specific Languages and Models for Robotic Systems*, 2015. [Cited on page 16]
- [47] I. O. for Standardization, “Iso 11783-1:2017 tractors and machinery for agriculture and forestry — serial control and communications data network — part 1: General standard for mobile data communication,” 2017. (Last accessed in 29/05/2022). [Cited on page 16]
- [48] I. O. for Standardization, “Iso 18497:2018 agricultural machinery and tractors — safety of highly automated agricultural machines — principles for design,” 2018. (Last accessed in 02/06/2022). [Cited on page 16]
- [49] I. O. for Standardization, “Iso/cd 3991:2022 agricultural machinery — robotic feed systems — safety,” 2022. (Last accessed in 02/06/2022). [Cited on page 16]
- [50] I. O. for Standardization, “Isofocus smart farming.” Available at [https://www.iso.org/files/live/sites/isoorg/files/news/magazine/ISOfocus%20\(2013-NOW\)/en/2017/ISOfocus_122/ISOfocus_122_EN.pdf](https://www.iso.org/files/live/sites/isoorg/files/news/magazine/ISOfocus%20(2013-NOW)/en/2017/ISOfocus_122/ISOfocus_122_EN.pdf), 2017. (Last accessed in 02/06/2022). [Cited on page 16]
- [51] I. O. for Standardization. Available at <https://www.iso.org/standard/70835.html>, 2022. (Last accessed in 03/06/2022). [Cited on page 16]
- [52] H. Rocha, “Autonomous vehicle/robotic platform for smart and precision agriculture inside greenhouses,” Master’s thesis, Faculdade de Engenharia do Universidade do Porto, Porto, 2022. [Cited on pages x, 39, and 56]
- [53] Luxonis, “Oak-d documentation.” Available at <https://docs.luxonis.com/projects/hardware/en/latest/pages/BW10980AK.html>, 2022. (Last accessed in 22/09/2022). [Cited on pages ix and 42]

-
- [54] Luxonis, “Imx378.” Available at <https://docs.luxonis.com/projects/hardware/en/latest/pages/articles/sensors/imx378.html#imx378>, 2021. (Last accessed in 12/9/2022). [Cited on page 41]
- [55] Luxonis, “Ov9282.” Available at <https://docs.luxonis.com/projects/hardware/en/latest/pages/articles/sensors/ov9282.html#ov9282>, 2021. (Last accessed in 12/8/2022). [Cited on page 41]
- [56] OpenCV, “Camera calibration with opencv..” Available at https://docs.opencv.org/3.4/d4/d94/tutorial_camera_calibration.html, 2022. (Last accessed in 12/8/2022). [Cited on page 43]
- [57] MATLAB, “Matlab camera calibration.” Available at <https://www.mathworks.com/help/vision/camera-calibration.html>, 2022. (Last accessed in 25/09/2022). [Cited on page 44]
- [58] ROS, “Rviz - ros wiki.” Available at <http://wiki.ros.org/rviz>, 2022. (Last accessed in 25/09/2022). [Cited on pages ix and 45]
- [59] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, “Ros: an open-source robot operating system,” in *ICRA workshop on open source software*, vol. 3, p. 5, 2009. [Cited on page 46]
- [60] “Understanding focal length and field of view..” Available at <https://www.edmundoptics.com/knowledge-center/application-notes/imaging/understanding-focal-length-and-field-of-view/>, 2022. (Last accessed in 12/9/2022). [Cited on pages x and 50]
- [61] Luxonis, “Rgb tinyyolov4 decoding on device.” Available at https://docs.luxonis.com/projects/api/en/v2.7.1/samples/tiny_yolo_v4_decoding_on_device/, 2021. (Last accessed in 28/9/2022). [Cited on page 54]
- [62] Gazebo, “Gazebo.” Available at <https://gazebo.org/home>, 2022. (Last accessed in 7/10/2022). [Cited on page 54]
- [63] SolidWorks, “Solidworks.” Available at <https://www.solidworks.com>, 2022. (Last accessed in 7/10/2022). [Cited on page 54]
- [64] ROS, “Solidworks to urdf exporter.” Available at http://wiki.ros.org/sw_urdf_exporter, 2022. (Last accessed in 7/10/2022). [Cited on page 54]
- [65] ROS, “Smach - ros wiki.” Available at <http://wiki.ros.org/smach>, 2018. (Last accessed in 28/9/2022). [Cited on page 64]

-
- [66] J. J. Martins, “Case - collision avoidance system evaluation for agriculture robots.” Available at <https://doi.org/10.5281/zenodo.6812509>, 2022. (Last accessed in 20/10/2022). [Cited on page 76]
- [67] M. A. R. de Oliveira, E. F. Pedrosa, A. S. P. de Aguiar, D. F. P. D. Rato, F. B. N. dos Santos, P. M. de Jesus Dias, and V. M. F. dos Santos, “Atom: A general calibration framework for multi-modal, multi-sensor systems,” *Expert Systems with Applications*, p. 118000, 2022. [Cited on page 76]
- [68] Z. Zhang, “Iterative point matching for registration of free-form curves and surfaces,” *Int. Jornal Computer Vision*, vol. 13, pp. 119–152, 1994. [Cited on page 76]
- [69] Livox, “Livox mid-70 basic specs.” Available at <https://www.livoxtech.com/mid-70/specs>, 2022. (Last accessed in 28/08/2022). [Cited on pages x and 86]

Appendix A

Specifications Tables

A.1 Livox Mid-70 specifications table

Figure A.1 presents the specification table of the Livox Mid-70.

Model	MID-70
Laser Wavelength	905 nm
Laser Safety	Class 1 (IEC60825-1:2014)(Eye Safety)
Detection Range (@ 100 klx)	90 m @ 10% reflectivity 130 m @ 20% reflectivity 260 m @ 80% reflectivity
FOV	70.4° (Circular)
Range Precision (1σ @ 20m)	1σ (@ 20m) ≤ 2 cm ¹ 1σ (@ 0.2~1m) ≤ 3 cm ²
Angular Precision (1σ)	$< 0.1^\circ$
Beam Divergence	0.28° (Vertical) $\times$ 0.03° (Horizontal)
Point Rate	100,000 points/s (first or strongest return) 200,000 points/s (dual return)
Data Latency	≤ 2 ms
Data Port	100 Mbps Ethernet
Data synchronization:	IEEE 1588-2008 (PTPv2), PPS (Pulse Per Second), GPS (PPS+UTC)
False Alarm Rate (@ 100 klx) ³	$< 0.01\%$
Operating Temperature	-4°F to 149°F (-20°C to 65°C)
IP Rating ⁴	IP67
Power ⁵	8 W (average)
Power Supply Voltage Range ⁵	10 ~ 15 V DC (with Converter 2.0: 9~30V DC)
Dimensions	97×64×62.7 mm
Weight	580 g

Figure A.1: Livox Specifications [69].