UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E INFORMÁTICA INDUSTRIAL

MARCO ANTONIO SIMÕES TEIXEIRA

MOBILE ROBOTS
A STUDY ON SENSING AND PERCEPTION SYSTEMS

TESE

CURITIBA
2021

MARCO ANTONIO SIMÕES TEIXEIRA

MOBILE ROBOTS: A STUDY ON SENSING AND PERCEPTION SYSTEMS

Robôs Móveis: Um estudo sobre sensores e sistemas de percepção

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial (CPGEI) da Universidade Tecnológica Federal do Paraná (UTFPR) como requisito parcial para obtenção do título de "Doutor em Ciências". Área de Concentração: Engenharia de Computação.

Orientador: Prof. Dr. André Schneider de Oliveira
Coorientadora: Prof. Dra. Lúcia Valéria Ramos de Arruda

CURITIBA

2021

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Curitiba

MARCO ANTONIO SIMOES TEIXEIRA

ROBÔS MÓVEIS: UM ESTUDO SOBRE SENSORES E SISTEMAS DE PERCEPÇÃO

Trabalho de pesquisa de doutorado apresentado como requisito para obtenção do título de Doutor Em Ciências da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Engenharia De Computação.

Data de aprovação: 28 de Janeiro de 2021

Prof Andre Schneider De Oliveira, Doutorado - Universidade Tecnológica Federal do Paraná

Prof Andre Eugenio Lazzaretti, Doutorado - Universidade Tecnológica Federal do Paraná

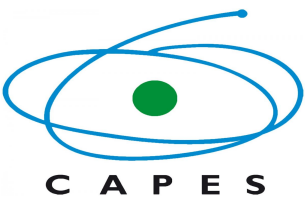
Prof Joao Alberto Fabro, Doutorado - Universidade Tecnológica Federal do Paraná

Prof Jose Luis Sousa De Magalhaes Lima, Doutorado - Instituto Politécnico de Bragança

Prof Julio Cesar Nievola, Doutorado - Pontifícia Universidade Católica do Paraná (Pucpr)

Prof.a Lucia Valeria Ramos De Arruda, Doutorado - Universidade Tecnológica Federal do Paraná

Este trabalho teve apoio financeiro da Agência Nacional do Petróleo, Gás Natural e Biocombustíveis - ANP, da Financiadora de Estudos e Projetos - FINEP, do Ministério da Ciência, Tecnologia e Inovação - MCTI por meio do Programa de Recursos Humanos da ANP para o Setor Petróleo e Gás - PRH-ANP/MCTI do Programa de Formação de Recursos Humanos da PETROBRAS - PRH10-UTFPR e da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES.



ACKNOWLEDGEMENTS

No victory is achieved without sacrifice, effort, and help. I am grateful to have had this experience and to be able to learn and grow with it. Throughout the period, I had the opportunity to meet countless exciting people who contributed to the development of this thesis or to my personal development in one way or another.

First, I would like to thank my advisor André Schneider de Oliveira (UTFPR), and co-advisor Lucia Valéria Ramos de Arruda (UTFPR). I have tried over the years to absorb as much of both, but I believe both still have a lot to teach me, and I hope to continue working and learning from you.

I want to thank all members of the *Laboratorio de Automaçao e Sistemas de Controle Avançado* (LASCA). Throughout the period, I was able to answer questions and collaborate with the work of other colleagues. The Laboratory has always provided all the necessary support, with equipment, staff, or financial resources, essential for developing the research.

I thank my entire family. In particular, my wife, who has been by my side for the entire period, and has often celebrated with me and for many other times has comforted me when something doesn't go as planned. To all those who will contribute to me in some way, and I ended up forgetting to mention, I would like to thank you very much.

I would like to especially thank CAPES for providing a scholarship throughout the doctorate, allowing me to dedicate to research. I am also grateful to all the other development agencies mentioned above and Nvidia Corporation for donating a Titan Xp video card, which was essential for the development and continuity of the research.

RESUMO

TEIXEIRA, Marco Antonio Simões. **MOBILE ROBOTS: A study on sensing and perception systems**. 2021. 106 f. Tese (Doutorado em Engenharia Elétrica e Informática Industrial) – Universidade Tecnológica Federal do Paraná. Curitiba, 2021.

Robôs móveis são equipamentos utilizados para executar tarefas que necessitam que o equipamento se locomova no ambiente. Estas tarefas podem ou não ser inteligentes, dependendo da necessidade da ação. Para que um robô seja capaz de tomar decisões, ele precisa coletar dados do ambiente, processar estes dados e convertê-los em informação. Esta tese tem por objetivo estudar técnicas de sensoriamento de ambiente utilizados tradicionalmente por robôs móveis e propor uma nova técnica de sensoriamento, que utiliza os dados fornecidos pelos sensores, os processa e retorna informações. Para alcançar o objetivo esperado, primeiro foi realizado um estudo sobre os sensores tradicionais utilizados na robótica móvel, para posteriormente ser proposto uma nova abordagem de sensoriamento. Primeiro, foi desenvolvida uma abordagem de mapeamento específico para tanques de armazenamento de Gás Liquefeito de Petróleo (GLP) para que um robô de inspeção escalador, desenvolvido pela UTFPR em parceria com a Petrobras, fosse capaz de realizar a inspeção preventiva em tanques de GLP. Esta técnica foi capaz de predizer toda a superfície do ambiente sem a necessidade de uma varredura completa. Posteriormente, foi estudado a aplicação de técnicas inteligentes de processamento de dados 3D para a navegação e autopreservação de veículos aéreos não tripulados (VANT's). Foi desenvolvido uma técnica de navegação em formação para 4 VANT's, evitando colisões com o ambiente e entre eles, sempre mantendo a formação durante todo o percurso. Esta ação só foi possível pelo processamento dos dados dos sensores 3D, convertidos em informação de distância a partir do centro do VANT e utilizada para a realização de tarefas de desvio de obstáculo, e autopreservação. A partir destes dois primeiros trabalhos, ficou evidente a necessidade de processar os dados fornecidos pelos sensores para que fossem geradas informações uteis para a tomada de decisão. O próximo trabalho da tese teve como objetivo o desenvolvimento de uma abordagem de processamento de dados provenientes de sensores 3D e imagens RGB para a geração de informações, que podem ser utilizadas por um robô. A abordagem consistiu no uso de uma técnica de visão computacional para identificar objetos em uma imagem RGB e posteriormente, na junção da imagem RGB com os dados 3D provenientes do sensor para a identificação destes objetos no mundo real em relação ao centro do equipamento. Posteriormente, a abordagem foi embarcada em um equipamento compacto, chamado de sensor *DeepSpatial*. Este equipamento foi acoplado a um robô, e validado para aplicações tradicionais em robôs móveis, comprovando a eficiência das informações fornecidas pelo sensor. Como resultado deste trabalho, uma nova abordagem de sensoriamento foi proposta, onde sensores tradicionais são utilizados para ações inteligentes. A abordagem é embarcada em um equipamento compacto, que pode ser considerado um novo sensor.

Palavras-chave: Robótica móvel. Sensores. Percepção 3D. YoLo. DeepSpatial.

ABSTRACT

TEIXEIRA, Marco Antonio Simões. **MOBILE ROBOTS: A study on sensing and perception systems**. 2021. 106 p. Thesis (Doctorate in Electrical and Computer Engineering) – Universidade Tecnológica Federal do Paraná. Curitiba, 2021.

Mobile robots are equipment used to perform tasks that require the equipment to move around the environment. Mobile robots are equipment used to perform tasks that require the equipment to move around the environment. Mobile robots can use machine learning techniques to perform intelligent tasks, such as recognizing objects and making decisions. For a robot to make decisions, it needs to collect data from the environment, process it, and convert it into information. This thesis aims to study environment sensing techniques traditionally used by mobile robots and to propose a new sensing technique, which uses the data provided by the sensors, processes them, and returns information. To achieve the objective of the thesis, a study was carried out on the traditional sensors used in mobile robotics and, then, a new sensing approach was proposed. First, a specific mapping approach was developed for Liquefied Petroleum Gas (LPG) storage tanks so that a climbing inspection robot, developed by UTFPR in partnership with Petrobras, was able to carry out preventive inspection on LPG tanks. This technique could predict the entire surface of the environment without the need for a complete scan. Subsequently, the application of intelligent 3D data processing techniques for navigation and self-preservation of unmanned aerial vehicles (UAVs) was studied. A navigation technique was developed in formation for 4 UAVs, avoiding collisions with the environment and between them, always maintaining the formation throughout the route. This action was only possible by processing the 3D sensor data, converting it into distance information from the center of the UAV, and performing obstacle avoidance tasks and self-preservation. From these first two works, the need to process the sensors' data to generate useful information for robot decision-making became evident. The next paper of the thesis aimed to develop an approach for processing data from 3D sensors and RGB images to generate information, which can be used by a robot. The approach consisted of using computer vision to identify objects in an RGB image and point cloud processing to identify these objects in the real world. Subsequently, the approach was embedded in a compact device, called a *DeepSpatial* sensor. This equipment was coupled to a robot and validated for traditional applications in mobile robots, proving the sensor's information's efficiency. As a result of this thesis, a new sensing approach was proposed, where traditional sensors are used for intelligent actions. The approach is embedded in a compact device, which can be considered as a new sensor.

Keywords: Mobile robotics. Sensors. 3D perception. YoLo. DeepSpatial.

LIST OF FIGURES

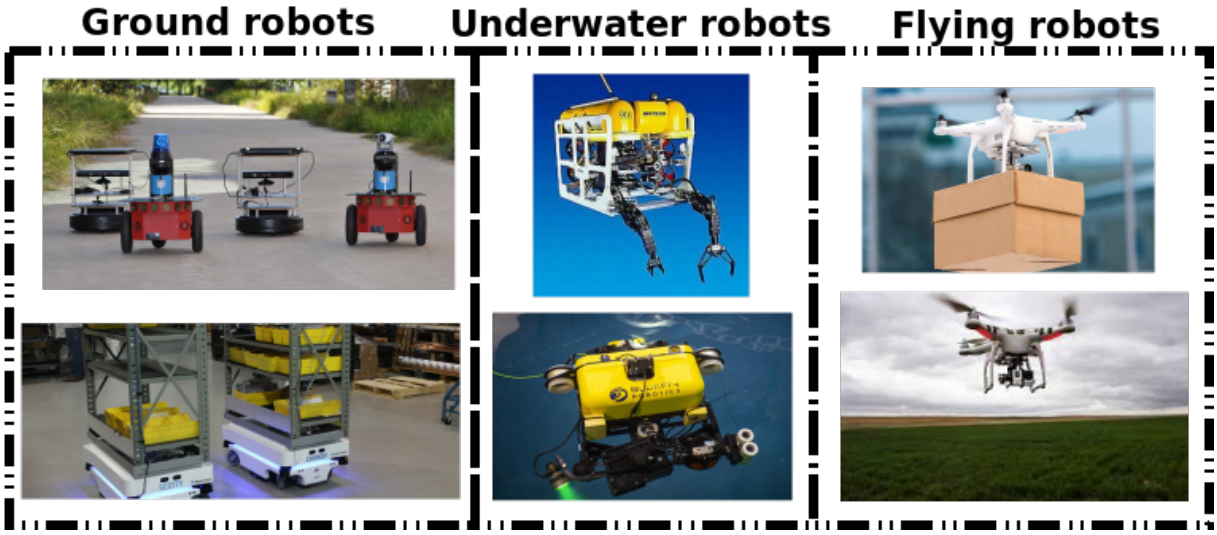
LIST OF TABLES

CONTENTS

1 INTRODUCTION

Mobile robotics is an area of science that studies the development of complex robotic systems, which can freely move in the environment (DUDEK; JENKIN, 2010; NEHMZOW, 2012; CAO *et al.*, 1997). Unlike a fixed robot, such as robotic arms used in factories (HAYATI, 1986), a mobile robot can move around in the environment, whether by land, water, and sky (POLLARD; TALLAPRAGADA, 2016; TAN *et al.*, 2017; RYLL *et al.*, 2017; PARKER *et al.*, 2016) (Fig. 1).

Figure 1 – Examples of mobile robots.



Source: Own authorship.

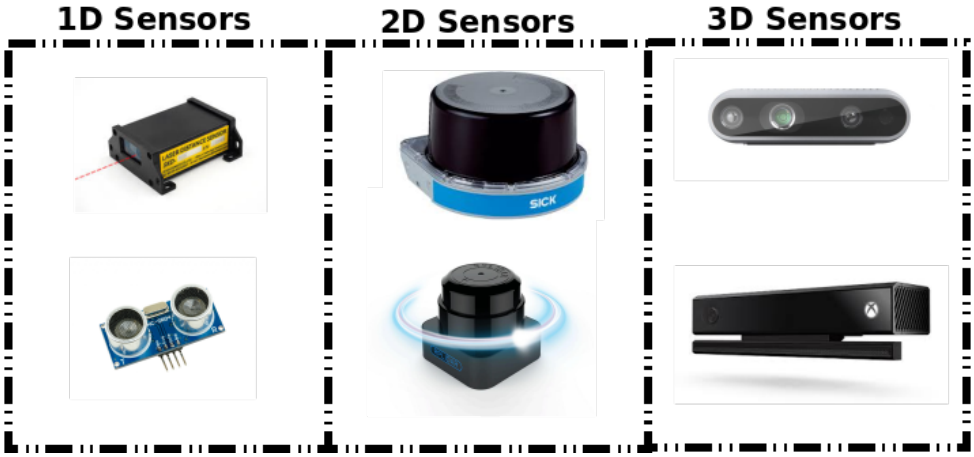
All robots in Fig. 1 have in common the need to understand the environment around them. Regardless of whether the robot is on the ground, the sea, or flying over a city, it needs to have a minimum of perception of the environment to avoid accidents. A robot without a sensor can be compared to a person without senses. If the robot is unable to identify at least if there is an obstacle in front of it, it will not take actions such as stopping or deflecting. However, the use of intelligent sensors can make the work of these robots collaborative and increase their productivity.

To perceive the environment, mobile robots can make use of different types of sensors. Land and air robots can make use of the same sensors, and underwater robots need individual sensors capable of operating underwater. There are several sensors for mobile robots. An ultrasonic sensor can be used to identify an object in front of the robot, for example (CARULLO;

PARVIS, 2001). The ultrasonic sensor can provide the distance of an object in front of the robot over the signal's flight time; however, its distance data is not accurate, as the signal is cone-shaped, having a high degree of openness, and can provide the wrong position of the obstacle. In addition to ultrasound, laser sensors are also capable of capturing the distance to an object through the time of flight (KONOLIGE *et al.*, 2008). There are sensors capable of providing more points in a single measurement, such as laser scanning sensors (LiDAR). LiDAR sensors use a fixed laser, and a rotating mirror to collect flight time at different angles, thus generating a 2D distance plan, which can be used by mobile robots for the most diverse tasks, such as avoiding obstacles, generating planar maps, and others (PALACÍN *et al.*, 2020; SABATINI *et al.*, 2014; GHORPADE *et al.*, 2017).

With the use of LiDAR sensors, it is possible to perform safe navigation by stopping when identifying an obstacle, for example. In addition to 2D sensors, there is another category of sensors capable of providing a higher amount of information. These are 3D sensors. 3D sensors provide data known as PointCloud (BOULCH *et al.*, 2017; RUSU; COUSINS, 2011). These data are matrices containing the 3D coordinates of each point. They can be collected in two ways, whether by time of flight (HAGEBEUKER; MARKETING, 2007), which is the case with ultrasonic sensors, for example, or by structured light (ROCCHINI *et al.*, 2001) being the most common and cheapest of the techniques. Usually, 3D sensors also provide colors associated with the 3D point; these data are known as RGB-D. This data can be used for various mobile robotics tasks, such as building 3D maps, performing object detection, and other activities (JAFARI *et al.*, 2014; HUANG *et al.*, 2017; GIORDANO *et al.*, 2016). A visual representation of the sensors mentioned can be seen in Fig. 2.

Figure 2 – Visual representation of sensors used in mobile robotics.



Source: Own authorship.

Independent of the type of sensor being used, the collected data must be processed to generate useful information for the robot. During navigation, it is necessary to collect all data and process to obtain distance information and check if there is any object close to the robot or not, in order to perform an emergency stop action, for example. For instance, during the map building phase, it is necessary to collect the data from the sensors, convert into distance information in relation to the robot's position, and save it to generate a map. Thus, the sensors are capable of supplying data to the robot and not information. It must continuously process them, often generating reworks. Some modern sensors try to integrate the processing with the sensor, thus facilitating the creation of intelligent mobile robot projects, as is the case with (MOKHTARI *et al.*, 2017) that propose the development of a sensor that can identify people or (RONAO; CHO, 2016; CHEN *et al.*, 2017) that make use of conventional sensors with processing techniques, generating useful information for the user.

In this way, sensors are essential for robotic applications because it is impossible to perform simple tasks such as navigating the environment without them. However, they are designed to provide raw data, which in some cases is not sufficient. Several papers focus on converting sensor data into information, for example, in (LIANG *et al.*, 2018; WANG *et al.*, 2018). However, these papers are not aimed at robotic applications and are usually performed on computers with high processing power.

This thesis aims to study and analyze the use of sensors in mobile robots to propose contributions in sensing techniques, mainly processing data provided by traditional sensors into information used by the robot. Intelligent sensors combine physical sensing with data processing techniques to generate intelligent information, which can be used for the robot to obtain information from the environment. A study of sensors was first carried out on terrestrial robots, where the limitations of traditional sensors were verified. Most sensors provide the same type of data, point cloud. Subsequently, converting point cloud data into useful information was carried out in a study with drones. After the preliminary studies with the sensors, an intelligent sensing approach was proposed, using deep learning techniques. Finally, a sensor architecture was proposed, where all the developed technique is embedded in a single equipment. Validations were made at all stages of the thesis.

1.1 OBJECTIVE

This thesis proposes a study, analysis, and new approach to environmental sensing. The main focus is to expand the sensors' information, transforming the measured data into useful information. This new technique of perception of the environment will allow robots to safely navigate the environment, knowing which objects are around them, making the robot capable of performing specific actions for each object. This technique must be embedded in hardware that is robust enough to perform all robot actions independently. Therefore, this thesis's general objective is to develop an intelligent sensor capable of identifying dynamic and static objects, extracting characteristics from the identified objects, and when coupled to a robot, interact with it to support an autonomous decision-making capacity by part of the robot.

1.1.1 Specific objectives

In order to achieve the main objective, the following specific objectives are outlined.

1. Develop a study and analysis of sensing techniques for mobile robots, using the manipulation of point cloud data provided by different sensors, for the creation of maps, object identification, and navigation in order to understand the use of traditional sensors applied to mobile robotics and identify possible improvements;

2. Apply point cloud techniques in aerial robot problems in order to validate sensing techniques. Aerial robots share the same sensing needs as terrestrial robots. This study aims to validate the data manipulation approach for generating information for the robot;

3. Propose a 3D perception system from the data processing of traditional sensors. The proposed system must be able to generate information such as the three-dimensional position of objects in the environment around the sensor and identify objects using computer vision techniques in RGB images;

4. Develop an approach to solving the egomotion problem so that intelligent sensing techniques can identify movements. In addition, it is expected to embark the entire approach proposed in a compact equipment, coupled to a mobile robot;

1.2 PUBLICATIONS

During this thesis development, the results were published both in scientific journals and in chapters of books and congresses. The publications directly related to the thesis are presented below.

1.2.1 Journal

1. TEIXEIRA, M. A. S.; NEVES-JR, F. ; KOUBAA, A. ; ARRUDA, L. V. R. ; OLIVEIRA, A. S. DeepSpatial: Intelligent Spatial Sensor to Perception of Things. **IEEE Sensors Journal**, 2020.

2. TEIXEIRA, M. A. S.; NEVES-JR, F.; KOUBAA, A.; ARRUDA, L. V. R.; OLIVEIRA, A. S. A Quadral-Fuzzy control approach to flight formation by a fleet of unmanned aerial vehicles. **IEEE Access**, 2020.

3. TEIXEIRA, M. A. S.; NOGUEIRA, R. C. M. ; DALMEDICO, N. ; SANTOS, H. B. ; ARRUDA, L. V. R.; NEVES-JR, F. ; PIPA, D. R. ; RAMOS, J. E. ; OLIVEIRA, A. S. Intelligent 3D Perception System for Semantic Description and Dynamic Interaction. **Sensors**, v. 19, p. 3764, 2019.

4. TEIXEIRA, M. A. S.; SANTOS, H. B.; DALMEDICO, N.; ARRUDA, L. V. R.; NEVES, F.; OLIVEIRA, A. S. Intelligent environment recognition and prediction for NDT inspection through autonomous climbing robot. **Journal of Intelligent & Robotic Systems**, v. 92, p. 323-342, 2018.

1.2.2 Book chapters

1. TEIXEIRA, M. A. S.; SANTOS, H. B.; OLIVEIRA, A. S.; ARRUDA, L. V.; NEVES, F. Robots Perception Through 3D Point Cloud Sensors. **Studies in Computational Intelligence.** 2ed.: Springer International Publishing, 2017, v. , p. 525-561.

1.2.3 Congresses

1. TEIXEIRA, M. A. S.;DALMEDICO, N.; SANTOS, H. B.; OLIVEIRA, A. S.; ARRUDA, L. V.; NEVES-JR, F. Enhancing Robot Capabilities of Environmental Perception through Embedded GPU. In: **2017 VII Brazilian Symposium on Computing Systems Engineering (SBESC)**, 2017, Curitiba. 2017 VII Brazilian Symposium on Computing Systems Engineering (SBESC), 2017. p. 217.

2. TEIXEIRA, M. A. S.; DALMEDICO, N.; OLIVEIRA, A. S.; ARRUDA, L. V.; NEVES-JR, F. A pose prediction approach to mobile objects in 2D costmaps. In: **2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR)**, 2017, Curitiba. 2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR), 2017. p. 1.

1.3 THESIS STRUCTURE

This thesis is developed in the form of a collection of articles. This new format allows the thesis to be composed of at least three articles published in journals, attached in full, where the author of the thesis must be the first author. This means that the format can vary from chapter to chapter, as some chapters are added articles, maintaining their publication format. Other sections are in the form of a thesis, and aim to contextualize the research developed. Thus, this first chapter aims to present the problem as a whole, adding the reader to the theme of sensing for mobile robots. Chapter 2 aims to present the relevance of the research topic, presenting comparisons with recent works and a literature review. The chapters 3,4,5 and 6 present the articles already published in full journals, while the chapter 7 presents a conclusion of the thesis.

In chapter 2, the research topic about recent academic works is presented. In chapter 3 it is presented the first paper published by the author (TEIXEIRA *et al.*, 2018) in *Journal of Intelligent Robotic Systems*. This chapter is part of the first specific objective, where the traditional sensors were used to the maximum, in order to study their applications and develop improvement strategies. Chapter 4 presents the second article published in *IEEE Access* (TEIXEIRA *et al.*, 2020). The focus is to understand the sensing needs of mobile robots. In this thesis, drones were used with a 3D perception sensor.

Chapter 5 presents a proposal for an intelligent sensor. The robot can use the proposed sensor without the need for the robot to know its operation. The results were published in the *Sensors* journal (TEIXEIRA *et al.*, 2019). However, in this thesis, the movement of the sensor or robot itself is not considered. The technique was improved and shipped with a set of equipment's that together can be considered as a new sensor, presented in chapter 6. In chapter 6, the proposed strategy is embedded and can be added directly to the robot, like a black box. The robot does not need to know how the information is processed. It has access to ready information, such as dynamic objects in front of it and the speed and direction of these objects, in addition to being able to measure its movements in the environment. The results were published in the *IEEE Sensors* journal (TEIXEIRA *et al.*, 2021).

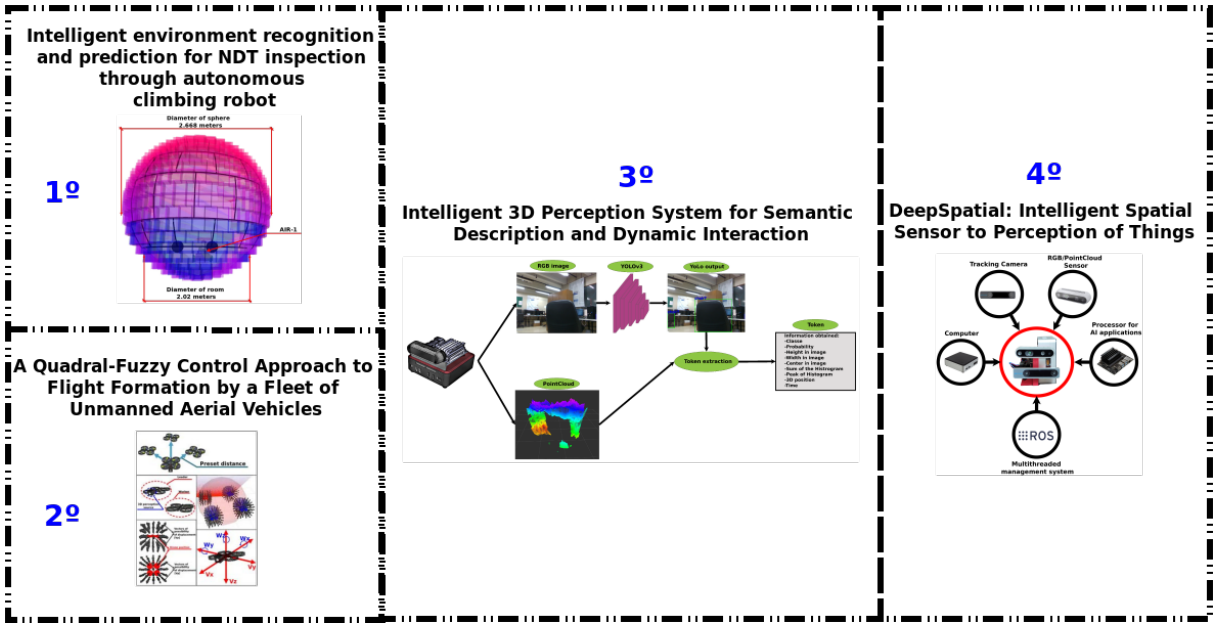
Finally, chapter 7 presents a conclusion to the thesis. Thus, this thesis aimed to study sensing techniques for mobile robots, aiming to generate the maximum amount of information that can be used by different intelligent strategies. This thesis leaves open future work, such as the improvement of the proposed sensor for the development of an intelligent map.

2 CONTEXTUALIZATION

This chapter aims to contextualize the papers attached to this thesis with the theme of studies, which is "*Mobile robotics: Perception of the environment*". In addition to bringing a contextualization of the works with the research area, a brief comparative theoretical framework will present the works developed and the state of the art. The specific theoretical review of each work is presented in its respective chapter.

With this thesis's focus on the study of sensors for use in mobile robots, the general objective was divided into four specific objectives. In the first specific objective, the work (TEIXEIRA *et al.*, 2018) was developed, while in the second specific objective, the result was the work (TEIXEIRA *et al.*, 2020). For the third specific objective, the work (TEIXEIRA *et al.*, 2019) was developed. The fourth specific objective is presented in the chapter 6. Figure 3 presents an illustration of the works attached to this thesis. It is worth mentioning that some works were developed for congresses and book chapters and are not attached to this thesis.

Figure 3 – Division of the works presented in this thesis by specific objectives proposed.



Source: Own authorship.

In the chapter 3, the work "*INTELLIGENT ENVIRONMENT RECOGNITION AND PREDICTION FOR NDT INSPECTION THROUGH AUTONOMOUS CLIMBING ROBOT*" (TEIXEIRA *et al.*, 2018) was developed. This work aimed to develop a mapping strategy for an inspection robot for liquefied petroleum gas (LPG) tanks. LPG tanks are spherical tanks that

require periodic inspections. In this work, Fuzzy control techniques and the use of multiple 3D sensors were used to create a mapping strategy. An environment prediction strategy was developed, based on data obtained by the sensor and knowledge inference about the inspection environment. For the prediction, artificial neural networks were used to infer a certain percentage of the predicted points.

Although the map developed in this thesis was not traditional mapping techniques, traditional 2D (THRUN, 2003; ELFES, 1989; MEHREZ *et al.*, 2017) and 3D (HORNUNG *et al.*, 2013; WURM *et al.*, 2010; SUN *et al.*, 2018) mapping techniques were studied. It was realized that in traditional mapping techniques, the robot must navigate throughout the environment, and this action is costly for the inspection robot. A solution to this problem was the creation of an environment prediction strategy. The work presented by this thesis infers knowledge in the algorithm, such as the construction format of the tanks, to predict the positions of the weld beads.

Regarding the perception sensors, four different sensors were used. Three were fixed on a mobile base. All sensors were different, but they provide the same type of information, distances. It was noticed that, with so many sensors, points overlapped. To address these problems, a merger of points was proposed. In related works, it is possible to quote (LIANG *et al.*, 2018; KNOOP *et al.*, 2006; PEŁKA *et al.*, 2019) where strategies were also presented to merge different sources of perception to decrease the amount of information coming from the sensors. In this thesis, the fusion took place due to the precision of each sensor with the distance from its points. The sensor that, according to its technical data sheet, presents a better accuracy, receives a higher percentage in the fusion. A separate work, only with the fusion of the sensor data, was developed, and can be accessed at (TEIXEIRA *et al.*, 2017).

Still, with the focus of understanding the sensors and improving the sensing techniques in mobile robotics, in the chapter 4 the work "*A QUADRAL-FUZZY CONTROL APPROACH TO FLIGHT FORMATION BY A FLEET OF UNMANNED AERIAL VEHICLES*" is presented (TEIXEIRA *et al.*, 2020). This paper consists of a new proposal for navigation in training with multiple drones. If Drones fly in formation, they can perform collaborative tasks, such as cargo transportation. The work used intelligent control techniques, such as the use of a Fuzzy system. The information was shared from the leading Drone to the worker Drones, who together performed an obstacle avoidance action maintaining the formation.

Regarding the control of mobile robots through Fuzzy systems, there are several re-

lated works (ABDALLA *et al.*, 2017; ABIYEV *et al.*, 2017; YEN; CHENG, 2018; HUANG, 2016). In (BACIK *et al.*, 2017), was presented using Fuzzy systems to control the position of a drone, where the position is defined using tags fixed to the ground. Unlike that work, the work presented in this thesis proposes using Fuzzy techniques for position controls, where the positions of the working drones are defined from the position of the leading Drone. Our work also proposes using Fuzzy techniques to avoid obstacles based on the 3D perception sensor data.

The work also covered flying techniques in formation. Similar recent work can be found (LI *et al.*, 2019; YU *et al.*, 2019; WANG *et al.*, 2019), where the goal is to develop smart strategies for collaborative work between multiple agents. The purpose of these works is to make the robots work collaboratively to act. In (ALONSO-MORA *et al.*, 2019) a flight training strategy is presented. However, the training is undone when trying to avoid an obstacle, which is different from the work proposed in this thesis, where the objective is to maintain the formation at any cost.

In this thesis, a single 3D perception sensor was used. The sensor data were processed by the Drone Leader and the information on identified obstacles shared with the other Drones in the formation. This single work made use of sensing techniques, intelligent controls, and multiple robot systems and was essential for this thesis. From it, it was possible to identify the main problems in sensing for mobile robots. 3D sensors provide only distance data, raising the following question: If the obstacle identified by the robot is known, the robot could not take a better action than choose one side and deflect? What if the object in question is also moving?

From these works, it became clear the need for a robot to identify objects and their positions. The work used the identification of the weld bead and its junctions to predict the entire environment. With the results obtained in the papers (TEIXEIRA *et al.*, 2018) and (TEIXEIRA *et al.*, 2020), the next work of this thesis aimed to develop a sensing strategy, not focusing on the robot, but in generating information from traditional sensors for used by the robot. The work "*INTELLIGENT 3D PERCEPTION SYSTEM FOR SEMANTIC DESCRIPTION AND DYNAMIC INTERACTION*" (TEIXEIRA *et al.*, 2019) was developed, where it aimed to develop a new environment perception strategy, using computer vision and sensing techniques.

In this thesis, computer vision techniques were used to identify objects in an RGB image. Related works can be seen at (LIU *et al.*, 2020; ZHAO *et al.*, 2019; HU *et al.*, 2018), where deep learning concepts are used to recognize objects in traditional images. For object recognition, YoLo (REDMON; FARHADI, 2018; REDMON; FARHADI, 2017) was used, as it

has a good ratio between performance and accuracy. The data obtained by YoLo are processed with the 3D perception sensors to identify the object's position. Related works can be found at (SIMONY *et al.*, 2018; SIMON *et al.*, 2019; YANG *et al.*, 2018), where computer vision techniques were used together with 3D perception data to identify objects. Unlike the works cited, the work present in this thesis, in addition to identifying objects in the real world, performed the tracking calculating speed, direction, and acceleration.

Chapter 4 resulted in a sensing strategy, in which objects in the environment and their 3D positions are identified and calculating their speeds, direction, and accelerations. However, the strategy was developed on a fixed computer and cannot be coupled onto the robot. From this paper, the work presented in the chapter 6 under the name "*DEEPSPATIAL: INTELLIGENT SPATIAL SENSOR TO PERCEPTION OF THINGS*" (TEIXEIRA *et al.*, 2021) was developed, where the objective is to improve the techniques developed and embed them on portable equipment and attach it to the robot, thus fulfilling the third and final specific objective proposed by this thesis. In this work, he was also concerned with the movement of the robot in the environment. If the robot moves, the calculated direction, speed, and acceleration measurement data can be wrong. To solve this problem, a sensor capable of measuring the robot's movement itself in the environment using a visual odometer was used.

Visual odometry uses computer vision techniques to calculate the robot's movement in the environment. With this technique, it is possible to calculate the robot's movement through RGB cameras, where a specialized algorithm compares these images and returns the displacement. Many works use visual odometry, with different techniques (FORSTER *et al.*, 2016; MUEGGLER *et al.*, 2017; ZHANG *et al.*, 2016; MUEGGLER *et al.*, 2018). In (MOHANTY *et al.*, 2016) a visual odometry technique is proposed using deep learning with the *AlexNet* network.In (FORSTER *et al.*, 2016), an algorithm is proposed that identifies specific points in the image, and according to the displacement of these points, the displacement of the equipment in relation to the image is calculated.

The proposed sensor, DeepSpatial, identifies objects in the environment and their 3D position, tracking them and calculating their speed and direction, compensating for their movements in the environment using visual odometry techniques and is mobile, and can be embedded onto the robot. In this way, this thesis fulfilled all the proposed objectives, being the study of traditional mobile robotics techniques, the proposal and validation of new sensing techniques.

3 INTELLIGENT ENVIRONMENT RECOGNITION AND PREDICTION FOR NDT INSPECTION THROUGH AUTONOMOUS CLIMBING ROBOT

This chapter presents the article published in the *Journal of Intelligent & Robotic Systems* (ISSN: 0921-0296). The data of the paper is shown in Table 1.

Table 1 – Data of the paper *Intelligent environment recognition and prediction for NDT inspection through autonomous climbing robot.*

Authors	Teixeira, M.A.S.; Santos, H.B.; de Oliveira, A.S.; Arruda, L.V.R.; Neves-Jr, F.
Title	*Intelligent environment recognition and prediction for NDT inspection through autonomous climbing robot*
Journal	*Journal of Intelligent & Robotic Systems*
ISSN	0921-0296
DOI	https://doi.org/10.1007/s10846-017-0764-6
Publication date	January 19, 2018 online and October 2018 printed

Source: Own authorship.

This work's motivation was due to the difficulty in carrying out inspections in storage tanks for liquefied petroleum gas (LPG). Inspections are carried out by people who need to put themselves in situations that are often dangerous to their health, such as in high places or an environment with a high concentration of gases harmful to the human body. In this way, the project aims to avoid exposing people to risk. For this, a mobile robot is proposed under development by the Federal University of Technology – Paraná in partnership with Petrobras. The creation of the mobile inspection robot, called *Autonomous inspection robotis* currently has three versions. This work uses version 1, called AIR1.

To be able to inspect an LPG storage tank by a robot autonomously and intelligently, the equipment needs to be able to collect data from the environment, analyze and process them, and make decisions. This work focuses on processing data from different sensors to create and predict the map of the environment. This map is also possible to carry out the trajectory planning, which would be the path to be taken by the robot during the inspection. Defining trajectory planning is not the focus of this work. The main objective is to create a three-dimensional representation of the environment, with as many features as possible, without going through the entire environment. The goal is achieved with the use of artificial intelligence techniques, knowledge inference, raw sensor data processing, among others.

This work concludes the first specific objective of the thesis, contributing to identifying possible improvements and contributions in the area of data processing from sensors used in robots. The license for this article with permission for use in this thesis is in Appendix A.

Journal of Intelligent & Robotic Systems (2018) 92:323–342
https://doi.org/10.1007/s10846-017-0764-6

Intelligent environment recognition and prediction for NDT inspection through autonomous climbing robot

Marco Antonio Simoes Teixeira[1] · **Higor Barbosa Santos**[1] · **Nicolas Dalmedico**[1] · **Lucia Valeria Ramos de Arruda**[1] · **Flavio Neves-Jr.**[1] · **Andre Schneider de Oliveira**[1]

Received: 26 April 2017 / Accepted: 13 December 2017 / Published online: 19 January 2017

Abstract
This paper presents a novel approach to environment mapping prediction with focus on autonomous climbing robot to NDT (Non-Destructive Technique) inspection. In industrial installations, the inspection of non-planar surfaces requires that NDT probes passe on whole surface, while the autonomous robot navigates over an unknown environment based only on its perception abilities. However, the path planning of inspection is not a trivial task specially when there is no precise information about environment. In this work, a special kind of climbing robot is used to inspect large metallic surfaces such as spherical pressure vessels used to store Liquified Petroleum Gas (LPG). The robot has adherence skills that allow it to safely navigate through the internal and external surface of the vessel. As a result, robot mobility suffers from hard magnetic adhesion constraints. A new approach is proposed to environment detailed prediction, including specific characteristic (like weld beads and plates) of inspected surface. The goal is the automatic extraction of some environment characteristics to predict the storage tank dimensions and robot localization, based on a group of 3D perception sources (laser rangefinder, light detection and ranging and depth camera) mounted over a rolling platform to improve its reach. The environment prediction is carried out after the robot visually detects two or more weld beads corners. A multi-measuring environment is firstly build by Fuzzy data fusion of the different perception measurements allowing to estimate plates and weld beads based on design and safety standards. Virtual and real experiments are carried out to illustrate proposed method performance.

Keywords Environment identification · Data fusion · Path planning · Inspection · Climbing

1 Introduction

The robotic inspection is a task where a robot must cover all surface to be inspected, searching for structural fails and damages through inspection probes. The inspection planning requires a reliable knowledge of the inspection environment to ensure an effective path planning which runs across whole surface. However, robot usually has not a precise description of the surface to be inspected and some automatic procedures must be applied to acquire this knowledge.

3D Perception promotes the spatial awareness on mobile robots, allowing the perception of obstacles around it, but without recognize them. The robot can estimate its distance from obstacles although it does not identify the kind of objects. This object perception is very important to allow automatic behaviors (such as navigation and mapping). Many studies discuss how to apply 3D perception to object recognition, as in [1] which uses a Kinect sensor to create 3-D maps and 3-D objects for recognition. Browatzki et al. [2] uses an iCub humanoid robot for recognition and to discover new objects. Singh et al. [3] creates a data set of 100 objects through depth and image sensors. However, perception is an instantaneous ability because the robot only

✉ Marco Antonio Simoes Teixeira
mantonio.t91@gmail.com; marcoteixeira@alunos.utfpr.edu.br

Higor Barbosa Santos
higorsantos@alunos.utfpr.edu.br

Nicolas Dalmedico
ndalmedico@alunos.utfpr.edu.br

Lucia Valeria Ramos de Arruda
lvrarruda@utfpr.edu.br

Flavio Neves-Jr.
neves@utfpr.edu.br

Andre Schneider de Oliveira
andreoliveira@utfpr.edu.br

[1] Federal University of Technology - Parana, Av. Sete de Setembro, 3165, Curitiba, Parana, Brazil

detects obstacles into the perceived region and it does not maintain the memory about previous perceptions.

Simultaneous Localization and Mapping (SLAM) is a crucial technique to autonomous navigation because it creates a memory of perceptions into an environment map. This classic approach is discussed in many studies, e.g., in [4], a Kinect is mounted on a PR2 robot to get data distance between Kinect and the environment and to build a 3-D map of the environment. [5] presents an open-source tool that facilitates maps development, this tool uses three-dimensional data to create a map based on octrees technique and probabilistic occupancy estimation. In [6], scanning laser sensor is used to create a cloud of 3-D points to represent the environment. The laser scan is widely used to create 2-D maps, but to create a 3-D map it is necessary to rotate the laser so that it is possible to obtain various environmental reference points and then build the map. Such map is dynamically designed with environment perception and previous environment statistical knowledge.

Path planning requires at least a superficial knowledge of the whole environment or map estimation. In large environments, it is not simple to use a long-range perception due the presence of many probably blind points. A solution can be the environment reconstruction using a unique or partial view and then to estimate the blind surfaces. The reconstruction is the act of modeling a real object to the digital world by creating a 3D model of the object. These techniques are discussed, mainly on computer vision area, e.g., in [7] RGB-D sensors are used to obtain data from the real object, then the data is refined and processed before being submitted to the reconstruction algorithm. Finally, textures are added to the model using data provided by the RGB camera present in the sensor. Also in the reconstruction area, a Kinect is used for the data scene and then its reconstruction [8]. The differential of this study is an interaction system which is developed between the user in real and virtual scene, opening up new possibilities in the reconstruction area and human-machine interaction. However, the presented reconstruction method develops the 3D models of real objects but without taking into account the object localization, not allowing its inclusion in an environment map.

Map prediction aims to estimate the environment topology based only on simple observations through 3D perceptions. Some studies use this approach to predict unobserved parts in partially known maps. [9] discusses the simultaneous localization and mapping with environmental-structure prediction (P-SLAM), where the unobserved areas are estimated from of known areas in indoor environments. This approach is based on the assumption that unknown sectors have correlation with known sectors, to delimit a standard in the environment topology and predict unobserved areas, and it is proposed to predict a small piece of map. In [10], this approach is expanded to predict the motion of moving objects. This kind of prediction is very dependent on observed areas or previous knowledge of environment and cannot be applied in dynamic maps where the maps are fully created during robot navigation.

Some papers discuss distinct mapping techniques and path planning, as is the case of [11] where a mapping and trajectory planning technique is developed for a robot with legs and applied to rough and unstructured terrains. Another related work can be seen in [12], where mapping and path planning based on long range sensors is developed and the uncertainties present in this type of measurement are quantified. These approaches have discussed the same primordial requirement despite to discuss different techniques to mapping and path planning. Both are very dependent of robot's observation and require a long motion in environment to perform a reliable mapping and allowing the path planning.

This present work is focused in propose an alternative to mapping and planning through a small and instantaneous observation of environment with a reliable prediction of unobserved part.

This work proposes a novel approach to intelligent prediction of occupancy map based on a minimal observation to allow a reliable path planning. The main focus is to ensure a precise planning of nondestructive inspection task to an intelligent autonomous climbing robot. The inspection environment is projected through a very small set of observation with use of distinct 3D perception sources, which are carefully fused in a most reliable and condensed perception data set. A voxels group (volumetric element) are applied to generate an occupancy grid wrapping the inspection tank and the confidence analysis of these voxels is performed to determine their uncertainties. The result is a reliable map prediction, including specific characteristics of inspection environment (as weld beads and plates), which allows a rigorous inspection planning.

The paper is organized as follows. In Section 2 the requirements for inspection of storage tanks are discussed. Section 3 presents the developed Autonomous Inspection Robot and its perception systems. Section 4 describes the whole approach of planning of nondestructive inspection in storage tanks. In Section 5, the benefits of proposed approach are experimentally proved. Section 6 presents the conclusions and future works.

2 Requirements of Storage Tanks Inspection

Storage tanks are metallic structures commonly used in industries to store liquids and gasses. The cylindrical tanks are preferred by some industries due to their reduced installation cost and easier maintenance. However, spherical

Fig. 1 Spherical pressure vessel

tanks (or spherical pressure vessels) are recommended to store great amounts of highly pressurized fluids, such as liquefied petroleum gas (LPG) because mainly it has no corners which weaken the structure (Fig. 1).

The inspection of spherical pressure vessels is a hazardous and complex task due to the unhealthy environment (especially, inside the tank), size of the inspection area and mainly to the structure height (about 18m). Therefore, an automation process by robot is highly recommended to ensure a reliable and accurate inspection. The robot must cover the entire tank's surface and detect the environment's inconsistencies during the autonomous inspection task. In this context, inspection robots are designed to navigate in several planes, including the Earths surface perpendicular ones, climbing on outside and inside tank surfaces. Thus, the gravitational force acts as a highly important disturbance and cannot be neglected during the conception of the mobile inspection robot.

The robot localization through the environment is crucial to a successfully inspection because when a surface fault is detected, its exact position must be saved to allow a posteriori depth analysis of flaws and maintenance. Spherical pressure vessels are fully dark, with no landmarks and no different faces (from any viewpoint, the robot will see a semi-sphere). The most appropriate perception source for this kind of environment are the rangefinder sensors, because they are active exteroceptive sensors. All these facts turn the robot localization a hard task that can be carefully implemented.

Inspection task must be accurately planned to ensure that the inspection probe evaluates the whole surface, without missing any areas. However, the planning of an inspection path without detailed tank information, such as dimensions and localization of the robot, becomes a very complex or even insoluble task.

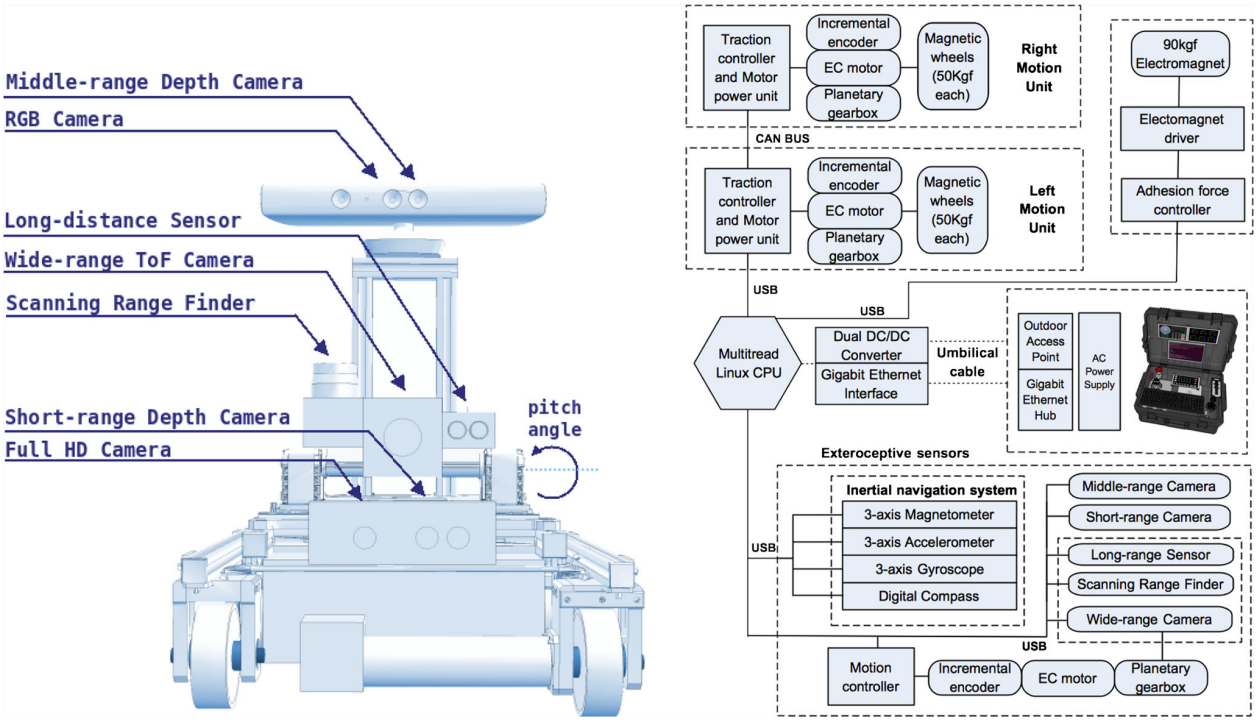


Fig. 2 Autonomous Inspection Robot (AIR-1)

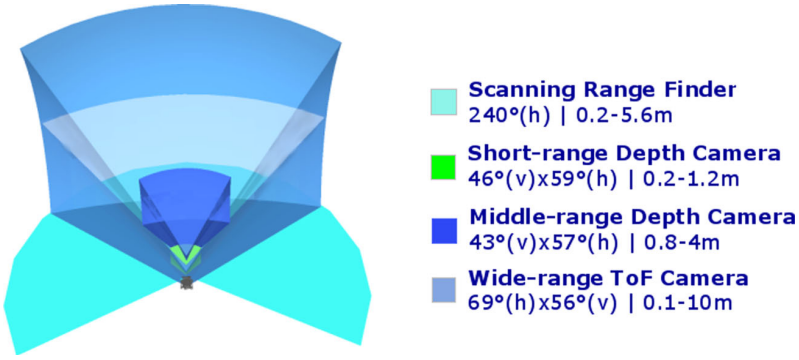


Fig. 3 Perception sources with its fields of view and ranges

3 Autonomous Inspection Robot

The Autonomous Inspection Robot (AIR) developed at UTFPR is designed to safely carry electronic and NDT components in order to inspect big metallic surfaces such as spherical pressure vessel shown in Fig. 1. For this, it is expected that the robot will carry also umbilical cords with length up to 20m. Besides, the robot also carries monitoring systems needed for its own navigation. The first UTFPR robot prototype (AIR-1) has a mechanical structure based on two parallel sets of fixed magnetic wheels (not steerable). Each wheel set is linked by a v-belt where only two motors are responsible to ensure torque for four magnetic wheels, i.e., the robot has two controllable degrees of freedom, as seen in Fig. 2.

Magnetic wheel sets are misaligned to overpass small obstacles (as weld beads) without decreasing their adhesion force. Each wheel consists of a set of two ring shaped neodymium magnets positioned between two steel disks and attached by screws with low magnetic permeability. A high hardness polyurethane rubber covers the bonded set. Its magnetic force is approximately 45kgf. The robot weighs 12kg (without NDT and monitoring systems) and the umbilical cord weights 0.4kg/m. The adhesion system was designed to provide a secure adherence between robot and surface during navigation and stopping to inspection. For this, a neural network analysis of magnetic field to the intelligent recognition of adhesion disturbances and to prevent robot overthrow is developed using an additional electromagnet during unsafe situations (wheels detachment). More details about the robot's mechanical design, wheel's adhesion and magnetic force analysis can be found in [13–17].

AIR-1 has a set of perception systems, composed by five main sources, as shown in Figs. 3 and 4 where the field of view and range of each sensor are displayed. A pair of depth cameras is used for environment mapping and obstacle detection. The first one is fixed in front of the robot (*short-range depth camera*) and the other is fixed in a structure above robot (*middle-range depth camera*).

A subset of perception sources is mounted on a mobile platform that can be rotated around pitch axis, improving sources sensing range and their precision (Fig. 4). One of them is a *long-range laser finder* (up to 70 meters with precision of millimeters) used to measure the relative distance between robot and environment, allowing the tank parameters estimation. A light detection and ranging sensor (*scanning range finder*) is used to detect any obstacle during navigation and to measure the lateral features of the environment. A time-of-flight industrial camera (*wide-range ToF camera*) provides 3D high-resolution image with wide field-of-view and it is used to improve the environment recognition.

4 Inspection Planning of Spherical Tanks

The inspection of storage tanks is only started after a rigorous planning that requires some specific information about the tank's topology. The robot must perform several

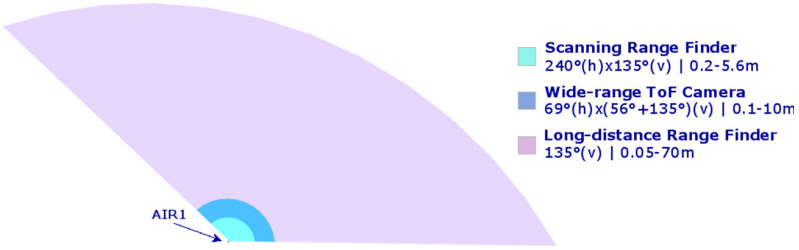


Fig. 4 Perceptions sources with improved range

procedures to understand the environment around it and prepare the inspection task, as seen in Fig. 5, where the robot can be operated in autonomous and non-autonomous mode. In non-autonomous mode, the operator manually controls the inspection robot (*in position/velocity mode*) or the operator delimits a specific fault to superficial or depth reinspection (*return to a detected fault*).

The robot can be inserted in any position of the storage tank and needs to perform an autonomous search for a standard observation point. From this special point, the robot is able to find the crucial features allowing a realiable estimation, based on computer vision techniques. After that, the robot performs the tank data measuring and improves this measure by fusing the multiple perceptions sources. The environment features are estimated based on the environment measures, considering the robot orientation and its position at the standard observation point. Thus, the environment topology can be estimated with basis on relative position of measured points. According to the tank's design standards, it is also possible to estimate some tank's specific components such as weld beads and plates. In the following, an occupancy grid is build up, including all storage tank's main features that are perceived and/or inferred by the robot. Finally, a prediction reliability analysis is carried out by an artificial neural network. After all these procedures, the robot will have a previous knowledge about the tank that will be inspected and a reliable inspection path can be planned. Preliminary results of these approach are discussed in [18].

4.1 Autonomous Search for the Standard Observation Point

AIR-1 robot needs to localize two corners in a weld bead to perform a reliable estimation of the storage tank. These two points compose a *standard observation point*.

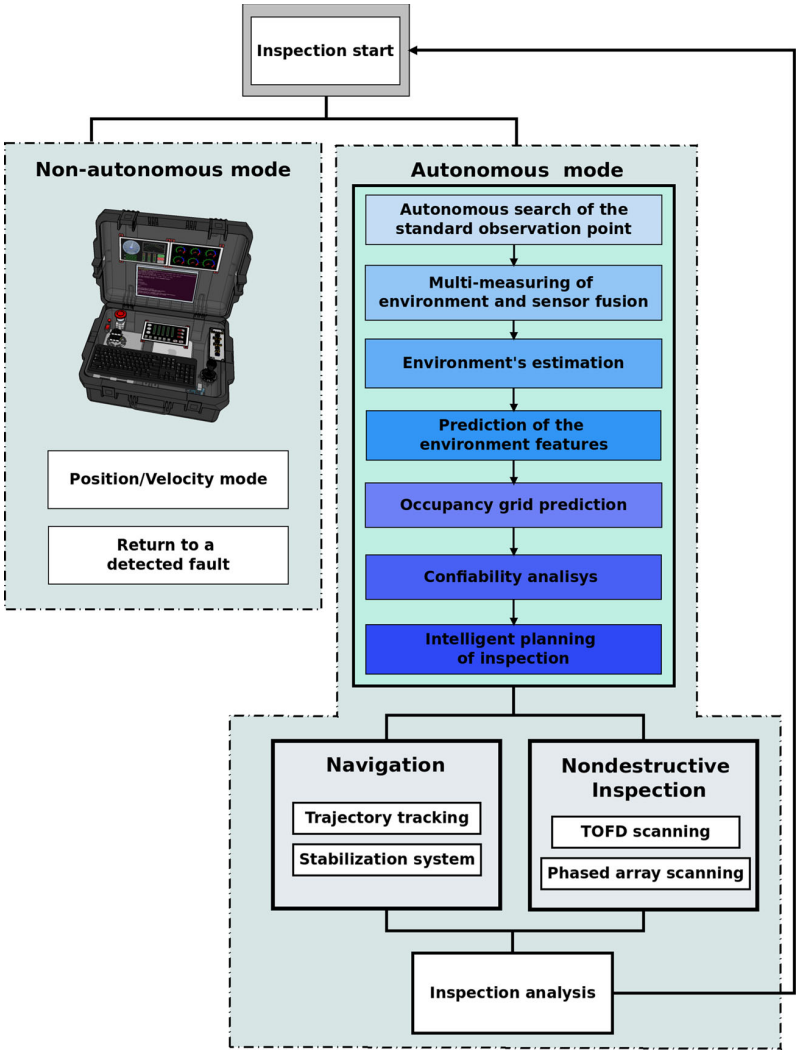


Fig. 5 Approach for the intelligent inspection planning

The characteristics of these corners (such as distance and angle between them) are crucial to the prediction. The search for the standard observation point must be executed autonomously without any operator interference, because the inspection robot can be initially placed in any tank position.

The robot uses two depth sensors for this task, *wide-range ToF camera* (Heptagon SR4000) and *middle-range depth camera* (Microsoft Kinect), which are aligned to expand the depth perception through of motion of TOF camera in 15 degrees about pitch to enlarge the vertical sensing. These sensors operate at reduced light or without light, which is ideal because the inspection environment is almost totally dark and the use of powerful lights must be avoided due to the risks of sparks generation by heating since the atmosphere inside the tank is highly explosive.

The procedure to implement the autonomous search are described in Fig. 6, where the robot performs a sequence of motions to find and align the weld beads corners based on its depth vision. The first step is to achieve the correct orientation with the weld beads. The robot performs this action through *wide-range ToF camera* with 10 meters of range. The depth perception is obtained in 3D points (point cloud) and it is converted to grayscale image to find corners through *Harris-Stephens algorithm*, as discussed in [19]. If no corner is found, the robot turns to the right and repeats this step until finding any corner. When the first corner is found, the robot turns to the opposite side to detect the second corner.

As soon as the robot is frontally aligned with a pair of weld bead corners, the robot moves forward until it detects and focuses the weld bead with its *middle-range depth camera*. This step is identified as *proximity correction* in Fig. 6. The weld bead recognition is achieved through the use of *Roberts filter* [20].

The heading correction is made to ensure the orientation between the robot and the target (i.e., weld bead corners)

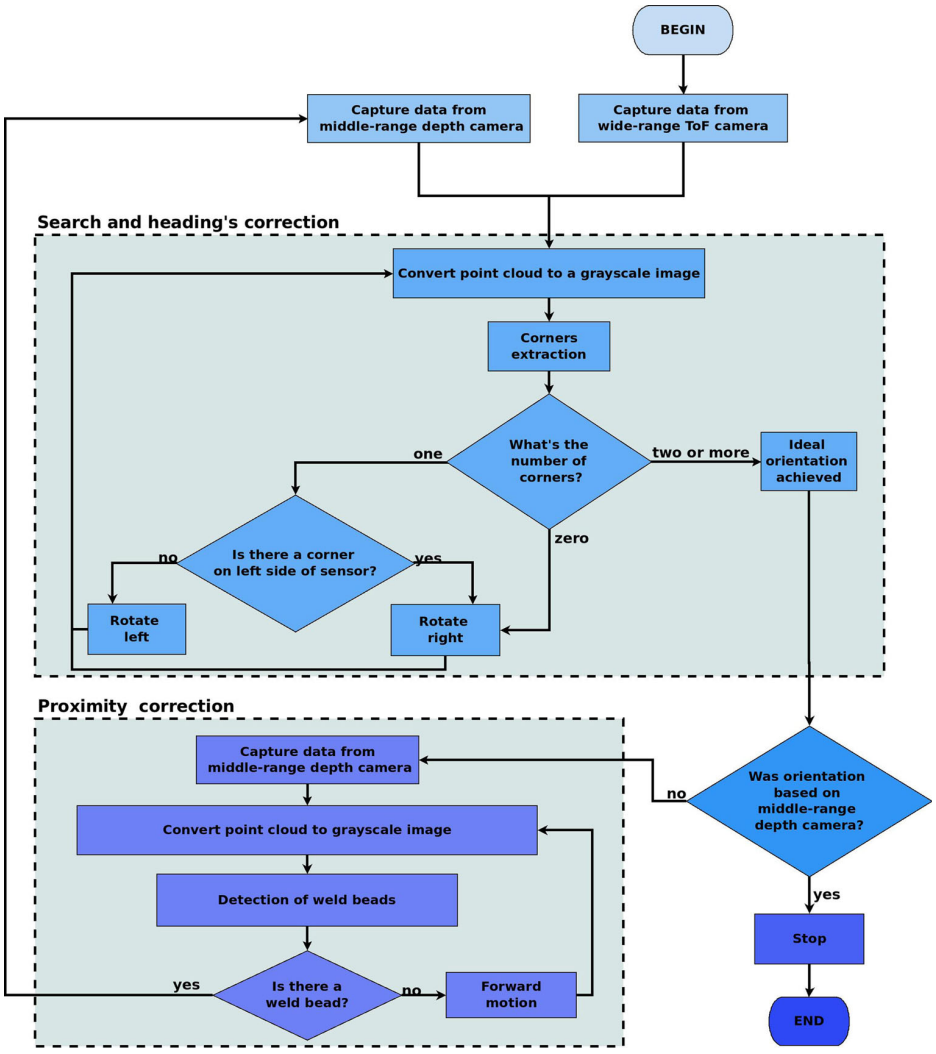


Fig. 6 Flowchart rules for autonomous search of the standard observation point

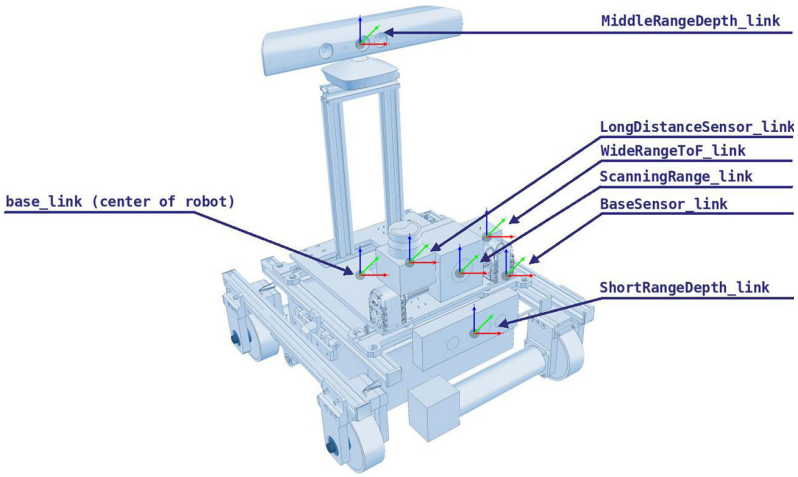


Fig. 7 Coordinate systems of perception sources

after the forward motion inside a spherical pressure vessel. At this time, the robot must look for a pair of weld bead corners with its *middle-range depth camera*. The difference of resolution between these perception sources can generate false corners, which must be avoided to achieve the correct alignment.

The point cloud from *middle-range depth camera* is also converted to a grayscale image and a *Roberts filter* is applied to recognize the weld beads. However, it is necessary to apply a *morphological open function* [21] to remove small noises from the obtained images. At last, the *morphological dilation function* [21] is applied so that the weld beads become larger and more visible, allowing the correct recognition.

4.2 Perception of the Environment and Sensor Fusion

The acquired points of the environment obtained by multi-measuring are related to different coordinates system, one for each perception source. Thus, they are represented by different coordinate systems and it is necessary to transform all points to the same reference system. In this case, the robot's reference system (fixed on its center), as seen in Fig. 7.

The proposed approach is totally independent of the robot's position and orientation, because it uses the inertial measurements information to represent the transformation between robot's center and environment. All information coming from perception sources is translated to be expressed by the same coordinate system, in this case the robot's center. Thus, this information can be analyzed and fused because it represents the same measure, regardless of its source.

The Autonomous Inspection Robot (AIR) has five kinds of 3D perception sources, each one dedicated to measure a specific condition, as discussed previously in Section 3. However, these systems present intersection of various measurement sections, where the measurements of different sources are overlapped. This condition can occur in the border measurements of perception sources or near an obstacle. In this case, two or more sources overlap because there is an obstacle near the robot as shown in Fig. 8.

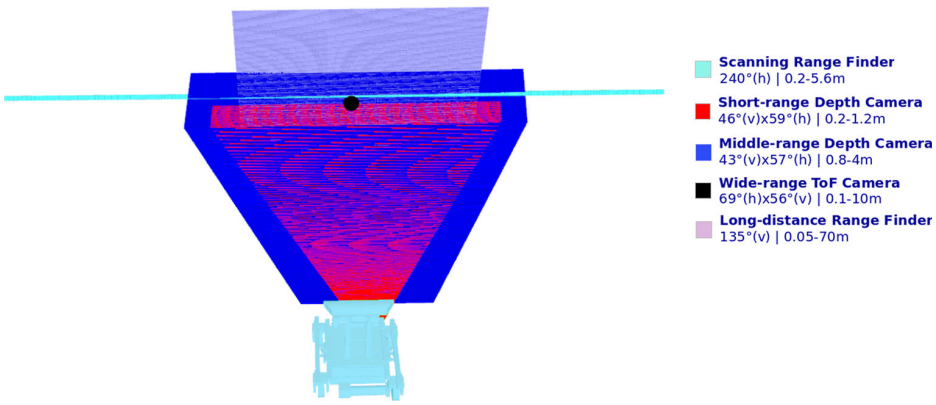


Fig. 8 Overlay of the perception sources

Table 1 Specifications of perception's sources

	Scanning Range Finder	Short-range Depth Camera	Middle-range Depth Camera	Wide-range ToF Camera	Long-distance Range Finder
Range	0.02 to 4m	0.2 to 1.2m	0.8 to 4m	0.1 to 10.0m	0.05 to 70m
Minimum Error	3cm	*ns	*ns	1cm	0.2cm
Maximum error	3%	5%	5%	1%	0.3%
Accuracy (in centimeters)					
0.5m	3	2.5	*ns	1	0.2
	3	5	5	1	0.2
2.5m	7.6	*ns	12.5	2.5	0.75
4m	12	*ns	20	4	1.2
5.5m	*ns	*ns	*ns	5.5	1.65
7m	*ns	*ns	*ns	7	2.1
9m	*ns	*ns	*ns	9	2.7
11m	*ns	*ns	*ns	*ns	3.3

$*ns$ is not specified

The fusion of the overlapped measurements is performed to reduce the number of environment points without the loss of the overall precision. The adopted strategy to data fusion is based on the relevance degree (δ_n) of each measured point (P_i), reflecting its Euclidean distance to neighboring points $P_i, i = 1..n$. The new fused point (P_{nf}) is defined by:

$$P_{n_f} = \sum_{n}^{i=1} \frac{\delta_i}{\delta_T} P_i \tag{1}$$

where δ_T is the global relevance. It is worthwhile to note that the value of the relevance degree (δ_i) is related to the sensor accuracy and the measured point distance from robot, as presented in Table 1.

The δ_i value of each perception data is applied as input in sensor fusion procedure where its value varies from 0 (least relevant) to 1 (most relevant). The rules are designed to consider an ideal region of each source of perception, where the error is smaller in relation to the other sources.

Table 2 shows the adopted rule basis developed from Table 1 information about distance and error for each perception source. It is possible to obtain an attribute value for each point depending on its origin and distance. The relevance is given according to the attribute, where VL (very low) has a relevance degree equal to 0.2, L (low) has a relevance degree equal to 0.4, R (reasonable)has a relevance degree equal to 0.6, H (High) has a relevance degree equal to 0.8 and VH (very high) has a relevance degree equal to 1.0.

4.3 Environment's Estimation

The estimation of spherical pressure vessel topology is carried out by minimizing the sum of the squared differences between the estimated and observed data, as in [22].

$$E = \sum_{i=1}^{m} r_i - r \tag{2}$$

where, r is the initial value of the estimated radius and r_i is the observed value.

The radius r can be obtained by

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2} \tag{3}$$

and the observed radius r_i can be computed as

$$r_i = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2 + (z_i - z_c)^2} \tag{4}$$

Table 2 Rules and concepts

	Perception source \ Distance [m]	>1	2.5	4	5.5	7	<10
1	Scanning Range Finder	R	R	R	VL	VL	VL
2	Short-range Depth Camera	L	VL	VL	VL	V.L	VL
3	Middle-range Depth Camera	VL	L	L	VL	VL	VL
4	Wide-range ToF Camera	H	H	H	H	H	VL
5	Long-distance Range Finder	VH	VH	VH	VH	VH	VH

where x_c, y_c and z_c represent the center of the sphere; x, y and z represent any point on the sphere; and x_i, y_i and z_i represent a point on sphere with index i.

From Eqs. 3–4, it is possible to rewrite Eq. 2 as

$$E = \sum_{i=1}^{m} \left((x_i - x_c)^2 + (y_i - y_c)^2 + (z_i - z_c)^2 - r^2 \right)^2 \quad (5)$$

and a coordinates estimation of the sphere center is obtained by solving

$$C = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = arg\ min\ E \quad (6)$$

The center of spherical pressure vessel is computed by LS approach as:

$$C = ((A'A)^{-1}A'B) \quad (7)$$

where

$$A = 2 * \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix} \quad B = \begin{bmatrix} {x_1}^2 & {y_1}^2 & {z_1}^2 \\ {x_2}^2 & {y_2}^2 & {z_2}^2 \\ \vdots & \vdots & \vdots \\ {x_n}^2 & {y_n}^2 & {z_n}^2 \end{bmatrix} \quad (8)$$

The sphere radius sr is estimated by a mean of all computed radios (Eq. 3), defined as

$$sr = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (x_i - x_c)^2 + (y_i - y_c)^2 + (z_i - z_c)^2} \quad (9)$$

4.4 Prediction of the Environment Features

There are 4 types of spherical pressure vessels according to the plate's arrangements [23] used to build them:

- ***expanded cube, square segment, or soccer ball type*** is commonly applied to small spheres with less than about 6 meters in diameter;
- ***meridian, orange peel, or watermelon type in 3-course version*** are related to spheres with diameter between 6 and 9 meters;
- ***partial soccer ball type*** is designed to spheres with diameter between 9 and 18 meters;
- ***meridian, orange peel, or watermelon type in 5-course version*** is applied to big sphered with more than 18 meters in diameter.

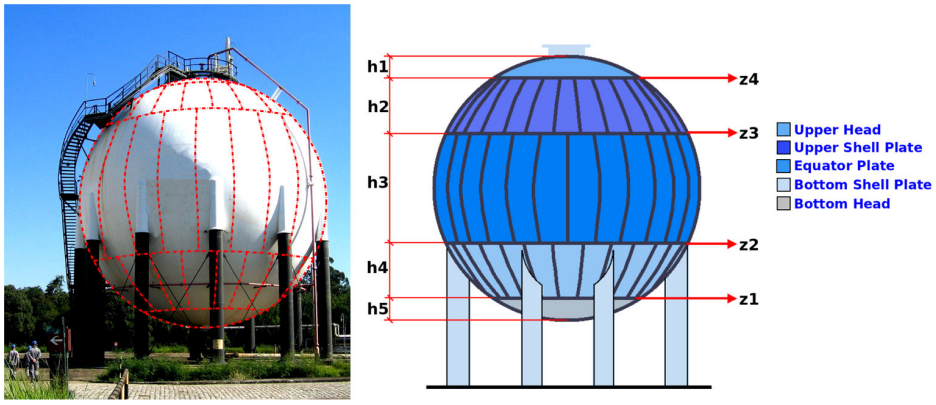
The spheres used in Brazilian petroleum refineries have more than 18 meters of diameter and they are built according to the manufacturing standards to spherical pressures vessels [24–27]. These spheres are meridian type in 5-course version. These types of sphere vessels are subdivided equatorially in five sections (as seen in Fig. 9), according to the distance between two plates (h_N), as *Upper Head* (h_1), *Upper Shell Plates* (h_2), *Equator Plates* (h_3), *Bottom Shell Plates* (h_4) and *Bottom Head* (h_5).

Considering at this time, the robot has reached its standard observation point, in front of two weld beads corners (Section 4.1) and it knows an estimation of the center and radius of spherical pressure vessel (Section 4.3). Thus, the height of each equatorial section (h_N) and the position of each equatorial section in relation to the tank's base (z_N), as shown in Fig. 9, can be calculated. The equatorial sections values computation considers the design and fabrication standards for pressure vessels, which specify the equilibrium of volumes between these areas [23], as

$$\begin{aligned} h_3h_4 &= z_4 - c \\ h_5 &= sr - h_3h_4 \\ h_4 &= 0.54065 * h_3h_4 \\ h_3 &= 0.91870 * h_3h_4 \\ h_2 &= h_4 \\ h_1 &= h_5 \end{aligned} \quad (10)$$

where, sr is the radius of spherical pressure vessel (given by Equation 9) and c is the center of the tank (estimated by Eq. 7).

Fig. 9 Spherical pressure vessel



(a) beads/plates highlights. (b) Sections schematics.

After these procedures, it is possible to estimate the position of all equatorial section in relation to the tank base (z_N), as

$$\begin{aligned} z_3 &= z_4 + h_4 \\ z_2 &= z_3 + h_3 \\ z_1 &= z_2 + h_2 \end{aligned} \tag{11}$$

where, z_4 is the position of observed equatorial section, estimated by autonomous search for the standard observation point procedure (Section 4.1), and h_n is the size of the plate as shown in Fig. 9.

4.5 Occupancy Grid Prediction

After environment estimation and features prediction, the occupancy grid can be now predicted, aiming to create all surfaces of the inspection environment. This task is based on the uniform distribution of voxels in the predicted occupancy grid, ensuring an equal spacing between each voxel and all its neighboring voxels. The occupancy grid is estimated to wrap the inspection environment, creating a regular grid in tridimensional space that represents the predicted tank, as shown in Fig. 10.

The procedure to determine the voxels' uniform distribution, in predicted occupancy grid, is based on environment prediction where its main features are the estimated center c (Eq. 7) and radius sr (Eq. 9). The occupancy grid resolution is determined by the size of NDT probe, identified as voxel size vs. Algorithm 1 describes all steps need to determine the voxel distribution.

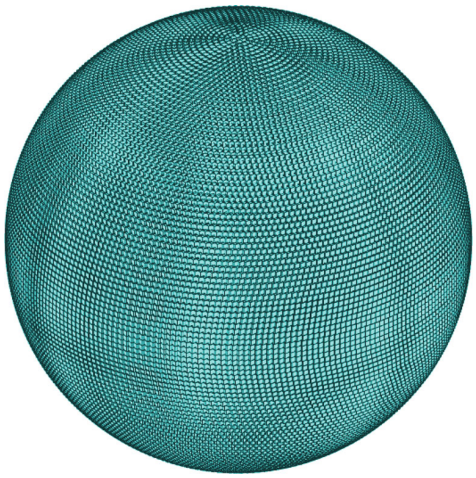


Fig. 10 Example of occupancy grid prediction of inspection environment

Algorithm 1: Prediction of occupancy grid.

Input: $c[x, y, z]$ is the vector of coordinates of tank's center [m]; sr is the tank radius[m]; vs is the voxel size [m].

Output: $o_p[m][3]$ is a $m \times 3$ matrix of the scatter points on the sphere.

```
/* length of a section of sphere
   (i.e., a slice or circle) */
```
$le \leftarrow 2 * \pi * sr$

```
/* ratio of the sphere section
   length per voxel size */
```
$np \leftarrow le/vs$

```
/* counter */
```
$i \leftarrow 0$

```
/* defines the spacing in the Z-axis
   */
```
for $\theta \leftarrow (-\pi/2) : \frac{2\pi}{np} : (\pi/2)$ **do**

```
    /* computes the next point on the
       axis Z */
```
    $z \leftarrow (\sin(\theta) * sr) + c[z]$

```
    /* radius of sphere's section in
       the axis Z */
```
    $rc \leftarrow \sqrt{(2 * sr) - (2 * (sr - (z + sr)))}$

```
    /* defines the spacing in the axes
       X and Y */
```
    for $\phi \leftarrow (-\pi) : \frac{2\pi}{np} : (\pi)$ **do**

```
        /* computes the coordinate of
           voxel on X axis */
```
        $o_p[i, x] \leftarrow (\cos(\phi) * rc) + c[x]$

```
        /* computes the coordinate of
           voxel on Y axis */
```
        $o_p[i, y] \leftarrow (\sin(\phi) * rc) + c[y]$

```
        /* computes the coordinate of
           voxel on Z axis */
```
        $o_p[i, z] \leftarrow z$

        $i \leftarrow i + 1$

    end

end

```
/* returns the matrix with
   coordinate of all voxels */
```
return $o_p[m][3]$

4.6 Confidence degree

After the occupation model is built, the modeled environment is estimated and decomposed into a group of

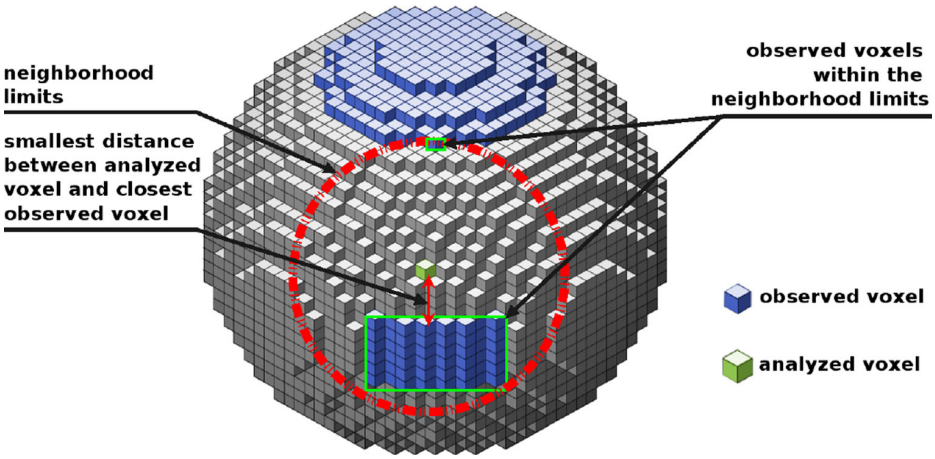


Fig. 11 Example of occupancy grid prediction of the inspection environment

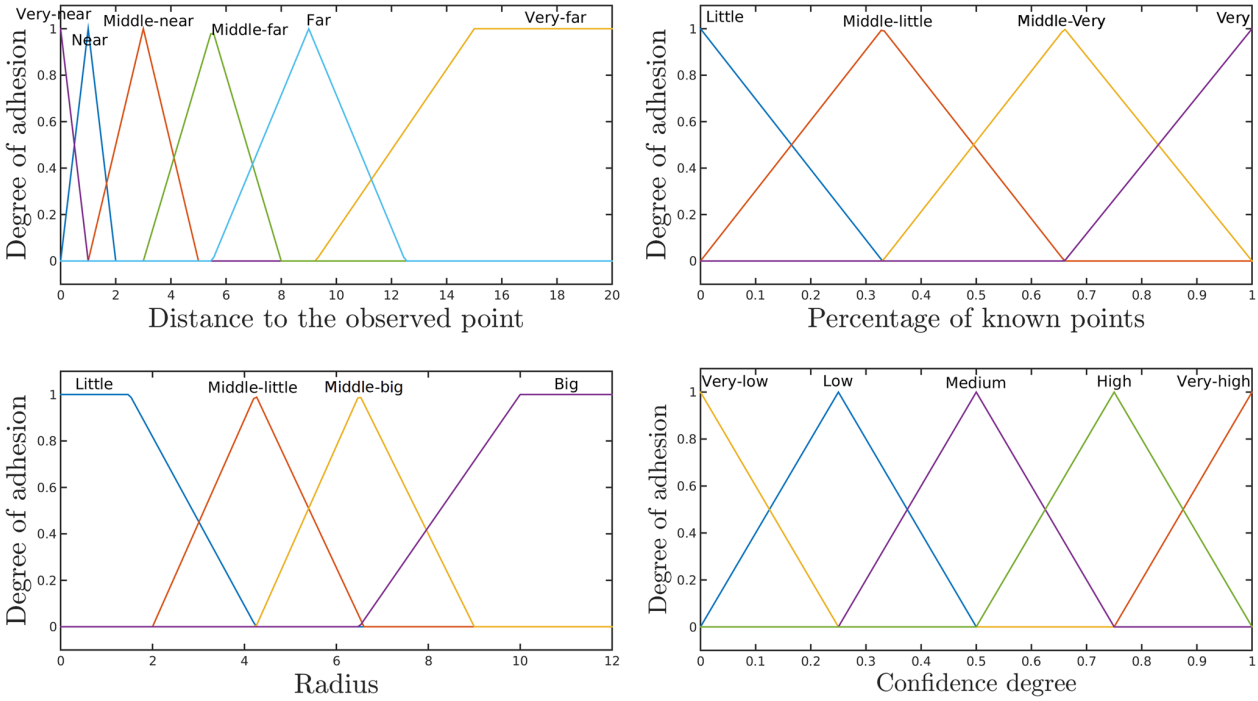


Fig. 12 Inputs and outputs of the Fuzzy system

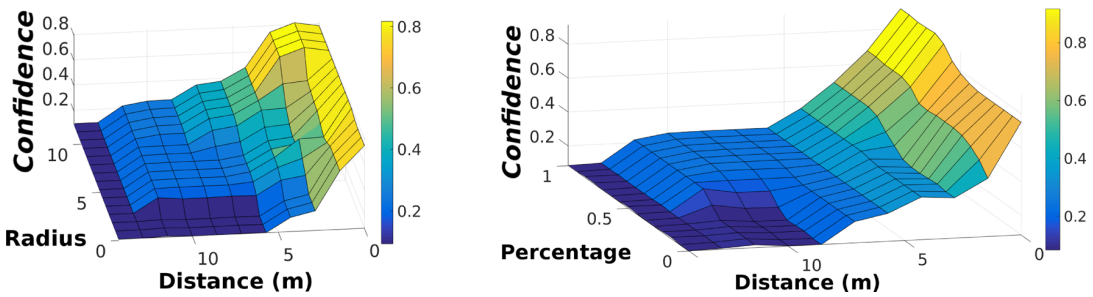


Fig. 13 Surface of Fuzzy rules

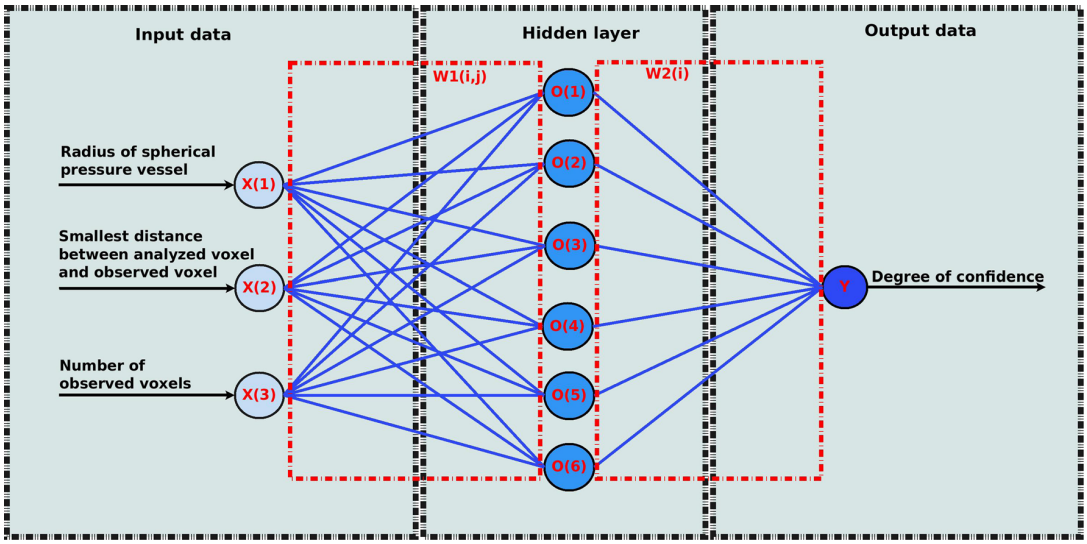


Fig. 14 ANN architecture to confidence degreee estimation

three-dimensional points (voxels) uniformly distributed on tank surface, creating the occupancy grid. However, most of these points are estimated and not observed, introducing uncertainties in environment estimated model. The confidence degree is the measurement of prediction's uncertainties which is directly related with its distance of measured voxels. Thus, a Fuzzy model is designed to quantify this uncertainty, in terms of an estimation confidence degree to each estimated voxel in the occupancy grid. A deep discussion about Fuzzy rules can be seen in [28, 29].

The Fuzzy inference of the voxel confidence degree uses as input the percentage of observed voxels in the neighborhood, the neighborhood of the analyzed voxel, the smallest distance among the analyzed and observed voxels, and the radius of the spherical pressure vessel. Figure 11 illustrates the approach, where the green voxel represents the point being analyzed and the voxels in blue represent the observed points through perception sources. The red circle shows the neighborhood limits used to identify the percentage of known points. Furthermore, the percentage

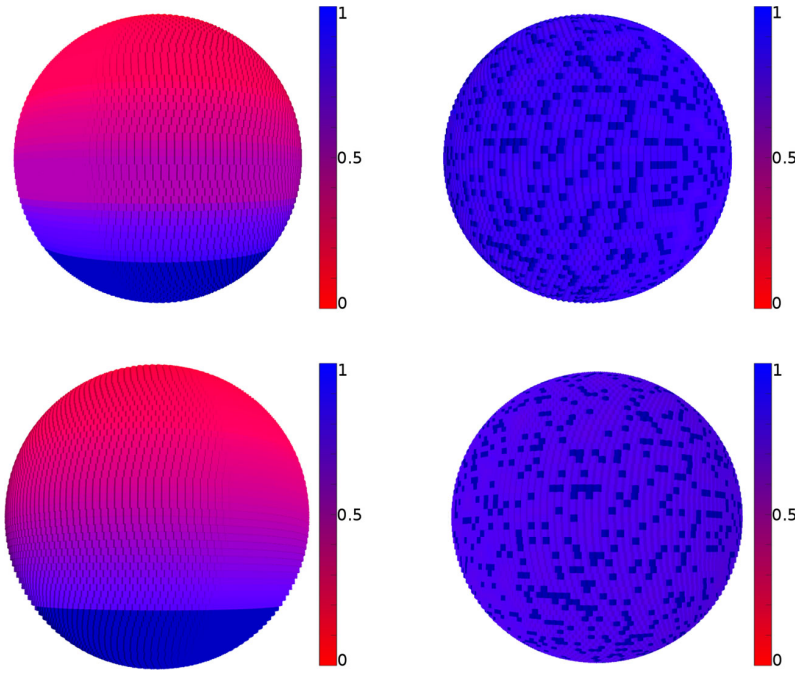


Fig. 15 Comparison between Fuzzy and ANN in a 10 meter sphere with known points grouped and distributed. Top left: Fuzzy with grouped points. Top right: Fuzzy with distributed points. Bottom left: ANN with grouped points. Bottom right: ANN with scattered points

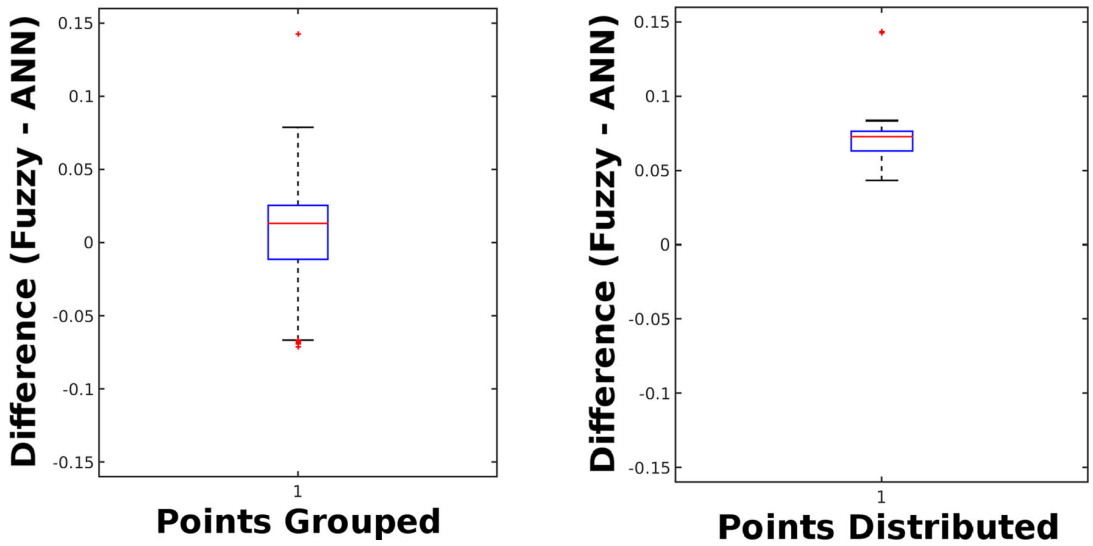


Fig. 16 A boxplot with the difference between confidence degree computed by the Fuzzy and ANN model for a 10 meters diameter sphere with 20% of the known voxels. Left: Grouped voxels. Right: Randomly distributed voxels

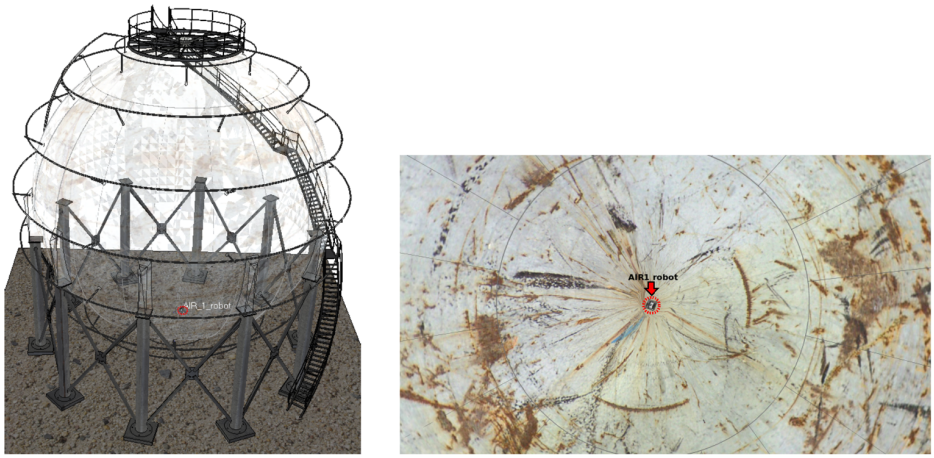
of observed voxels is the sum of the known points in the neighborhood divided by the total of known points.

The smallest distance between the analyzed voxel and the closest observed point is illustrated in Fig. 11 by a red arrow. If there are no observed points (blue square is null in Fig. 11) inside the neighborhood, the nearest point of the analyzed point is considered the only observed point and it is taken to compute confidence degree.

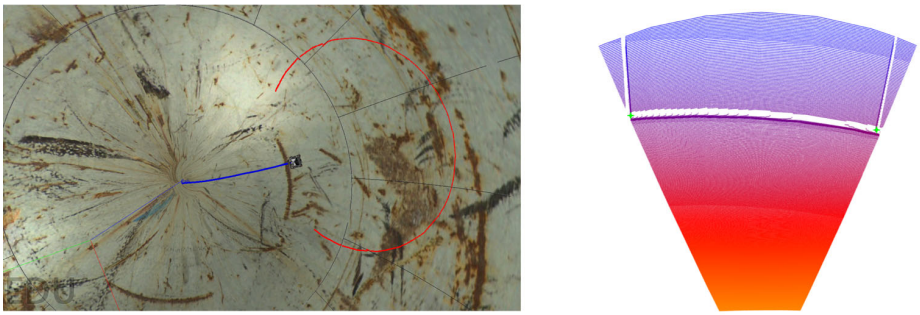
The membership functions to Fuzzy model are designed in such a way that the radius of the sphere influences the confidence degree. For example, in a large sphere, the degree of confidence decays slowly and over a larger region. In a small sphere, all points on the sphere will receive a high degree of confidence. Thus, the degree of confidence is decay along the sphere is directly related to sphere size. Figure 12 shows the input and output membership functions of the Fuzzy model while Fig. 13 presents the output surface mapping generated by the considered rule base. The proposed Fuzzy model has presented good results, however aiming to reduce computational cost, an *Artificial Neural Network* (ANN) was developed to reproduce the developed Fuzzy model behavior.

The ANN architecture, observed in Fig. 14, has 3 layers with 6 neurons in the hidden layer and the same input and output of the Fuzzy model. The inputs are represented by the vector X, the connection weights are represented by the

Fig. 17 Virtual scenario for validate the proposed approach



(a) Virtual pressure vessel. (b) Starting position.



(a) Achieving the standard observation point. (b) Point cloud.

Fig. 18 Search for standard observation point in virtual pressure vessel

matrix $W1$ and vector $W2$, the output of the hidden layer's neurons is represented by the vector O and the final output is represented by the variable Y.

The output neuron Y has a linear activation function (*purelin*), whereas the activation function of neurons $O(n)$ is the Tangent Sigmoid function (*tansig*). The output of neuron O is given by Eq. 12 while output Y is given by Eq. 13. More information about design and operation of artificial neural networks can be found in [30–34].

$$O(n) = tansing\left(\sum_{i=1}^{M}(X(i) * W(i,n) + bias)\right). \quad (12)$$

$$Y = purelin\left(\sum_{i=1}^{M}(O(i) * W(i) + bias)\right). \quad (13)$$

The experiments to train the neural network model were organized according to the quantity of observed points and their localization. These points are either grouped and placed in a specific sphere regions or they can be randomly scatered over the sphere surface. The data set is composed by 1%, 20%, 40%, 60% and 99% (5 data set for each grouped or scattered experiment) of observed points. Four pressure vessels with diameters of 6, 10, 14.6 and 18.25 meters are considered totalizing 40 experiments.

The set of voxels used to ANN training is composed by the points of all pressure vessels (539.410 points) minus the number of observed voxels (270.705 points), resulting in 268.705 points, where, 70% were used for training (188.793), 15% for validation (40.456) and 15% for test (40.456). The Mean Squared Error (MSE) of training step was 8.21x10-04. For the test step the MSE attained 8.16e-04 and to validation was 8.21x10. Figure 15 shows the comparison between Fuzzy and ANN models applied to a tank with 10 meters in diameter, with both grouped and scattered known points, where it is possible to observe that ANN model absorbs the Fuzzy model characteristics, generating similar results in both grouped and distributed points.

ANN model has a shorter runtime, while the Fuzzy model takes 0.05 seconds to compute the confidence degree to a voxel, ANN model takes 0.03 seconds. For a 10 meters diameter sphere, 7926 voxels must be analyzed, making the run time difference between the Fuzzy model and ANN model around 158.53 seconds. To validate the results obtained by ANN model, two *boxplot* built. These

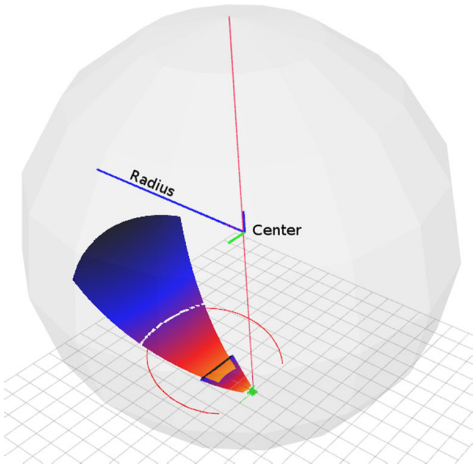


Fig. 19 Estimation of the pressure vessel

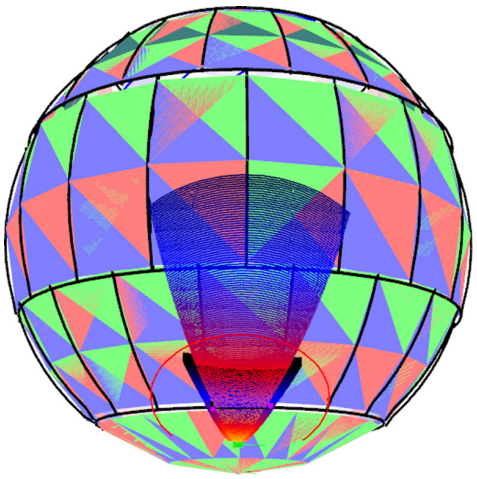


Fig. 20 Prediction of pressure vessel features

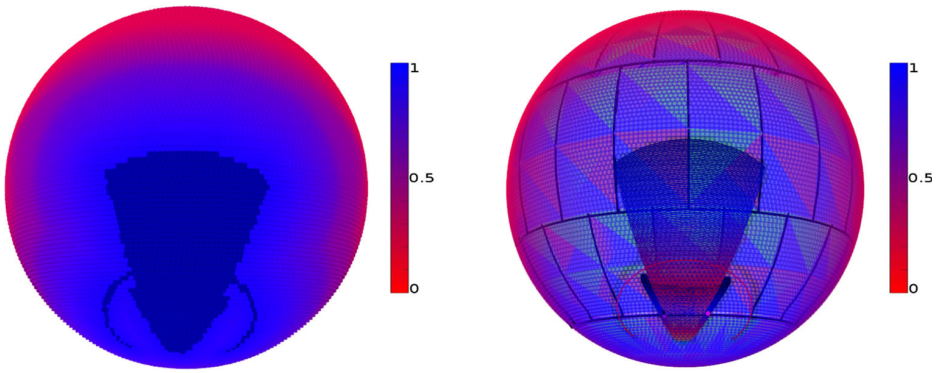


Fig. 21 Occupancy grid prediction and confiability analysis

(a) Occupancy grid. (b) Occupancy grid with tank features

plots shown in Fig.16, consider the difference of the degree of reliability obtained with both models to the above cited sphere. The first boxplot considers 20% of grouped known voxels and the other ones takes 20% of scattered knows voxels. These boxplot results confirm that RNA model can replace Fuzzy model with a good accuracy and a better result time.

5 Experiments in Virtual Pressure Vessel

Simulation experiments were carried out to validate the complete proposed approach. A virtual spherical pressure vessel, considering all characteristic of a real sphere, was implemented into *Virtual Robot Experimentation Platform* (V-Rep) [35] (Fig. 17). The virtual robot was designed with the same perception sources and magnetic adhesion skills as the real robot.

During the automatic search for the standard observation point, discussed in Section 4.1, the robot moves around looking for two corners of weld beads. When the robot finds the corners, it moves to reach this position in the middle of the shorter distance between these corners, by using the *wide-ranging ToF camera*. After that, robot moves forward in a straight line until its *middle-range depth camera* finds the pairs of corners and then a fine angular robot's position correction is carried out. The result of this procedure is shown in Fig. 18.

When the robot achieves the correct observation point, the procedure of *multi-measuring of environment and sensor fusion* is started. The robot measures the environment with its several perceptions sources (Fig. 19). In virtual experiment, the radius is estimated obtaining 9.105 meters against the real value of 9.125 meters, that is an error of 0.021%, and the estimated center is [0.098,-0.088,9.037] meters with only 1.47% of error.

The components of computed pressure vessel are now predicted and used to create a detailed estimation of storage tanks characteristics including its specific features, as well as its plates and weld beads, as discussed in Section 4.4. All calculations are based on the location and distance between the two weld beads corners and this result can be seen in Fig. 20.

After estimation of main environment features, it is possible to predict the occupancy grid of pressure vessel, wrapping the whole tank with a voxel mesh, as discussed in Section 4.5. However, these voxels have different confidence degree, which varies in accord with their distance of a observed known voxel, as detailed in Section 4.6. Each voxel receives a confidence degree with range from 0 (unobserved) to 1 (observed). The Fig. 21 shows the result of this approach in the virtual pressure vessel, where each voxel has color varying from blue (cold) to red (hot), where the hottest temperature represents the lower confidence index.

6 Real Experiments

A preliminary experiment has been carried out with real Autonomous Inspection Robot (AIR-1) in a square room

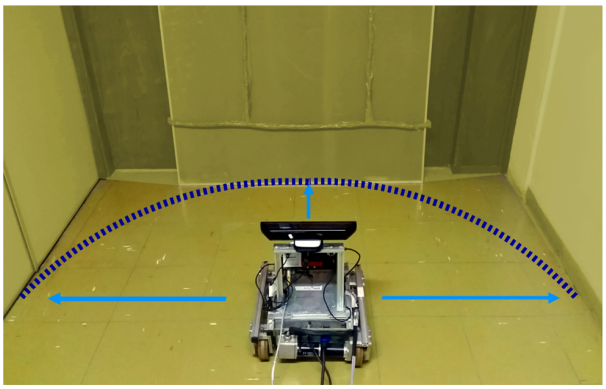


Fig. 22 Scenario of real experiment with AIR-1

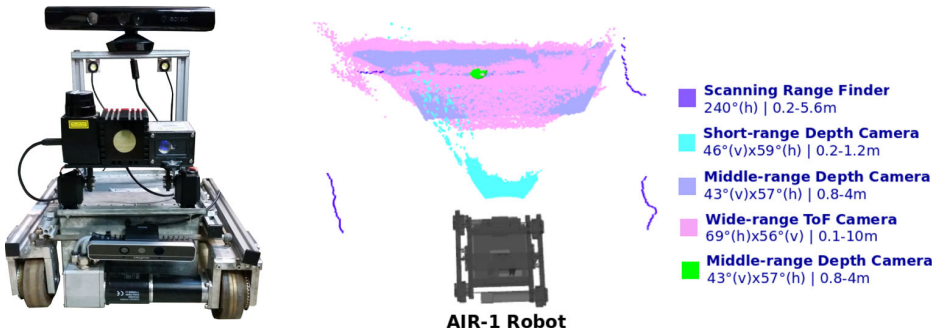


Fig. 23 Experiment with real inspection robot

(a) AIR-1 (b) Perception data

emulating a spherical pressure vessel. The weld beads are placed in an appropriate location of this room to promote a similar scenario with a real storage tank, as seen in Fig. 22, where the room has a diameter of 2.02 meters. In this experiment, the points acquired from perception sources were processed to reduce the error in sphere estimation, i.e., the points related to room corners are discarded because these features are not present in a real tank. A sample of the real perception data acquired by the Autonomous Inspection Robot in experimental scenario can be seen in Fig. 23.

The AIR-1 has estimated a small spherical tank, due to experiment characteristics, with 1.334 meters in radius. The tank features (as weld beads and plates) are also estimated based in the pair of weld beads corners, found by the automatic search, as seen in Fig. 24a. The tank occupancy prediction grid is also performed, where each voxel is analyzed to determine its confidence degree. The obtained occupancy grid is illustrated by Fig. 24b. The difference between the diameter of the predicted sphere and the diameter of the room, 64 centimeters (24.28%), refers to the curvature of the sphere. The diameter in the Bottom Head (Fig. 9) is smaller than in the middle of the sphere, as shown in Fig. 25.

7 Comparison with Classic Navigation Approach

This work presented a unique technique of mapping environment for NDT inspection by mobile robots without the need to navigate the whole sphere, besides identifying characteristics of the environment as the weld beads and position of the metal plates. A comparison with classic navigation method is presented aiming to emphasize the skills of proposed approach.

The technique used as reference is presented in [5]. The data captured by the robot perception sources are converted into structures known as octree, are the environment map is created only during robot navigation. This reference method is fully dependent of environment observation and the full

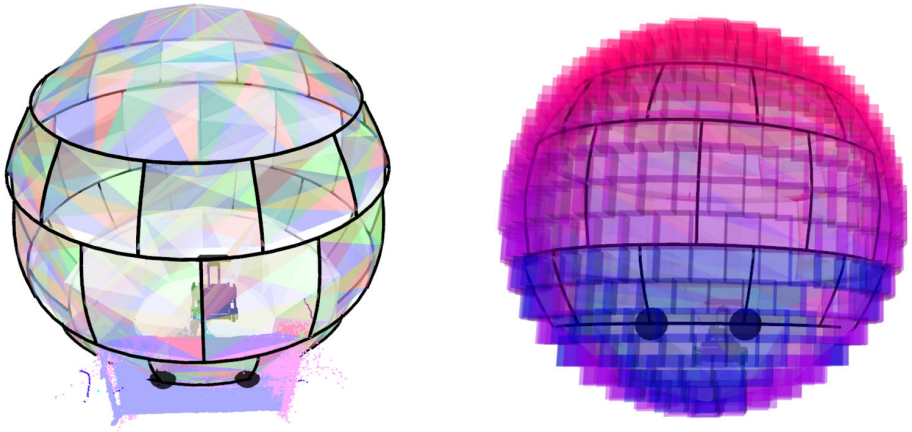


Fig. 24 Results of experiment with real AIR-1

(a) Estimation of tank features (b) Confidence degree analysis of predicted occupancy grid

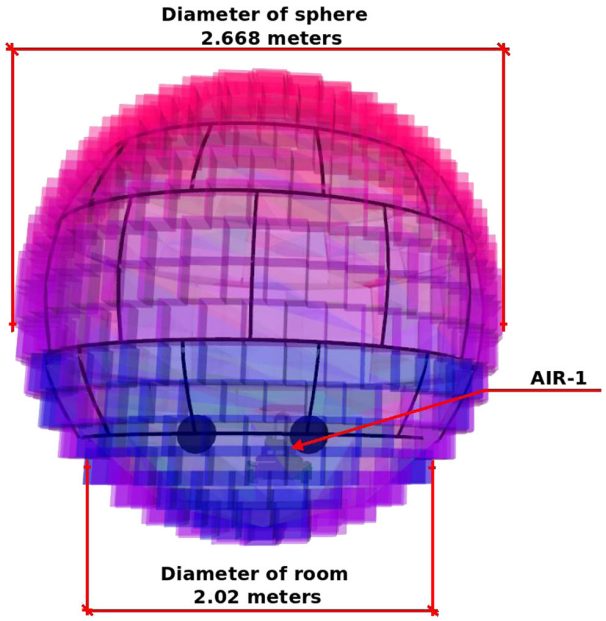


Fig. 25 Result of experiment with real AIR-1, difference between the diameter value of the room and the predicted sphere

resulting map is achieved only after the robot navigates the entire environment.

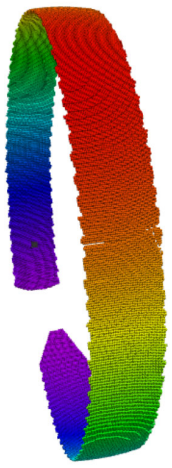
The comparison scenario is a spherical vessel with diameter of 18.25 meters and an area of 1046.346 square meters. For both methods,the robot always start the mapping in the bottom center of the sphere (*Bottom Head*, see Fig. 1). The robot will perform the navigation over the virtual tank during 25 minutes. After this the total distance traveled by the robot, percentage of observed area, observed aspects and actions performed by robot with both methods will be computed and analyzed.

With the traditional method, the robot needs to navigate the entire spherical vessel to perform the mapping. At the end of 25 minutes, the robot walked a total of 52 meters, and observed a total of 165 meters^2 of the sphere, corresponding to only 15.769% of the whole sphere's surface. The Fig. 26 shows the path traveled by the robot over the sphere, as well as the generated map.

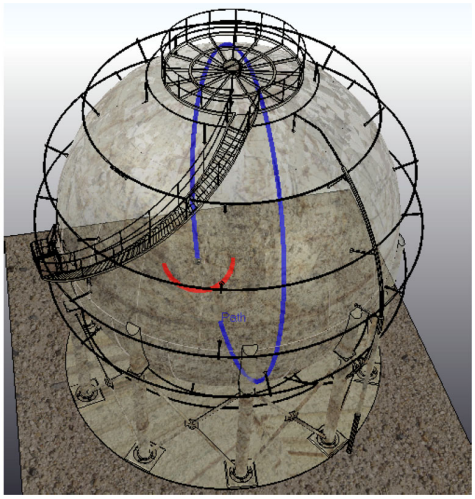
With the proposed method, covered a total of 3.930 meters and performed a pseudo-observation of 100% of pressure vessel features. The followed trajectory and generated map by the proposed approach are those shown in Figs. 17, 18, 20 and 21. This successful mapping is only possible because relevant information such as the characteristics of the tank are taken into account in the prediction of the map during navigation, making the method efficient for mapping of spherical storage tanks.

In classic method, robot should cover all tank surface (*locomotion effort*) to observe whole environment and to registration its observations (*processing effort*) in a map accord to its localization global. In proposed method, robot not needs to perform whole observation to identify all structural landmarks and to predict entire pressure vessel surface. Thus, the motion is reduced (*locomotion effort*) but the processing are increased. However, more effective results are attained during less execution time with the propose method. The Fig. 27 presents a diagram comparing the robot's processing and locomotion times, for both method where it is possible to observe that the proposed method uses only 1.85 minutes of total time to locomotion

Fig. 26 Followed trajectory and generated map after 25 minutes of navigation using classic mapping technique



(a) Generated map



(b) Navigated trajectory

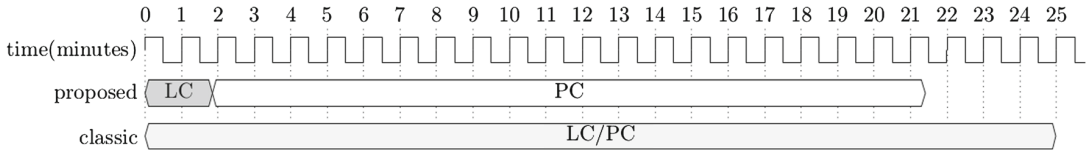


Fig. 27 Time description of robot actions in environment mapping [LC = locomotion and PC = processing]

and the remainder for processing. The processing time can be significantly reduced it a more powerful processing system is used or a parallel version of the method is implemented as discussed in preliminary results of [36].

The Table 3 presents the comparison between the two techniques, where is evidenced which the classic method not identify the tank features even after observation of entire environment.

8 Conclusion

This work has discussed an approach to identification and prediction of environment features aiming the reliable planning of inspection path. This method is focused on automatic inspection of spherical pressure vessel by an autonomous climbing robot.

The proposed approach is designed in several steps. First, the robot automatically finds and moves to a correct viewpoint to environment identification. The second step consists of the environment measure by a group of distinct perception sources, in which all sensors are merged into a more reliable measured data set and reducing the computational complexity. In sequence, the inspection environment is predicted including its fundamental sphere characteristics such as radius and center.

The environment features can be predicted, after the fundamental identification of environment, where it is estimated the specific components (as weld beads, plates and corners) in accord with international standards of storage tank design. The next step is related to estimation of occupancy grid, which wraps the storage tank with a group of voxels, where each voxel is designed with a confidence degree based in observed voxels in its neighborhood. At last, the climbing robot has a deep knowledge of inspection environment and can to make a reliable inspection planning.

Future works will discuss the reliability updating of occupancy grid during the robotic inspection, application of evolutionary methods to the inspection planning and a more in-depth discussion of application in real pressure vessels.

Table 3 Comparison between the traditional mapping method and the proposed method

	Classic method	Proposed method
distance traveled	52.006m	3.930m
total time	25min	21.42min
observed area	15.77%	3.82%
predicted area	0%	96.18%
knowledge of the environment	15.77%	100%
observed aspects	only distance	distance and tank features

Acknowledgements The authors thanks to National Counsel of Technological and Scientific Development of Brazil (CNPq), Coordination for the Improvement of Higher Level People (CAPES), National Agency of Petroleum, Natural Gas and Biofuels (ANP) together with the Financier of Studies and Projects (FINEP) and Brazilian Ministry of Science, Technology and Innovation (MCTI) through the ANP Human Resources Program for the Petroleum and Gas Sector - PRH-ANP/MCTI PRH10-UTFPR for financial support of this work.

References

1. Ren, X., Fox, D., Konolige, K.: Change their perception: Rgb-d for 3-d modeling and recognition. IEEE Robot. Autom. Mag. **20**(4), 49–59 (2013)
2. Browatzki, B., Tikhanoff, V., Metta, G., Bülthoff, H.H., Wallraven, C.: Active object recognition on a humanoid robot. In: 2012 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 2021–2028 (2012)
3. Singh, A., Sha, J., Narayan, K.S., Achim, T., Abbeel, P.: Bigbird: A large-scale 3d database of object instances. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 509–516 (2014)
4. Endres, F., Hess, J., Sturm, J., Cremers, D., Burgard, W.: 3-d mapping with an rgb-d camera. IEEE Trans. Robot. **30**(1), 177–187 (2014)
5. Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W.: Octomap: An efficient probabilistic 3d mapping framework based on octrees. Auton. Robot. **34**(3), 189–206 (2013)
6. Lee, Y.-J., Song, J.-B., Choi, J.-H.: Performance improvement of iterative closest point-based outdoor slam by rotation invariant descriptors of salient regions. J. Intell. Robot. Syst. **71**(3-4), 349–360 (2013)
7. Prankl, J., Aldoma, A., Svejda, A., Vincze, M.: Rgb-d object modelling for object recognition and tracking. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 96–103 (2015)
8. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., et al.: Kinectfusion: real-time 3d reconstruction and interaction using a

moving depth camera. In: Proceedings of the 24th annual ACM symposium on User interface software and technology. ACM, pp. 559–568 (2011)
9. Chang, H.J., Lee, C.G., Lu, Y.-H., Hu, Y.C.: P-slam: Simultaneous localization and mapping with environmental-structure prediction. IEEE Trans. Robot. **23**(2), 281–293 (2007)
10. Chung, S.Y., Huang, H.P.: Simultaneous topological map prediction and moving object trajectory prediction in unknown environments. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 1594–1599 (2008)
11. Wermelinger, M., Fankhauser, P., Diethelm, R., Krüsi, P., Siegwart, R., Hutter, M.: Navigation planning for legged robots in challenging terrain. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 1184–1189 (2016)
12. Sermanet, P., Hadsell, R., Scoffier, M., Muller, U., LeCun, Y.: Mapping and planning under uncertainty in mobile robots with long-range perception. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS. IEEE, pp. 2525–2530 (2008)
13. Santos, H.B., Teixeira, M.A.S., Oliveira, A.S., Arruda, L.V.R., Neves-Jr, F.: Scheduled fuzzy controllers for omnidirectional motion of an autonomous inspection robot with four fully steerable magnetic wheels. In: XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium, pp. 263–268 (2016)
14. Espinoza, R.V., de Oliveira, A.S., de Arruda, L.V.R., Neves-Jr, F.: Navigation's stabilization system of a magnetic adherence-based climbing robot. J. Intell. Robot. Syst. **78**(1), 65–81 (2015)
15. Espinoza, R.V., de Oliveira, A.S., de Arruda, L.V.R., Junior, N.F.: Adhesion loss prediction of a climbing robot through magnetic field analysis by artificial neural networks. In: 22nd International Congress of Mechanical Engineering, pp. 3–7 (2013)
16. de Oliveira, A., de Arruda, L., Neves-Jr, F., Espinoza, R., Nadas, J.: Adhesion force control and active gravitational compensation for autonomous inspection in lpg storage spheres. In: Robotics Symposium and Latin American Robotics Symposium (SBR-LARS), 2012 Brazilian, pp. 232–238 (2012)
17. Santos, H.B., Teixeira, M.A.S., de Oliveira, A.S., de Arruda, L.V.R., Neves-Jr, F.: Quasi-omnidirectional fuzzy control of a climbing robot for inspection tasks, Journal of Intelligent & Robotic Systems (2017)
18. Teixeira, M.A.S., Santos, H.B., Oliveira, A.S., Arruda, L.V.R., Neves-Jr, F.: Environment identification and path planning for autonomous ndt inspection of spherical storage tanks. In: XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium, pp. 193–198 (2016)
19. Pahlberg, T., Hagman, O., Thurley, M.: Recognition of boards using wood fingerprints based on a fusion of feature detection methods. Comput. Electron. Agric. **111**, 164–173 (2015)
20. Senthilkumaran, N., Rajesh, R.: Edge detection techniques for image segmentation–a survey of soft computing approaches. Int. J. Recent Trends Eng. **1**(2), 250–254 (2009)
21. Dougherty, E.R., Lotufo, R.A.: T. I. S. for Optical Engineering SPIE, Hands-on morphological image processing. SPIE Optical Engineering Press Washington, vol. 71 (2003)
22. Muralikrishnan, B., Raja, J.: Computational surface and roundness metrology. Springer Science &Business Media, Berlin (2008)
23. Moss, D.R., Basic, M.M.: Pressure vessel design manual. Butterworth-Heinemann, Oxford (2012)
24. N-268: Fabrication of Pressure Vessels, PETROBRAS Technical Standards, Std (2012)
25. N-1281: Design, Fabrication, and Assembly of Spheres, PETROBRAS Technical Standards, Std (2014)
26. N-1520: Storage Spheres - Data Sheet, PETROBRAS Technical Standards, Std (2010)
27. N-2111: Safety in Cleanup, Inspection and Repair of Storage Tanks and Pressure Vessels, PETROBRAS Technical Standards, Std (2011)
28. Passino, K.M., Yurkovich, S., Reinfrank, M.: Fuzzy control, vol. 20. Addison-wesley Reading, Boston (1998)
29. de Jesús Rubio, J., Bouchachia, A.: Msafis: an evolving fuzzy inference system. Soft Comput. **21**(9), 2357–2366 (2017)
30. Haykin, S., Network, N.: A comprehensive foundation. Neural Netw. **2**(2004), 41 (2004)
31. Rekabdar, B., Nicolescu, M., Kelley, R., Nicolescu, M.: An unsupervised approach to learning and early detection of spatio-temporal patterns using spiking neural networks. J. Intell. Robot. Syst. **80**, 83 (2015)
32. Zurada, J.M.: Introduction to artificial neural systems. West St. Paul, vol. 8 (1992)
33. de Jess Rubio, J.: Stable kalman filter and neural network for the chaotic systems identification. J. Frankl. Inst. **354**(16), 7444–7462 (2017). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0016003217304325
34. de Jess Rubio, J.: Discrete time control based in neural networks for pendulums, Applied Soft Computing. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494617302387 (2017)
35. Rohmer, E., Singh, S.P., Freese, M.: V-rep: A versatile and scalable robot simulation framework. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 1321–1326 (2013)
36. Teixeira, M.A.S., Dalmedico, N., Barbosa, H.S., de Oliveira, A.S., de Arruda, L.V.R., Neves, F.: Enhancing robot capabilities of environmental perception through embedded gpu. In: 2017 Brazilian Symposium on Computing System Engineering (SBESC). IEEE (2017)

Marco Antonio Simoes Teixeira received the B.Sc degree in Information Systems from Santa Catarina State University, Brazil, in 2014, the M.Sc. degree in Electrical and Computer Engineering from Federal University of Technology - Paraná, Brazil, in 2017. He is currently Ph.D student in Electrical and Computer Engineering at Federal University of Technology - Paraná (UTFPR). His research interests comprise Mobile Robots, in the areas of perception and intelligent systems.

Higor Barbosa Santos received the B.Sc. degree in Industrial Electrical Engineering (2014) and M.Sc. degree in Automation and System Engineering (2016) from Federal University of Technology - Paraná (UTFPR). He is currently Ph.D student in Automation and System Engineering at Federal University of Technology - Paraná (UTFPR). His research field is Robotics with particular focus in control systems for navigation of mobile robots, embedded systems, fuzzy control, intelligent systems, Robot Operating System (ROS).

Nicolas Dalmedico received the B.Sc. degree in Electronic Engineering (2016) from Federal University of Technology - Paraná (UTFPR). He is currently M.Sc. student in Automation and System Engineering at Federal University of Technology - Paraná (UTFPR). His research field is Mobile Robots focusing on perception, embedded systems, general-purpose computing on GPU and navigation.

Lucia Valeria Ramos de Arruda received the Electrical Engineer degree from Federal University of Ceara, Brazil, in 1985, the M.Sc. degree from the Graduate School of Electrical Engineering, Campinas State University (FEE/UNICAMP), Brazil, in 1988, and the Ph.D. degree in Electrical Engineering from the University of Nice-Sophia Antipolis, France, in 1992. Since 1995, she joined Federal University of Technology - Paraná, and actually, she is Full Professor. Her research interests comprise soft computing methods to model and control of dynamic systems.

Flavio Neves-Jr received the B.Sc and M.Sc Electrical Engineer degrees from Federal University of Technology - Paraná, Brazil, in 1987 and 1989, respectively and the Ph.D. degree in Electrical Engineering from the University Paul Sabatier (LAAS), France, in 1998. Since 1992, he joined the Federal University of Technology - Paraná, and actually, he is Full Professor. His research interests comprise hardware and software to instrumentation and automation.

Andre Schneider de Oliveira is Adjunct Professor at Federal University of Technology - Paraná (UTFPR). He is member of Advanced Laboratory of Robotics and Embedded Systems (LASER) and Automation and Advanced Control System Laboratory (LASCA). He received his Ph.D. degree in engineering of automation and systems (2011) with thesis focused on differential kinematics through dual-quaternions for vehicle-manipulator systems. In 2007, he received his M.Sc. degree in mechanical engineering, focused in force control of rigid manipulators, at Federal University of Santa Catarina (UFSC). His research interests lie in the areas of Robotics, Mechatronics and Automation with special focus in navigation and localization of mobile robots; autonomous and intelligent systems; perception and environment identification; cognition, and deliberative decisions; human-interaction and navigation control.

4 A QUADRAL-FUZZY CONTROL APPROACH TO FLIGHT FORMATION BY A FLEET OF UNMANNED AERIAL VEHICLES

This chapter presents the paper published in the *IEEE Access* journal (ISSN: 2169-3536). The data of the paper is shown in Table 2.

Table 2 – Data of the paper *A Quadral-Fuzzy Control Approach to Flight Formation by a Fleet of Unmanned Aerial Vehicle.*

Authors	TEIXEIRA, Marco Antonio Simoes; NEVES-, FLAVIO ; KOUBAA, ANIS; DE ARRUDA, LUCIA VALERIA RAMOS ; DE OLIVEIRA, ANDRE SCHNEIDER
Title	*A Quadral-Fuzzy Control Approach to FlightFormation by a Fleet of UnmannedAerial Vehicles*
Journal	*Sensors*
ISSN	2169-3536
DOI	https://doi.org/10.1109/ACCESS.2020.2985032
Publication date	April 2, 2020

Source: Own authorship.

This work is motivated by tasks that need to be performed by multiple agents (drones). Some tasks, such as transporting heavy loads, monitoring, exploring environments, sensing, among others, in many cases need to be carried out separately by different agents. When the task is performed by a single agent/robot, such as cargo transportation, the action is limited by the robot's carrying capacity. When executed by more agents, the maximum weight of the product transported can be increased, increasing the number of agents involved in the task.

This work aims to carry out a navigation approach in formation for multiple drones. To perform collaborative tasks, the first step to be taken is an approach to navigate the environment without colliding, maintaining a pre-defined formation. This paper is concerned with the approach of navigation and high preservation of agents. This work does not focus on odometry errors, considered non-existent here.

This paper fulfills the second specific objective of the thesis by proposing the use of 3D perception sensors in aerial robots (drones). In addition to using 3D sensors for navigation and obstacle avoidance actions, an approach was proposed where only one agent can perceive the environment and share the information already processed with the other agents. Therefore, each agent does not need to have a sensor attached to it.

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 19, 2020, accepted March 22, 2020, date of publication April 2, 2020, date of current version April 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2985032

A Quadral-Fuzzy Control Approach to Flight Formation by a Fleet of Unmanned Aerial Vehicles

MARCO ANTONIO SIMOES TEIXEIRA[1], FLAVIO NEVES-JR[1], ANIS KOUBAA[2,3], (Member, IEEE), LUCIA VALERIA RAMOS DE ARRUDA[1], (Member, IEEE), AND ANDRE SCHNEIDER DE OLIVEIRA[1], (Member, IEEE)

[1]Graduate School of Electrical Engineering and Computer Science (CPGEI), Universidade Tecnológica Federal do Paraná (UTFPR), Curitiba 80230-901, Brazil
[2]Prince Sultan University, Robotics and Internet-of-Things Lab (RIOTU), Riyadh 12435 3276, Saudi Arabia
[3]CISTER, INESC-TEC, ISEP, Polytechnic Institute of Porto, 4200-465 Porto, Portuga

Corresponding author: Marco Antonio Simoes Teixeira (marcoteixeira@alunos.utfpr.edu.br)

This work was supported in part by the National Counsel of Technological and Scientific Development of Brazil (CNPq), in part by the Coordination for the Improvement of Higher Level People (CAPES), in part by the Brazilian Ministry of Science, Technology, Innovation and Communication (MCTIC), and in part by the Robotics and Internet-of-Things Lab in Prince Sultan University.

ABSTRACT This paper addresses the control of a fleet of unmanned aerial systems (UAVs), termed as drones, for flight formation problems. Getting drones to fly in formation is a relevant problem to be solved when cooperative cargo transportation is desired. A general approach for this problem considers the coordination of a fleet of UAVs, by fusing all information coming from several individual sensors posed on each UAVs. However, this approach induces a high cost as every UAV should have its advanced perception system. As an alternative, this paper proposes the use of a single perception system by a fleet composed of several elementary drones (workers) with primitive low-cost sensors and a leader drone carrying a 3D perception source. We propose a Quadral-Fuzzy approach to ensure that all drones fly in formation and will not collide with each other or with environment obstacles. We also develop a new way to compute potential fields based on possibility fuzzy (fuzziness) measure with the focus of avoiding collisions between the drones. The proposed approach encompasses four high-coupled intelligent controllers that respectively control the leader and worker drones' motion and implement obstacle and collision avoidance procedures. Simulation results using a fleet of four aerial drones are presented, showing the potential for solving usual problems to flights in formation, such as dodging obstacles, avoiding collisions between the drones, among others.

INDEX TERMS Unmanned aerial vehicles (UAVs), multi-agent systems, distance-based formation, flight-formation control, autonomous flight.

I. INTRODUCTION

Nowadays, unmanned aerial vehicles (UAVs) are used in several applications from military and civilian domains such as forest fire monitoring, surveillance, terrain mapping, and surveying, tracking, disaster management, blood or medical equipment delivery, and others [1]–[5]. Aerial drones are fast, flexible, lightweight, low-cost, and easy to use UAV with the potential to reduce the cost and time in the logistic field. An extensive survey of aerial drones for civilian applications is given in [6]. From this survey, one of the most promising applications of aerial drones is for autonomous cargo transport and delivery by e-commerce retailers and also for express delivery of perishable goods such as food or medicines. However, several challenges must be solved for the service to be effective [6]–[8]: (1) Limited payload: in general, goods must not weigh more than 2 kg; (2) Integration of low-cost sensors and positioning system, that is, several sensors like gyroscope, accelerometer, among others, can be used to create the odometry. The sensor fusion with accurate high location sensors, such as Real Time Kinematic GPS [9], allows to obtain the drone position in a global reference system; (3) Avoid obstacles and collisions: it is necessary to establish a flyable collision-free path in a

The associate editor coordinating the review of this manuscript and approving it for publication was Yongping Pan.

dynamic environment; (4) Communication and connectivity: communication links with the ground control station are need to receive instructions; (5) Landing at specified locations or the use of a parachute to delivery goods, (6) Limited flight range due to energy requirement (battery duration): the traveled distance depends on power transfer efficiency for motor, cruising velocity and power consumption of electronics, (7) Other concerning including government regulation and public acceptation.

This paper addressed challenges 1 to 3 above mentioned. The problem of the limited payload can be circumvented by the use of multiples drones that cooperatively transport the product. However, the use of a fleet of drones introduces new problems such as flight formation, drones communication issues, the need for a distributed system for collision, and obstacles avoidance. In this paper, collision avoidance refers to a drone trying to avoid another drone, while obstacle avoidance means that the drones try to avoid obstacles from the environment.

This work aims to present a novel intelligent and cooperative strategy of load transportation using multiple drones through a *quadral-fuzzy* approach. The added value of this work is the use of a predefined formation for navigation, for a fleet of drones composed of a leader drone equipped with precise 3D perception system and worker drones equipped with low-cost positioning systems [10], [11]. The multiple drones system can autonomously deflect obstacles, thus avoiding collisions and possible damage to load and agents.

Concerning challenge 2, the proposed approach has as the prerogative the use of a unique perception system, with resolution and amplitude to supervise the whole group and environment. The fleet leader computes the goal locations of worker drones based on its perception of the situation at every instant of time, based on the distance to the nearest obstacle. Workers know their locations in the environment using sensor fusion (e.g., accelerometer, gyroscope, and GPS) to estimate the position of the drone in the environment. The goal is to use a single 3D perception sensor in the leader to reduce the cost of the system and avoid overlapping information from multiple 3D sensors.

Based on such sensor data, the fleet leader can compute the flyable collision-free path to the entire formation (challenge 3). Moreover, each worker drone is also equipped with a collision-avoidance system based on potential field [12].

Our main contribution lies in the design and development of a highly-coupled *quadral-fuzzy* approach to managing the multiple-agent motion applied to cooperative transport. A fuzzy controller is developed to control the leader motion, considering the current position of the drone and its desired position, as well as the environment perception information to deviate from obstacles. Worker drones act with a similar fuzzy controller, but without the obstacle deviation skill, i.e., their position is defined by the leader. Another fuzzy system performs obstacle deviation for the whole formation and ensures the optimal configuration for cargo transportation. A fuzzy self-preservation strategy is adopted to prevent collisions, assuring a safe flight to drones and cargo.

This work is divided into five sections. Section 2 brings some related works. Section 3 describes in detail the proposed *quadral-fuzzy* approach. Section 4 discusses the approach evaluation based on some experiments results. At last, section 5 presents the work conclusions.

II. RELATED WORKS

Several studies have contributed to making cargo delivery by drone a reality [6]. The researches are looking for improving navigation capabilities such sensing ability [13], [14], intelligent control [15]–[19] and obstacle and collision avoidance [12], [20], [21].

The integration of visual sensing techniques in drone applications is a trend for researches on position-attitude control, pose estimation and mapping, obstacle detection as well as target tracking [13], [22]. Following this trending, we use a 3D perception source providing a cloud of points (or PointClouds), which can collect spatial information from the environment that is combined with information from low-cost sensors (multi-fusion sensor), allowing run procedures for collision and obstacle avoidance and also for path planning.

Nowadays, fuzzy systems have been successfully used for navigation, guidance, and control of autonomous vehicles and mobile robots [23]–[25]. This extensive use is explained by the simple control structure and also the natural and practical design of fuzzy systems [26]. A survey of nonlinear and adaptive intelligent control techniques for a quadcopter drone, as the drones used in this paper, is given in [15] in which the use of fuzzy control is highlighted. For example, the work in [27] develops an autonomous drone, able to follow planned trajectories by using a robust fuzzy controller based on a precise dynamic and kinematic models of the drone. Different from [27] but similar to some works cited in [15], the proposed *quadral-fuzzy* approach developed herein does not require any model and can adapt to unforeseen situations, providing excellent coverage of wide-ranging operating conditions.

Formation control is an essential issue for the development of collective and collaborative behaviors in multi-agents systems. Potential field and leader-follower are the two main approaches for formation control [12]. Hybrid approaches, combining both theories, are often used to build and move formations because they are effective, robust, and easy to handle [28]–[31]. In this paper, besides the use of fuzzy theory to develop intelligent controllers for drones motion and obstacle avoidance as usual in literature [15], [20], we combine potential field and leader-follower approaches to develop a fuzzy system to avoid the collisions in formation. This fuzzy self-preservation system is based on a fuzzy possibility map in which the potential field reflects the fuzziness of each direction vector in the field. This fuzzy potential field computation is a contribution of this work. Finally, all fuzzy controllers in this paper are modeled as recommended

by [32], using centroid defuzzification, conjunction in minimum, disjunction in maximum, activation in minimum and accumulation in maximum.

It is worthwhile to note that the main contribution of this work is an original solution (*quadral-fuzzy* approach) for the multiple drones flight formation. Cargo transportation has been only cited as a motivating example for flight formation problems.

III. THE PROPOSED *quadral-fuzzy* APPROACH

In this section, we propose a multi-drone formation flight strategy for cooperative cargo transportation. We consider two main safety requirements in the design of the system:

- all drones must maintain formation (position and orientation);
- all drones must avoid collision among them and with the environment's obstacles to assure their integrity.

For this purpose, the proposed approach adopts a leader-workers configuration developed through four subsystems: leader control, worker control, self-preservation, and formation maintenance. In this configuration, the leader drone can perceive the environment and presents an active/reactive behavior while workers exhibit only reactive behavior. Each subsystem is an independent fuzzy system that models the corresponding individual (leader/workers) or collective (team) behavior. All subsystems run in a highly-coupled way composing the new *quadral-fuzzy* approach for cooperative cargo transportation, as shown in Figure 1.

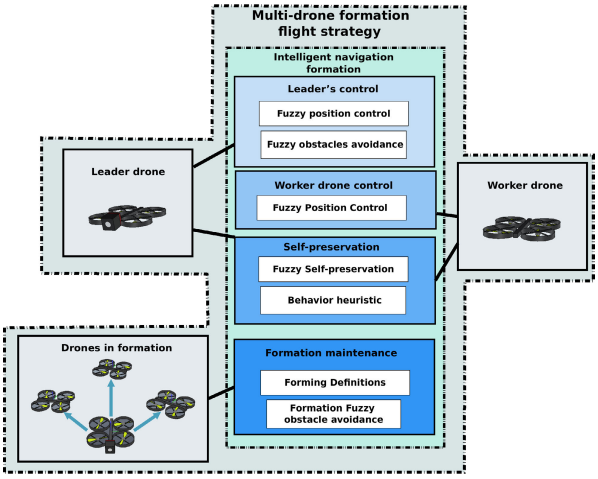


FIGURE 1. schematic of the complete proposed system for multi-drone formation flight (*quadral-Fuzzy* approach).

In this Figure, the first block is related to the *leader control* subsystem that generates the position and orientation control for the leader drone, including obstacle avoidance skill. The obstacle deviation maneuvers should not affect the other drones in the formation; thus, only linear deviations are allowed. The second block refers to the *worker control* subsystem that controls the worker drones' position, assuring a global formation stabilization. Each worker is endowed with the *worker control* subsystem. The third fuzzy system concerns the *self-preservation* ability, where a knowledge-based method is developed to avoid collisions inside formation that can damage the drones. All fleet members are endowed with this security system. Finally, the block called *formation* is responsible for establishing the flying formation rules assuring cargo transportation safety. This module computes the positions of workers around the leader and develops an intelligent strategy for detecting and avoiding external obstacles. This subsystem manages the entire fleet.

In the proposed *quadral-fuzzy* approach, all fuzzy subsystems correspond to fuzzy non-linear functions $g(x)$ that map fuzzy input variables ($x \in U_x \subset \Re^n$) to fuzzy output variables ($y = g(x) \in U_y \subset \Re$). The universe of discourse is defined as $U = [0, lim]$ or $U = [-lim, lim]$, according to the mapped variables. The membership functions describing the fuzzy sets $W_x = \{(x, \mu_A(x)|x \in U_x\}$ and $W_y = \{(y, \mu_A(x)|x \in U_x\}$ are pseudo-trapezoid ones defined in $\Re$, and given by Eq. (1), where $[a, d] \in \Re$; $a \leq b \leq c \leq d$ and $a < d$ ([32]).

$$\mu_A(x; a, b, c, d) = \begin{cases} \dfrac{x-a}{b-a}, & if\ x \in [a, b) \\ 1, & if\ x \in [b, c] \\ \dfrac{x-d}{c-d}, & if\ x \in (c, d] \\ 0, & if\ x \in \Re - (a, d) \end{cases} \quad (1)$$

All fuzzy controllers are based on product inference engine with center average defuzzifier.

The aerial vehicles modeled in this paper is a quadcopter drone with six degrees of freedom corresponding to the linear and angular velocities about X, Y, and Z axis in a 3D environment. Such degrees of freedom are labeled as V_x, V_y and V_z for linear motion and W_x, W_y and W_z for angular movement (roll, pitch, and yaw) as shown in Figure 2.**A**.

The position and orientation of the drone are determined by inertial sensors (i.e., IMU, Gyroscope, GPS). When using multiples drones, all positions must be given in the same coordinate system. Figure 2.*C* shows an example in which the positions of four drones are converted to the same reference frame. Thus, linear and angular transformations based on rotation and translation matrices are used to convert data captured by a sensor to any reference point by the use of transformation trees [33]–[35]. As a result, data from multiples sensors placed on several drones can be translated and processed to the same coordinate system.

In this paper, the coordinates of each drone are transformed into a coordinate in the global frame, that is used as a reference frame for all the drones positioning. Furthermore, homogeneous transformation matrices translate the sensor data to a unique reference point, such as the center of the drone, for example, supporting data processing at the same coordinate system, independent of drone and type of sensor used to data capture. All systems composing the *quadral-fuzzy* approach in Figure 1 are detailed in the next sections.

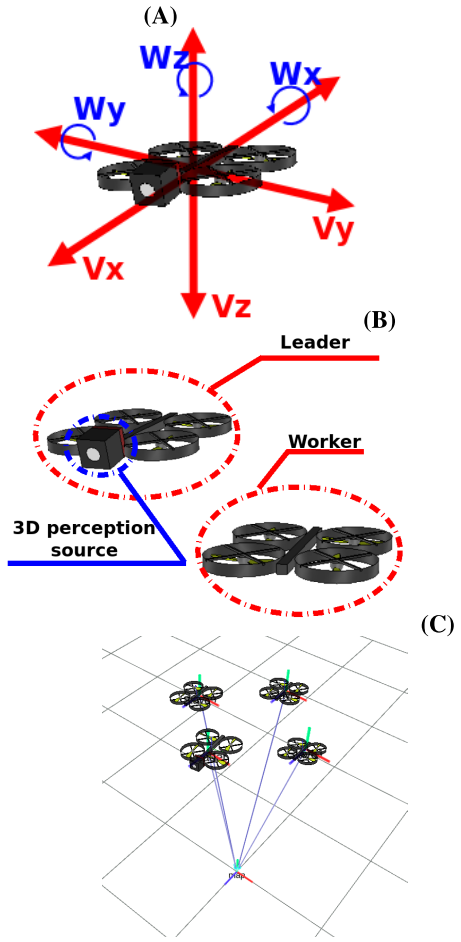


FIGURE 2. Overview of used aerial vehicles (drones). (A) Degrees of freedom of an aerial vehicle (drone). (B) Differences between leader and work drones. (C) Representation of all drones in the same coordinate system.

A. LEADER's CONTROL SYSTEM

A leader drone composes the fleet considered in this paper. This drone can perceive the environment and other drones and also it can estimate their displacement. The leader drone is endowed with a source of 3D perception, and the other drones are considered worker drones, as illustrated in Figure 2.B.

The 3D perception source embedded in the leader drone delivers a cloud of points at a resolution of 176 by 144 points with a viewing angle of 69° (h) x 56°(v). This source can operate up to 10 meters away, and the simulated sensor is based on the Mesa SR4000 3D ToF sensor [36]. This perception source allows detecting the environment's obstacles by the leader drone. All drones (leader and workers) are identical, but only the leader has a source of perception.

The *leader control system* aims to ensure a safe flight for the leader drone. It consists of two fuzzy subsystems, as shown in Figure 3. The *Fuzzy position control* is the motion controller driving the drone to reach the desired position while the *Fuzzy avoid obstacles* block determines leader

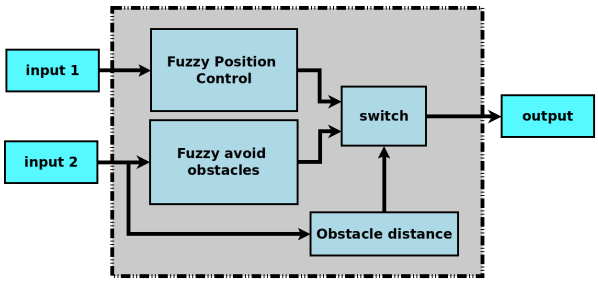


FIGURE 3. Representation of the leader drone position control system. The two Fuzzy controls (Fuzzy position control and Fuzzy avoid obstacles) compete for drone leader speed control, where selection takes place by the distance of identified obstacles.

maneuvers to avoid obstacles while trying to achieve the desired point. Moreover, the block called *switch* turns off the motion controller in the presence of imminent collision, leaving the leader free to carry out obstacle detour, under *Fuzzy avoid obstacles* control.

1) FUZZY POSITION CONTROL SUBSYSTEM

The fuzzy subsystem for position control of the leader drone implements three fuzzy mappings to calculate leader velocities in direction to the desired point that is defined as the goal for the leader, considering the current position of the drone, R_p, and the desired position, D_p. Given in the 3D reference coordinate frame, three measurement distances among these positions are used as input (*input*1 in Figure 3) for this fuzzy subsystem: (1) the Euclidean distance in meters (δ_e) between the position of the drone and the desired point, (2) the angular distance over z-axis that corresponds to the angular orientation error in degrees (δ_a) and (3) linear distance over z-axis computing the linear position error in meters (δ_z). These measured distances are computed by equations 2 and 3.

$$\delta_e = \sqrt{(R_p(x)-D_p(x))^2+(R_p(y)-D_p(y))^2+(R_p(z)-D_p(z))^2} \tag{2}$$

$$\delta_z = (D_p(z) - R_p(z)) \tag{3}$$

The δ_a calculation considers the drone's current position as well as its angular orientation. This input provides the necessary angular rotation so that the drone has pointed to the desired angle. The computational procedure used to obtain δ_a is given by Algorithm 1.

The fuzzy sets $A_i(x)$ are defined for each input variables ($x = [\delta_e, \delta_a, \delta_z]^T$) over their universe of discourse ($\delta_e \in [0, 20]$, $\delta_a \in [-180, 180]$ and $\delta_z \in [-20, 20]$). These fuzzy sets and their correspondent membership functions $\mu_{A_i}(x; a, b, c, d)$ are given in Table 1. Based on these fuzzy sets, the fuzzy values of the euclidean distance δ_e are combined with the other two input variables (δ_a, δ_z) to fire three rule bases generating the output of position controller.

The outputs of the leader's control subsystem are the speeds that will be directly applied to the leader. Although the leader drone is omnidirectional, this control subsystem only

TABLE 1. Input variables of position control: fuzzy sets $A_i(x)$ and membership functions $\mu_{A_i}(x; a, b, c, d)$.

	Euclidean distance (m/s)				Angular distance (°)					Linear z distance (m/s)				
	A_1	A_2	A_3	A_4	A_1	A_2	A_3	A_4	A_5	A_1	A_2	A_3	A_4	A_5
a	0	1	2	3	-180	90	-50	0	50	-20	-4	-2	0	2
b	0	2	3	5	-180	50	0	50	100	-20	-2	0	2	5
c	1	2	3	20	-100	50	0	50	180	-5	-2	0	2	20
d	2	3	5	20	-50	0	50	90	180	-2	0	2	4	20

Algorithm 1 Compute δ_a

```
angleDifference ← antan2(D_p(V_y) − R_p(V_y), D_p(V_x) − R_p(V_x));
if angleDifference < 0 then
    angleDifference ← π + (π − abs(angleDifference));
δ_a ← angleDifference − R_p(W_z);
if abs(δ_a) > π then
    δ_a = 2 * π − abs(δ_a);
    if R_p(W_z) < angleDifference then
        delta_a ← delta_a * −1;
return δ_a;
```

promotes linear motion over $x - z$ plan, preventing sideways sliding. Thus, the three outputs are the linear velocities over the axis x (v_x) and over the axis z (v_z), given in meters per second (m/s), and the angular velocity (ω_z) in radians per second (rad/s). Since decision block (*switch*) in Figure 3, only chooses if the leader's control system output (*output*) comes from *Fuzzy position control* or *Fuzzy avoid obstacles* subsystems, such velocities are also the output variables of position controller. The fuzzy sets $A_i(y)$ defined for each output variables and their correspondent membership functions $\mu_{A_i}(y; a, b, c, d)$ are given in Table 2.

The three non-linear maps (one for each output velocity), generated by the rule bases of fuzzy position control subsystem are shown in Figure 4. As discussed above, the Euclidean distance that corresponds to the 3D position error affects all velocities (angular and linear). It is worthwhile to note that the x-axis linear velocity (v_x) decreases when angular error ($delta_a$) grows-up (see the upper surface in Figure 4). This behavior indicates that the drone must first correct its orientation angle before proceeding to the desired point.

As soon as a goal position is set to the leader drone, the fuzzy position control subsystem continuously computes linear and angular speed that should be applied to the drone's motors according to the decision taken by *switch* block. This decision takes into account the proximity of obstacles, as will be explained in the next section.

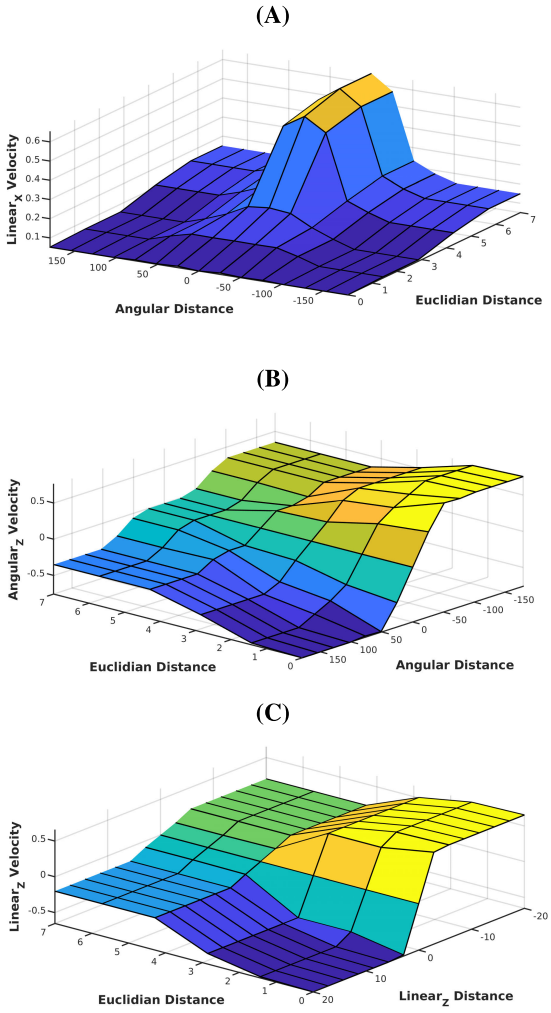


FIGURE 4. Lead drone Position Control - Velocity maps generated by fuzzy position control system. (A) Map of the linear velocity in the X-axis, about the linear error (Euclidean distance) and angular error (angular difference) between the drone position and the desired position. (B) Map of the angular velocity in the Z-axis, about the linear error (Euclidean distance) and angular error (angular difference) between the drone position and the desired position. (C) Map of the linear velocity in the Z-axis, about the linear error (Euclidean distance) and linear Z-axis error (height difference) between the drone position and the desired position.

2) FUZZY OBSTACLES AVOIDANCE SUBSYSTEM

This subsystem aims to prevent collisions with external obstacles. In this paper, any object that is not identified by the leader as a worker drone is considered as obstacles and the formation must deviate it. For this, the cloud of points obtained by 3D perception sensor is processed to detect objects in front of the leader drone, as proposed in [14]. Thus, the closest object is identified, and three distance measurements, in meters, among this object and the leader are

TABLE 2. Output velocities of position control: fuzzy sets $A_i(y)$ and membership functions $\mu_{A_i}(y; a, b, c, d)$.

	$v_x \in [0, 1]$				$\omega_z \in [-1, 1]$					$v_z \in [-1, 1]$				
	A_1	A_2	A_3	A_4	A_1	A_2	A_3	A_4	A_5	A_1	A_2	A_3	A_4	A_5
a	-0.15	0.05	0.15	0.25	-1.5	-0.625	-0.375	0	0.375	-1.5	-0.4	-0.2	0	0.2
b	-0.05	0.15	0.25	0.355	-1	-0.375	0	0.375	0.625	-1	-0.2	0	0.2	0.4
c	0.05	0.15	0.25	1	-0.75	-0.375	0	0.375	1	-0.35	-0.2	0	0.2	1
d	0.15	0.25	0.35	1.5	-0.375	0	0.375	0.625	1.5	-0.2	0	0.2	0.4	1.5

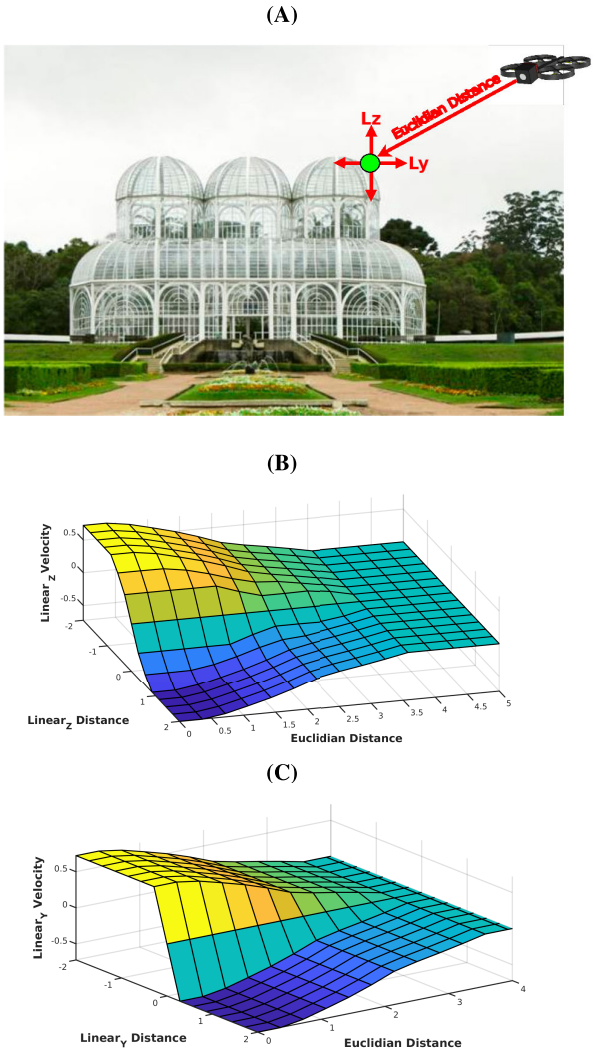


FIGURE 5. Lead drone avoid obstacles control system - Velocity maps generated by fuzzy position control system. (A) Obstacle detection and measured distances to the drone. (B) Map of the linear velocity in the Z-axis, about the linear error (Euclidean distance) and linear error in Z-axis (Z-axis difference) between the drone position and obstacle detected. (C) Map of the linear velocity in the Y-axis, about the linear error (Euclidean distance) and linear error in Y-axis (Y-axis difference) between the drone position and obstacle detected.

computed and used as input (*input*2 in Figure 3) to this subsystem: Euclidean distance (δ_e), the linear distance over the y-axis (δ_y) and the linear distance over the z-axis (δ_z). Figure 5.A illustrates these measured distances, where the green dot represents the obstacle closest to the drone.

Some linguistic predicates are used to define input fuzzy sets. Concerning *Euclidean Distance*(δ_e), the object can be *close* to, *far* or *so far* from the leader. The object position over z-axis (δ_z) can be *in front of*, *above* or *below* the drone. The object position over y-axis (δ_y) can be *in face*, *to the right* or *to the left* of the leader drone. Therefore, the fuzzy set for *Fuzzy obstacles avoidance* subsystem and their correspondent membership functions $\mu(x; a, b, c, d)$ are given in Table 3.

Similar to the position control subsystem, the outputs of this block are speeds in meters per second (m/s) that will be directly applied to the leader drone. The adopted strategy only implements obstacle detour maneuvers in the z - y plan. Thus the leader drone linearly moves along these two axes. Any angular actions are carried out to deflect obstacles; that is, the leader drone is ever oriented to the goal position.

The fuzzy sets $A_i(y)$ and their correspondent membership functions $\mu_{A_i}(y; a, b, c, d)$ for both output variables (*linear velocities* v_y and v_z) are the same and they are given in Table 3. The fuzzy non-linear function (derived from the rule bases) used to compute both velocities are given in Figure 5. Both functions are smooth surfaces, assuring that the leader drone maneuvers to obstacle avoidance do not cause bumps in the cargo.

As the position controller, the obstacle avoidance subsystem is always active; that is, it is always sending speed information to the leader drone. However, the decision about which velocities signals should act on the motors, whether they are the outputs of the position controller, or they come from the obstacle avoidance subsystem, is taken by the *switch* block in Figure 3. For this decision, a simple threshold test is carried out. If the Euclidean distance among the closest obstacles and the leader drone is less than a threshold, then the leader's control system output (*output* in Figure 3) comes from obstacle avoidance subsystem otherwise they are the velocities computed by fuzzy position control subsystem.

B. WORKER CONTROL SYSTEM

The worker drones have only position sensors. Thus, these drones only know their current position relative to their initial position. Moreover, the only worker drone goal is to maintain the formation, and for this, it has a position controller a little simpler than the one described above for the leader. A worker drone safety flight is assured by the *Self-preservation* system that will be described in next section III-C

The worker drone control system can be seen in Figure 6. Its inputs are the same measured distances (δ_e, δ_a, δ_z) of the leader's position controller. However, in this case, the desired

TABLE 3. Fuzzy variables of *Obstacles avoidance* block.

	Input variables									Output variables				
	$\delta_e \in [0,5]$			$\delta_y \in [-2,2]$			$\delta_z \in [-2,2]$			*Linear velocities*				
	Close	*Far*	*So far*	*Rigth*	*Face*	*Left*	*Below*	*Front*	*Above*	**A1**	**A2**	**A3**	**A4**	**A5**
a	-2.5	-0.2	0	-2.5	-0.8	0	-1.5	0.5	2	-1.5	-0.6	-0.3	0	0.3
b	-2	0	0.2	-2	0	0.8	-1	2	3.5	-1.3	-0.3	0	0.3	0.6
c	-0.2	0	2	-0.8	0	2	0.5	2	5.5	-0.6	-0.3	0	0.3	1.3
d	0	0.2	2.5	0	0.8	2.5	2	3.5	6	-0.3	0	0.3	0.6	1.5

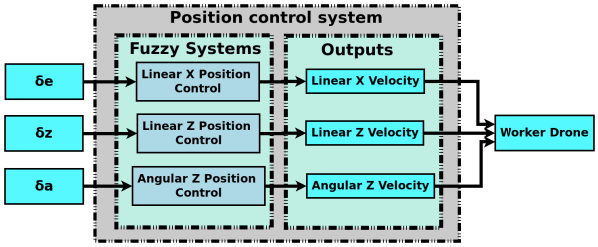


FIGURE 6. Representation of the workers drone position control system.

position corresponds to the position that the worker drone should occupy in the formation. The system outputs also are the same: the linear speeds (v_x and v_z), and the angular velocity (ω_z). However, the position control of the worker drone is organized into three independent fuzzy functions, one for each velocity to be computed. Unlike the leader drone who does not need agility but precision, the workers must take quick actions to keep their place in the formation. Thus, the use of 3 independent controllers provides agility enabling better results in worker drones control.

As the leader and workers are the same kinds of drones, the only difference among them is their embedded perception system, the fuzzy sets and memberships functions for worker drones input/output variables are the same given in Figure 4. However, the universe of discourse for such variables has been expanded ($v_z, \omega_z \in [-4,4]$), narrowed ($\delta_e \in [0,1]$ and $\delta_z \in [-5,5]$) or maintained ($\delta_a \in [-180,180]$ and ($v_x \in [0,1]$) to assure the agility requirements.

The rule base developed to drive the worker drone along x-axis contains rules modeling heuristic knowledge such as

- if the worker drone is very close ($\delta_e \in A1$) to desired position then the speed is very slow ($v_x \in A1$).
- if the worker drone is far ($\delta_e \in A3$) from desired position then speed is fast ($v_x \in A3$).

Similarly, the rules driving movements along z-axis are for example,

- if the vertical distance among the worker drone and the desired position is very down ($\delta_z \in A1$) then the speed is positive and very fast ($v_z \in A5$).
- if the worker drone is in the face of ($\delta_z \in A3$) the desired position, then the speed is near zero ($v_z \in A3$).

As a result of rules firing, if the output of the *Linear Z Position control* subsystem in Figure 6 is negative, the drone descends, and if the linear velocity δ_z is positive, the drone goes up. The *Linear Z Position control* goal is vertically to keep the drone as close as possible to its desired position in the formation.

Furthermore, the *Angular Z Position control* subsystem is always looking to keep the worker drone pointed to its desired position in the formation. For this, it computes drone rotation speed (ω_z) around the z-axis in radians per second (rad/s) according to rules such as

- if the angular error among the worker drone and the desired position is very high and positive ($\delta_z \in A5$) then rotation speed to the right is very fast ($\omega_z \in A1$).
- if the worker drone is aligned to ($\delta_z \in A3$) the desired position, then the rotation speed is near zero ($\omega_z \in A3$).

C. SELF-PRESERVATION SYSTEM

A self-preservation strategy is developed to prevent collisions between the drones during flight. This strategy adopts for each drone, a security cube centered at the drone. If any object (another worker drone) is detected over this cube surface, then the drone must perform a detour maneuver. This maneuver is based on the drone's expectation to move towards the obstacle position $P(x, y, z)$. Thus a possibility map is built describing all directions that a drone can take. A direction is represented by a vector linking the drone center to a position $P(x, y, z)$ over the cube's surface. In addition, a fuzzy system is used to infer the degree of possibility (*fuzziness*) associated to each direction in this map [32], considering drones angular and linear velocities, as well as the Euclidean distance among the drone in the center of the cube and the position $P(x, y, z)$ defining the direction. During flight, positions with angular errors into the interval $[-90°, 90°]$ are in front of the drone, on the contrary these points are behind the drone ($\delta_a \in [-180°, -90°] \cup [90°, 180°]$). In the same way, if the linear drone velocity is positive ($v_x > 0$), the drone is approaching the position; otherwise it is flying away from the position ($v_x < 0$).

The first step to build such maps is to generate a vector field around each drone to assign all possible directions (vectors) for drone displacement, as shown in Figure 7.A. Each vector *Pv[i]*, associated with a direction, is uniquely determined by its position $P(x, y, z)$ and *fuzziness*, corresponding to a degree of possibility of the drone moving in this direction. This degree of fuzziness is computed based on drone position and velocity information by a fuzzy system composed of two modules, as shown in Figure 7.B: linear motion and angular motion.

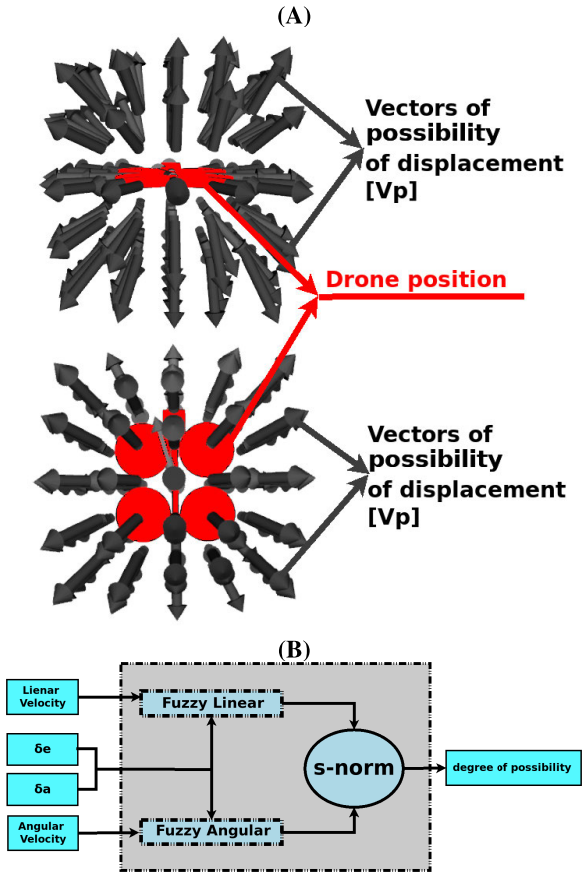


FIGURE 7. Self-preservation system overview. (A) Possibility vector [Vp] scattered around the drone's. (B) Self-preservation system strategy to apply degree of possibility to each possibility vector [Vp].

The first one assigns a fuzziness degree due to the drone's linear velocity uncertainty, weighting forward, or backward drone movements (upper vector field in Figure 7.A). The second fuzzy system considers the angular velocity uncertainty, and it weights turn left or right actions (lower vector field in Figure 7.A). The output of both systems is the degree of possibility of the drone moving in the direction of the vector *Pv[i]*. The s-norm algebraic sum is used to combine these two fuzzy variables giving the final possibility degree of each vector *Pv[i]* [32].

The input variables for both fuzzy systems are the Euclidean (δ_e from equation 2) and angular (δ_a from algorithm 1) distances among the drone and the desired position $P(x, y, z)$ of a *Pv[i]* vector. A third input variable for the fuzzy linear motion system is the linear velocity ($v_x \in [-1, 1]$) over x axis that indicates forward or backward motion. The angular velocity ($\omega_z \in [-1, 1]$) that indicates the left or right rotation is the third input for the fuzzy angular motion system. The input membership sets are the same given above for these variables: euclidean distance and linear velocity are given in Figure 5.D, angular distance, and angular velocity are given in Figure 4.

The output of both fuzzy systems reflects the possibility for the drone to carry out the linear and angular motion in each direction *Pv[i]*. This degree of possibility can be very low (*VL*), low (*L*), high (*H*) or very high (*VH*), in a range from 0 to 1.

For the sake of clarity, the fuzzy surface of both systems is displayed in two graphs combining inputs two by two. For fuzzy linear motion, the rule base has generated the two graphs in Figure 8. From these graphs, when the drone is moving forward, for example, and there are points in the face of it having a low angular difference, that is a small angular error, there is an excellent possibility to the drone reach these points. On the other hand, the fuzzy angular motion system is developed to predict the future position of the drones when performing an angular rotation. Thus *Pv[i]* with a high left angular error, for example, has a higher degree of possibility if the angular velocity is high on the left. The two surfaces in Figure 8 model such inference resulting from fuzzy angular motion rule base.

An example of the two vector fields resulting from both fuzzy linear and angular motion inferences is shown in Figure 9. In this Figure, a high degree of possible results in a light color vector, otherwise a dark vector indicates a direction with a low degree of possibility. The combination (by s-norm algebraic sum) of both fields in Figure 9 generates the final vector field, reflecting the possibilities of the drone moving in each direction.

Finally, a heuristic procedure is developed to prevent collisions based on the vector field computed by a fuzzy motion system. The cube volume around the drone is divided into four quadrants, from 0 to 90 degrees, from 90° to 180°, from −180° to −90° and from −90° to 0°. When another drone is detected in one of these areas, a diversion maneuver is carried out, causing the drone to slide to the opposite side, according to a repulsive force. The degree of probability of the drone moving in a particular direction is used to assign the force of the deviation to be executed in case of another drone has been detected in such direction. All directions *Pv's* that meet the surface of the detected (blue) drone, as shown in Figure 9.A, are identified, defining an interception area. The degree of possibility for all vectors in the interception area is averaged, generating the repulsive force. This deviation maneuver uses the linear velocities on the x and y-axis, weighted by the repulsive force, to move the drone into directions of 45, 135, −135, and −45 degrees from the area containing the detected drone.

When there are two drones around the drone performing the detour maneuver, it calculates the weights (repulsive force) based on the directions *Pv's* connecting it with both detected drones, and then the drones detour is carried out relative to the detected drone with the highest possibility of collision (highest repulsive force).

D. FORMATION MAINTENANCE SYSTEM

This section develops the technique of flying in fleet formation proposed by this work. Firstly, the leader drone is set

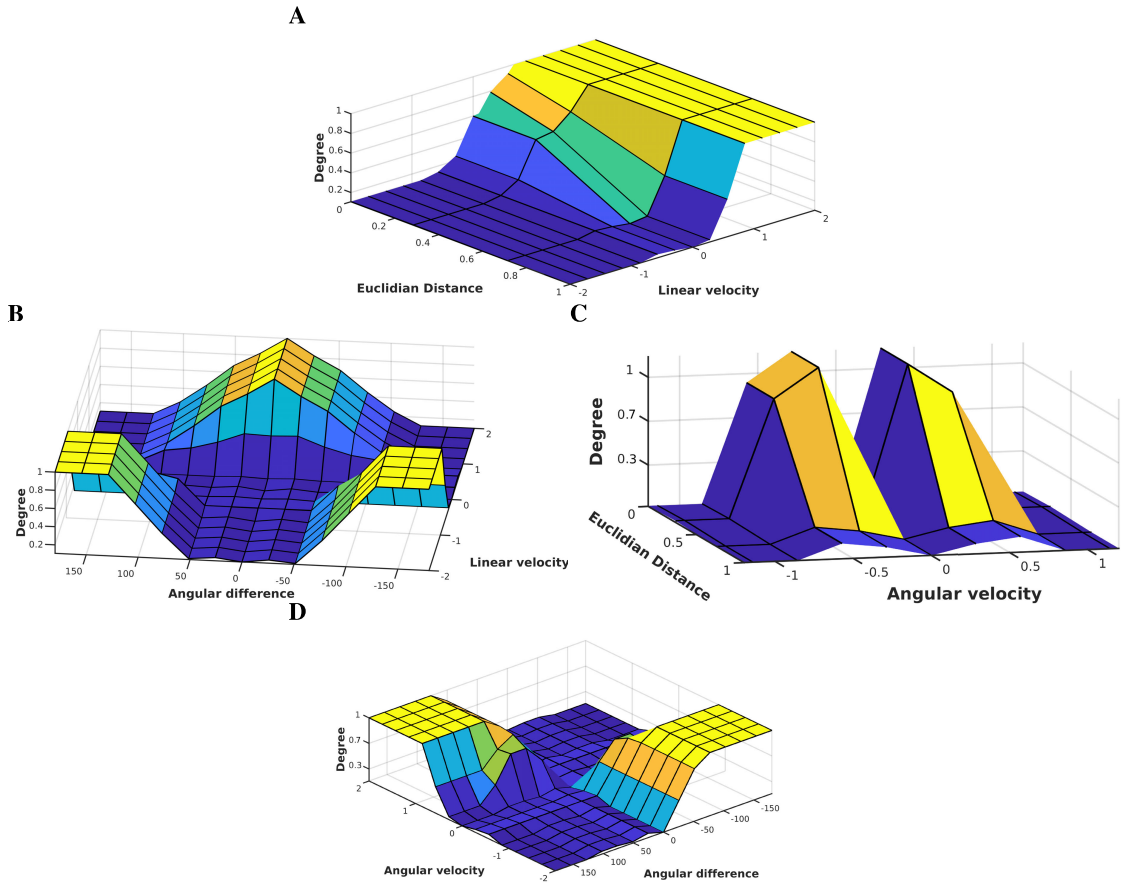


FIGURE 8. Self-preservation system - Degree of possibility maps generated by the Fuzzy linear system [(A), (B)] and Fuzzy angular system [(C), (D)]. (A) Map of degree of possibility, about the linear error (Euclidean distance) and linear velocity (in m/s) between the drones positions. (B)Map of degree of possibility, about the angular error and linear velocity (in m/s) between the drones positions. (C) Map of degree of possibility, about the linear error (Euclidean distance) and angular velocity between the drones positions. (D) Map of degree of possibility, about the angular error and angular velocity between the drones positions.

to reach point B from point A. During the flight, all worker drones must quickly reach specified positions, set from a leader's position, so that it is possible to assume and maintain the formation. The worker drone positions are computed by a formation function specifying the desired geometry shape of the formation. The formation function inputs are the leader position and information about obstacles that have been detected by the 3D perception sensor. There are two types of flight formation: cruiser formation that corresponds to an arbitrary geometrical shape defined according to cargo transportation requirements and safe formation (line mode) to be assumed in the presence of two or more external obstacles.

If the perception sensor identifies only one obstacle in front of the leader drone, the obstacle data is captured (δ_x, δ_z and δ_a) and inputted to a deviation obstacles fuzzy system, computing the necessary sliding maneuver to be carried out by the worker drones around the leader. Thus the formation function computes the new position of each drone in the fleet, assuring that all drones can deflect the obstacle while maintaining the formation. If two or more obstacles are detected in the face or both sides of the leader drone, and the euclidean distances among them are less than 1 meter each, the formation assumes the safe mode. In this safe formation, each drone lines up at a predefined order, forming a shell. As before, the formation function computes the new position of each drone in the fleet.

Both cruiser and safe formations take into account the position of the leader drone to determine the worker's position. During a flight, these relative positions are always the same, concerning the position and orientation of the leader. When the leader drone bypasses a nearby obstacle, for example, the deviation degree computed by the fuzzy system is added to the preset position of each drone, causing it to spin around the leader. Each worker drone tracks its desired position all the time except when it slides to deviation from another drone in the fleet, as explained in section III-C. This proposed strategy for formation maintenance is summarized in Figure 10.

The formation is a unique entity, where each drone is part and plays a role. Virtual markers inform the desired and

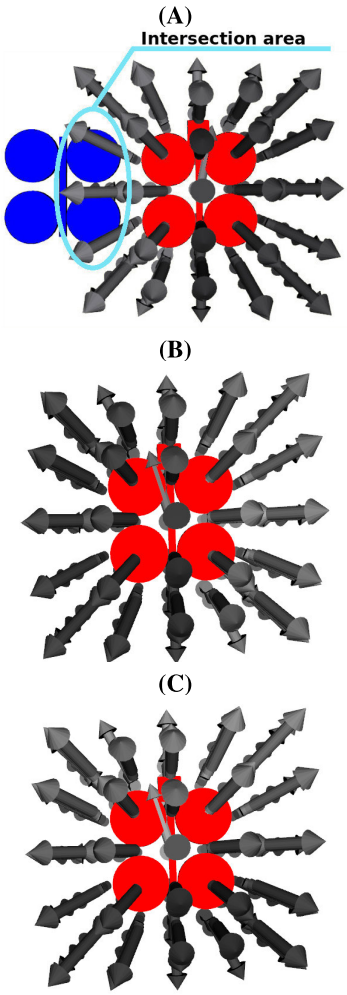


FIGURE 9. Example of self-preservation strategy in operation. (A) Representation of an intersection between two drones. (B) Vectors of possibility indicating that there is a possibility of drone collision in the right side. (C) Vectors of possibility indicating that there is a possibility of drone collision in the front.

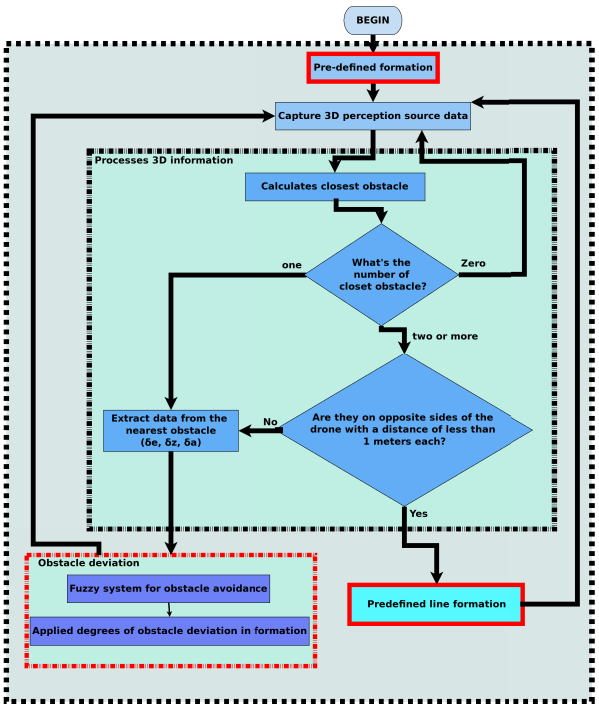


FIGURE 10. Diagram representing all strategy developed for formation maintenance system.The strategy can be divided into 4 main parts, namely *predefined formation, perception source processing, obstacle deviation* and *saved mode* (predefine line formation).

actual position of each drone, as well as define the formation limits. These markers are created from the actual position of each drone, obtained through the position sensors and transformation trees. The Figure 11.A shows such markers.

Considering a 3D coordinate [x, y, z] system, the current position of the leader is written as $L_p = [x, y, z]'$ and the actual position of the i-th worker drone is $W_p[i] = [x, y, z]'$ (i also refers to the initial position of the drone in formation). The desired position of the working drone to maintain formation is $Wd_p[i] = [x, y, z]'$.

1) CRUISER FORMATION

The first step to establishing the cruiser formation geometrical shape is to choose all drones' relative positions. For this, the initial leader's position $L_p = [x, y, z]'$ is set, and the positions for each worker drone in the formation are established. Each drone must be at a specific angle and a particular distance from the leader, to assure pose maintenance. A vector $V_d[i]$ indicating the displacement of each drone in each $[x, y, z]$ - axis relative to the leader's position L_p is created to save this geometrical shape. This vector can be added to the leader's current position during the flight, giving the desired new position of each worker drone all the time. However, a $V_d[i]$ has always the same orientation, independent of the leader drone has made an angular displacement. Thus, an angular correction term must be introduced, allowing to compute every time, the worker drone desired position

$$\begin{aligned} Wdp[i].x &= ((Vd[i].x * \cos(Lw.z)) \\ &\quad -(Vd[i].x * \sin(Lo.z))) + Lp.x \\ Wdp[i].y &= ((Vd[i].x * \cos(Lw.z)) \\ &\quad -(Vd[i].x * \sin(Lo.z))) + Lp.x \\ Wdp[i].z &= Lp.z, \end{aligned} \tag{4}$$

where we assumes for simplicity that $L_p.x$ corresponds to the values of x-coordinates of vector $L_p = [x, y, z]'$ and Lw represents the rotation of the drone leader (Figure 2.A). The same is valid for all vectors in the equation 4.

In a fleet with four drones, an initial position of the cruiser formation is represented in Figure 12.A. During the flight, the obstacle deviation maneuvers are carried out to always maintain the displacements ($V_d[i]$) according to this position, revising only its orientation.

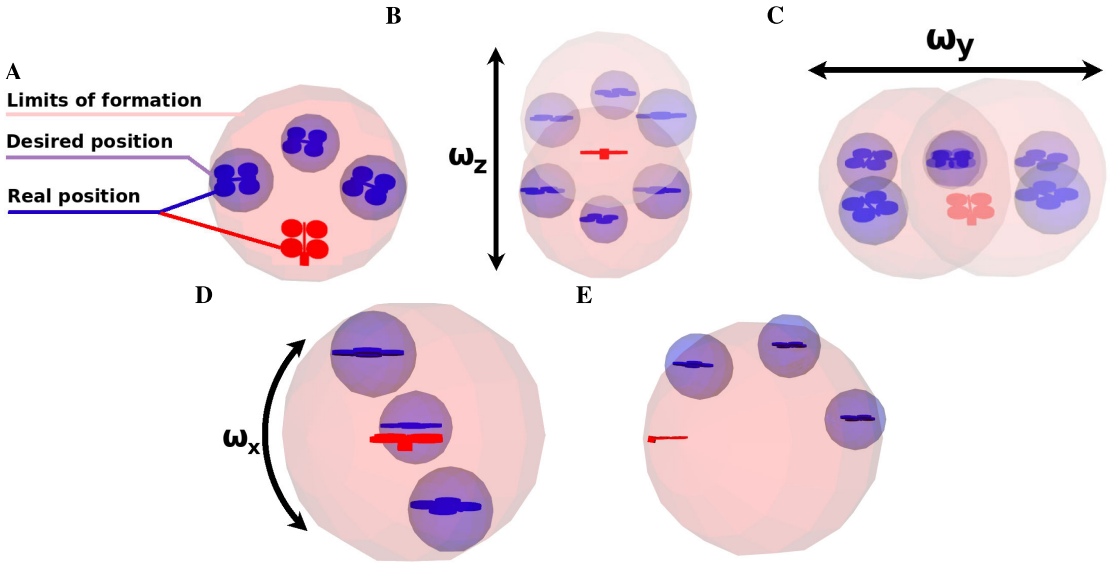


FIGURE 11. The virtual representation of the formation flight, presenting the limits and positions imposed on each drone, as well as the possible deviation maneuvers of the whole formation. (A)Formation limits set by virtual markers, the control proposed by this paper aims to ensure that each drone stays within specific limits. (B) Deviation of all formation in linear Z-axis (when there are obstacles above or below the formation). (C) Deviation of all formation in linear Y-axis (when there are obstacles to the right or left of the formation). (D) Deviation of all formation in the angular X-axis. (E)Line formation (when there are obstacles on both sides).

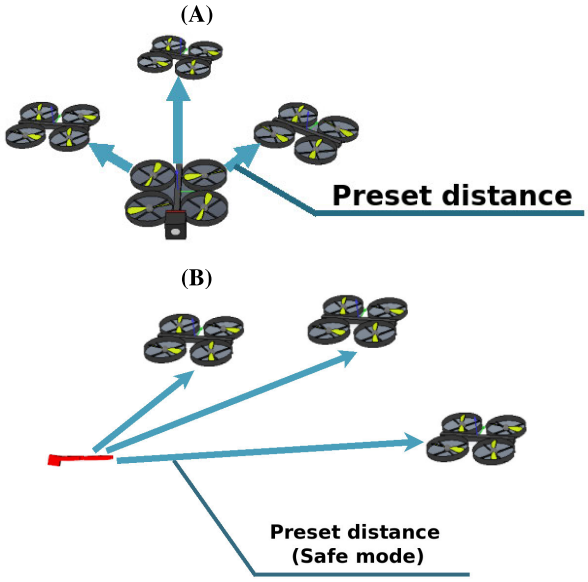


FIGURE 12. Pre-established drone formation. (A) Example of drones' position in cruiser formation. (B) Example of drones' position in safe formation (line mode).

2) SAFE FORMATION

When there are obstacles on both sides of the leader drone, at a distance of fewer than one meter on each leader side, the formation must assume the *safe mode*. It consists of a predefined geometrical shape in which worker drones form a shell behind the leader drone, causing all drones to pass through the same space. This shape assures no drone will hit off the charge, unlike what would happen if the formation rotated 90 degrees along the x-axis (roll rotation V_x in Figure 2.A). The worker drones' relative position computation is also given by Equation 4, considering that the distance vector (V_d) now corresponds to position for this safe formation as presented in Figure 12.B.

3) OBSTACLE DEVIATION

In the presence of an external obstacle, workers can slide around the leader to maintain formation while avoiding the obstacle. Rotation and translation matrices are used to modify the desired position of each worker drone in order to carry out the detour maneuver. The rotation movement changes the position of the worker drone without drawing it from the formation. In this way, the preset distance showed in Figure 12.A is always kept; only the corresponding vector is rotated around the leader drone.

Therefore, the procedure for obstacle deviation by the formation can be implemented into two steps. In the first one, the deviation obstacles fuzzy system infers the angle needed to execute the detour. In the second step, the desired position of each drone is rotated by this angle, considering the leader orientation.

The obstacle information used by the deviation obstacles fuzzy system is the same input variables of the fuzzy obstacles avoidance system in section III-A.2. They are δ_e, δ_y and δ_z. Their universes of discourse, fuzzy sets, and associated membership functions are the same given in Figure 5.D. δ_e refers to the Euclidean distance in meters between the leader and

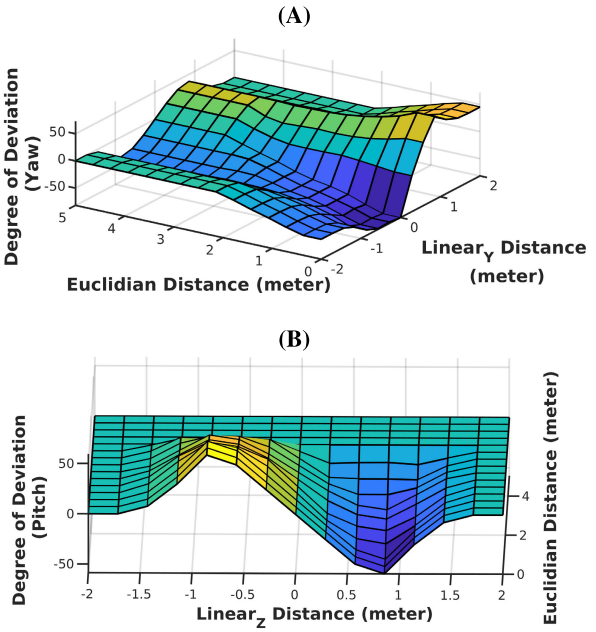


FIGURE 13. Surface graph of the Fuzzy system used for the diversion of obstacles in formation. (A) Degrees applied to the deviation in Yaw, about the linear error (Euclidean distance) and linear error in Y-axis (Y-axis difference) between the leader drone position and obstacle detected. (B) Degrees applied to the deviation in Pitch, about the linear error (Euclidean distance) and linear error in Z-axis (Z-axis difference) between the leader drone position and obstacle detected.

the closest detected obstacle. δ_y refers to the orientation of the obstacle, considering the leader drone current position, whether it is on the right or the left of the leader, and how many meters. δz refers to the obstacle orientation related to the leader drone, whether it is above or below the leader drone, in meters.

The proposed fuzzy system to define the angular deviation of the formation has two outputs: the yaw angle (δ_{yaw}) around the Z-axis and the pitch angle (δ_{pitch}) around Y-axis (V_z and V_y rotation in Figure 2.A). Both outputs are defined over the same universe of discourse $\delta_{yaw}, \delta_{pitch} \in [-150°, 150°]$ and their fuzzy sets have five triangular membership functions, equally spaced over this universe. Yaw deviation angle (δ_{yaw}) provides an angular value that will be added to the formation calculation so that all the worker drones rotate around the leading drone. If δ_{yaw} is positive, the drones will advance to the right; if it is negative, they will move to the left. In the same way, if δ_{pitch} is positive, the workers move up above the leader's position, if it is negative, the movement is to the bottom.

Figure 13 presents the surface graphs of both fuzzy subsystems. It is possible to observe that closer the obstacle is, the higher the angular deviation. These computed angular deviations are summed up to the initial positions of the cruiser formation discussed in section III-D.1. Then the rotation and translation matrices are applied to compute the news positions to be assumed by all working drones so that the formation always maintains the distance between all the drones, thus preserving the original formation geometrical shape.

Figure 11 illustrates the possible motions for the formation, where ω_z refers to the rotation around the leader drone on the Z-axis. This rotation is given by the outputδ_{yaw} of the fuzzy system added to leader drone actual orientation relative to z-axis $Lw.z$. A similar motion is carried out in the y-axis where the rotation ω_y of the working drones around the leader is established from the output δ_{pitch} of the fuzzy system added to leader drone actual angle $Lw.y$ relative to the y-axis. The rotation ω_x concerns the ability of the working drones to rotate around the leader drone along the x-axis. This rotation, also shown in Figure 11.D, is only applied in safe formation mode and corresponds to the roll angle δ_{roll} added to leader drone actual orientation relative to z-axis $Lw.z$.

The rotation values ω_x, ω_z, and ω_y are applied in sequence, considering the working drones' position Wdp, the leader position Lp and the vector Vd. This last vector indicates the displacement vector of each worker drone about the leader drone characterizing the formation of geometrical shape (cruiser or safe), as discussed above.

Equation 5 computes the 3D coordinates position values $[x, y, z]$ for the worker drones considering a rotation along x-axis ($\omega_x = \delta_{roll} + Lw.x$).

$$\begin{aligned} Wdp[i].x &= (Vd[i].x); \\ Wdp[i].y &= (Vd[i].y * \cos(\delta_{roll} + Lw.x)) \\ &\quad -(Vd[i].z * \sin(\delta_{roll} + Lw.x)); \\ Wdp[i].z &= (Vd[i].y * \sin(\delta_{roll} + Lw.x)) \\ &\quad +(Vd[i].z * \cos(\delta_{roll} + Lw.x)); \end{aligned} \tag{5}$$

After computing the ω_x rotation, it is possible to compute rotation along z-axis ($\omega_z = \delta_{yaw} + Lw.z.$) through equation 6. Note that Vd is no longer used; instead, we use now Wdp computed by equation 5 that corresponds to the new worker drone position already rotated by ω_x.

$$\begin{aligned} Wdp[i].x &= (Wdp[i].x * \cos(\delta_{yaw} + Lw.z)) \\ &\quad -(Wdp[i].z * \sin(\delta_{yaw} + Lw.z)); \\ Wdp[i].y &= Wdp[i].y; \\ Wdp[i].z &= (Wdp[i].x * \sin(\delta_{yaw} + Lw.z)) \\ &\quad +(Wdp[i].z * \cos(\delta_{yaw} + Lw.z)); \end{aligned} \tag{6}$$

Finally, the rotation along y axis can be computed by Equation 7 ($\omega_y = \delta_{pitch} + Lw.y.$), considering Wdp compute by equation 6.

$$\begin{aligned} Wdp[i].x &= (Wdp[i].x * \cos(\delta_{pitch} + Lw.y)) \\ &\quad -(Wdp[i].y * \sin(\delta_{pitch} + Lw.y)); \\ Wdp[i].y &= (Wdp[i].x * \cos(\delta_{pitch} + Lw.y)) \\ &\quad +(Wdp[i].y * \sin(\delta_{pitch} + Lw.y)); \\ Wdp[i].z &= Wdp[i].z; \end{aligned} \tag{7}$$

All rotations, ω_x, ω_z, and ω_y are computed based on the displacement vector of each working drone and angular

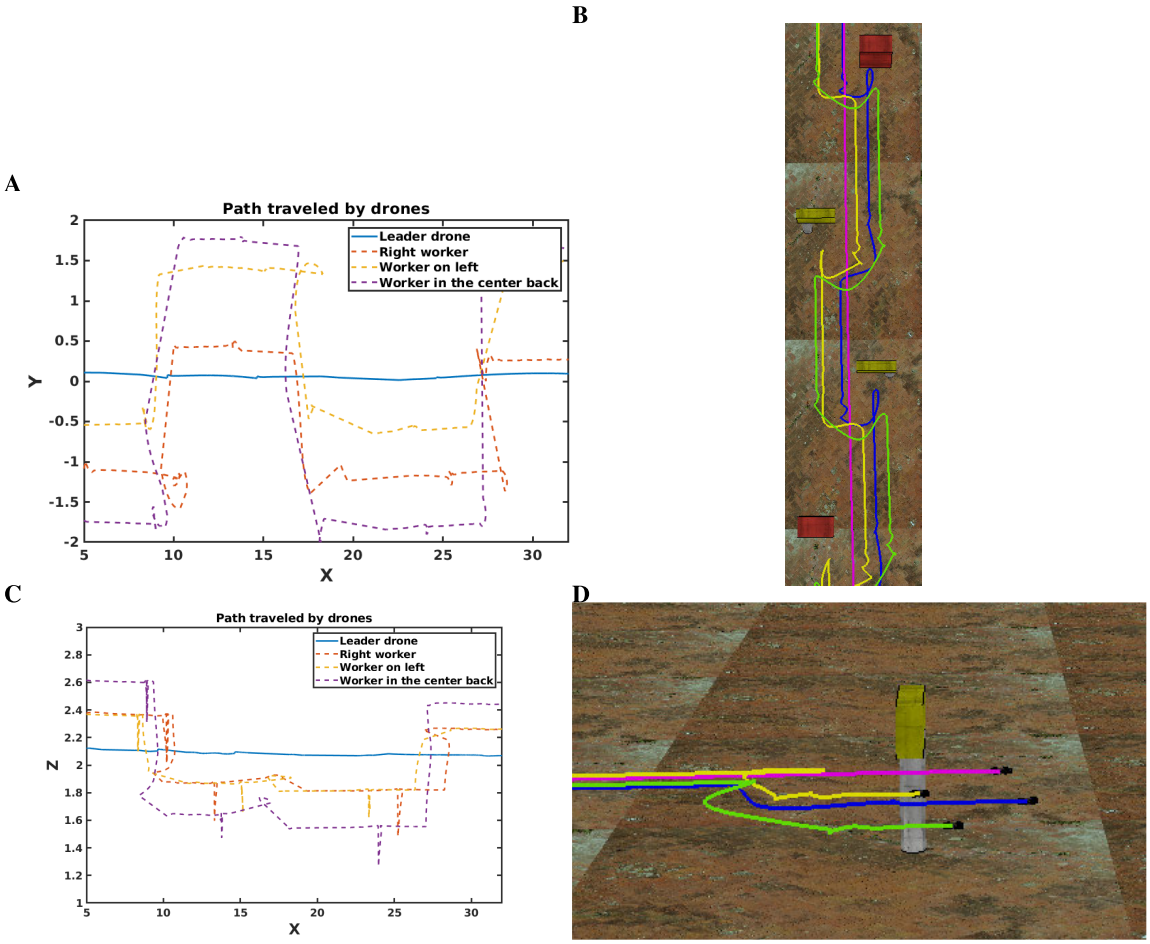


FIGURE 14. Graphical representation of the environment and path taken during navigation. (A) Graph representing the path traveled by all drones on the Y-axis. (B) Graphical representation of the path traveled by the drones on the Y-axis. (V-rep simulator). (C) Graph representing the path traveled by all drones on the Z-axis. (D) Graphical representation of the path traveled by the drones on the Z-axis. (V-rep simulator).

orientation of the leader drone. The resulting position *Wdp* must be now translated, considering the actual position of the leader drone, in order to maintain the formation. This motion is carried out through Equation 8, where *Lp* refers to the actual position of the leader drone in 3D space.

$$\begin{aligned} Wdp[i].x &= Wdp[i].x + Lp.x; \\ Wdp[i].y &= Wdp[i].y + Lp.y; \\ Wdp[i].z &= Wdp[i].z + Lp.z; \end{aligned} \tag{8}$$

However, as discussed in section III-C, the position control of the drone interleaves between avoiding collisions with other drones or going to the desired position. The result of the formation maintenance module only generates a new desired point; the decision about to reach or not the desired position depends on the self-preservation system.

IV. EXPERIMENTAL RESULTS

The proposed *quadral-fuzzy* approach has been tested and validated through simulated experiments using the *Virtual Robot Experimentation Platform* (V-Rep) [37]. Firstly, each module has been one to one tested, and finally, the complete system was validated.

This simulated experiment is carried out with a four drones fleet. The cruiser formation has the leader in front of the platoon and in the back line, a worker drone just behind the leader, one on its left and the other on the right, as shown in Figure 12.A. The three workers drones in the backline keep the same Euclidean distance from the leader.

A scene with several obstacles has been build to asses *quadral-fuzzy* approach performance. This scene is shown in Figure 14.B contains several obstacles that force the fleet to perform deviation maneuvers in the three axes of the 3D space. The drones must traverse the whole scene without colliding with any obstacle and with each other while maintaining the formation.

The position of the four drones is monitored throughout the course. Figure 14) presents the complete traveled paths by all drones during the simulation. The goal is to verify if the formation is maintained during flight navigation, even

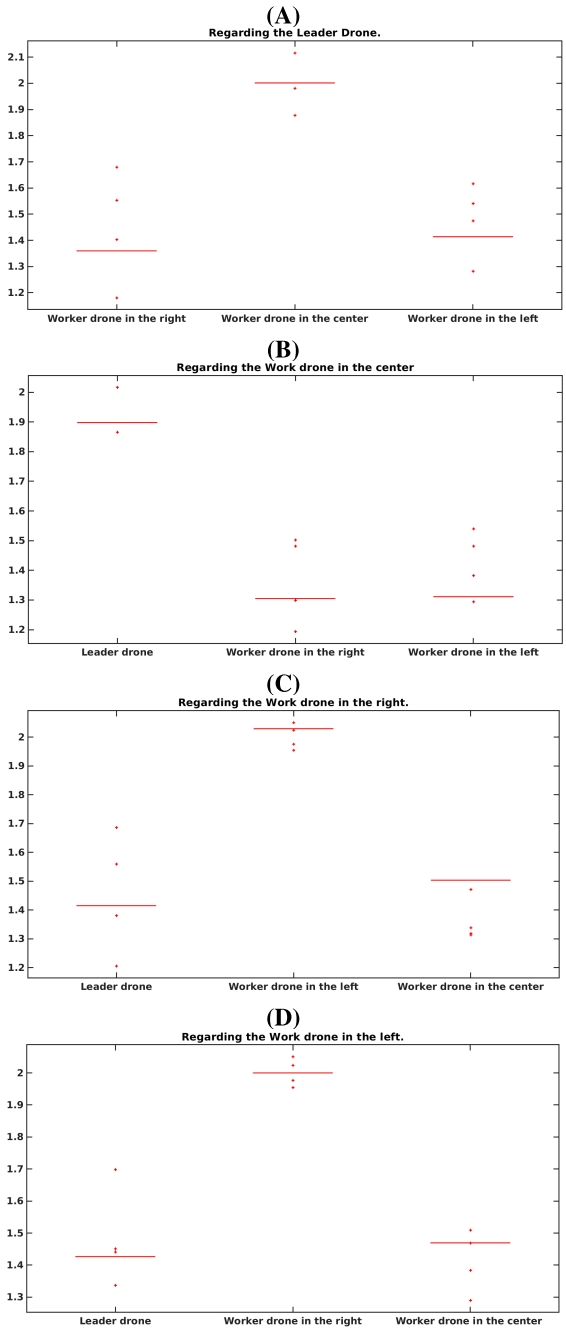


FIGURE 15. Boxplot representing the results obtained during navigation. (A) Distance from working drones to drone leader. (B) Distance between the drones in relation to the middle drone. (C) Distance between the drones in relation to the right drone. (D) Distance between the drones in relation to the left drone.

in the diversion of obstacles. An analysis of the distance between the drones is carried out to show the efficacy of the self-preservation system. Figures 14) A and C show, respectively, the paths developed by all drones in the X-Y and X-Z plans. Figures 14) B and D are respectively, the top views of the trajectories in the X-Y plan and the side view of the trajectories in the X-Z plan. From the latest figures, we can note that some of the drones have sometimes flown in the loop (ellipses shape). This occurred when the leader drone identified an obstacle close to the worker drone that has been forced to carry an evasive action to avoid collisions. After the maneuver, the drone accomplishes a loop to resume its path.

During the simulation, the position of each drone was taken in a rate of 0.2 seconds, when the Euclidean distance between them was calculated. Those measurements are presented in the boxplots at Figure 15 having each drone as reference: Figure 15.A shows the distances from the leader to all other drones, while Figure 15.B shows the Euclidean distance of the middle worker drone to the others, and so on.

As discussed in previous sections, the goal is to keep the distance between the drones throughout the navigation, causing them to fly in formation and do not crash at any moment. As shown in the results, this objective was reached, showing that most of the time the distance between the drones was preserved, with the exception of a few moments, where the obstacle diversion action is performed, moving from a maximum of 20 centimeters.

V. CONCLUSION

This work has presented a *quadral-fuzzy* approach for flight formation using multiple drones. This approach can be applied for several purposes, such as cooperative transport of goods, or patrolling in multi-robot environments, for example. The approach is composed of four central systems: leader drone control, worker drones control, self-preservation, and formation maintenance.

The leader drone commands the platoon since this drone is equipped with a 3D sensor to map the environment. The leader's flight control is toggled between two subsystems: position control and obstacle bypass control. The switching between these two subsystems can determine a change in the positions of worker drones in order to maintain formation. The position control of worker drones is very simple. It only receives a point to be reached and moves the drone towards it. This desired position can be modified at any time during a flight. The self-preservation system prevents collision among the drones inside the formation. The deviation maneuvers are based on the possibility (*fuzziness*) degree of a drone moving in a particular direction. If two drones are detected inside a path intersection area, a repulsive force proportional to their *fuzziness* degree is applied moving away both drones.

The formation maintenance strategy consists in continually assigning positions to the worker's drones based on actual leader drone position and information about external obstacles detected by the perception 3D sensor embedded in the leader drone. In the presence of a detected obstacle, the formation is rotated, causing all drones to deviate from it. If there are several detected obstacles, the fleet can quit cruiser formation, assuming an in-line safety formation until there are no more obstacles. Simulation experiments have been carried out, showing that the proposed approach provides safe

navigation for multiple drones, avoiding collisions between the drones and with obstacles presented in the environment.

REFERENCES

[1] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, "Car detection using unmanned aerial vehicles: Comparison between faster R-CNN and YOLOv3," in *Proc. 1st Int. Conf. Unmanned Vehicle Syst.-Oman (UVS)*, Feb. 2019, pp. 1–6.

[2] A. Koubaa and B. Qureshi, "Dronetrack: Cloud-based real-time object tracking using unmanned aerial vehicles over the Internet," *IEEE Access*, vol. 6, pp. 13810–13824, 2018, doi: 10.1109/ACCESS.2018.2811762.

[3] E. T. Alotaibi, S. S. Alqefari, and A. Koubaa, "LSAR: Multi-UAV collaboration for search and rescue missions," *IEEE Access*, vol. 7, pp. 55817–55832, 2019.

[4] N. Smolyanskiy and M. Gonzalez-Franco, "Stereoscopic first person view system for drone navigation," *Frontiers Robot. AI*, vol. 4, p. 11, Mar. 2017.

[5] D. Tezza and M. Andujar, "The state-of-the-art of human–drone interaction: A survey," *IEEE Access*, vol. 7, pp. 167438–167454, 2019.

[6] A. Otto, N. Agatz, J. Campbell, B. Golden, and E. Pesch, "Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey," *Networks*, vol. 72, no. 4, pp. 411–458, Dec. 2018.

[7] R. D'Andrea, "Guest editorial can drones deliver?" *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 647–648, Jul. 2014.

[8] E. Tuci, M. H. M. Alkilabi, and O. Akanyeti, "Cooperative object transport in multi-robot systems: A review of the state-of-the-art," *Frontiers Robot. AI*, vol. 5, p. 59, 5 2018. [Online]. Available: https://doi.org/10.3389/frobt.2018.00059

[9] J. Peterson, H. Chaudhry, K. Abdelatty, J. Bird, and K. Kochersberger, "Online aerial terrain mapping for ground robot navigation," *Sensors*, vol. 18, no. 2, p. 630, 2018.

[10] J. Palacin, I. Valgañon, and R. Pernia, "The optical mouse for indoor mobile robot odometry measurement," *Sens. Actuators A, Phys.*, vol. 126, no. 1, pp. 141–147, Jan. 2006.

[11] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, 2004, p. 1.

[12] T. Paul, T. R. Krogstad, and J. T. Gravdahl, "Modelling of UAV formation flight using 3D potential field," *Simul. Model. Pract. Theory*, vol. 16, p. 1453–1462, Oct. 2008.

[13] C. Kanellakis and G. Nikolakopoulos, "Survey on computer vision for UAVs: Current developments and trends," *J. Intell. Robotic Syst.*, vol. 87, no. 1, pp. 141–168, Jul. 2017.

[14] M. A. S. Teixeira, H. B. Santos, A. S. de Oliveira, L. V. Arruda, and F. Neves, "Robots perception through 3D point cloud sensors," in *Robot Operating System (ROS)*, vol. 707, A. Koubaa, Ed. Cham, Switzerland: Springer, 2017, pp. 525–561. [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-3-319-54927-9_16, doi: 10.1007/978-3-319-54927-9_16.

[15] H. Mo and G. Farid, "Nonlinear and adaptive intelligent control techniques for quadrotor UAV—A survey," *Asian J. Control*, vol. 21, no. 2, pp. 989–1008, Mar. 2019.

[16] M. Varga, J.-C. Zufferey, G. H. M. Heitz, and D. Floreano, "Evaluation of control strategies for fixed-wing drones following slow-moving ground agents," *Robot. Auto. Syst.*, vol. 72, pp. 285–294, Oct. 2015.

[17] J. Hu and A. Lanzon, "An innovative tri-rotor drone and associated distributed aerial drone swarm control," *Robot. Auto. Syst.*, vol. 103, pp. 162–174, May 2018.

[18] C. P. Bechlioulis and K. J. Kyriakopoulos, "Collaborative multi-robot transportation in obstacle-cluttered environments via implicit communication," *Frontiers Robot. AI*, vol. 5, p. 90, Aug. 2018. [Online]. Available: https://www.frontiersin.org/article/10.3389/frobt.2018.00090

[19] X. Hou, Z. Ren, J. Wang, S. Zheng, W. Cheng, and H. Zhang, "Distributed fog computing for latency and reliability guaranteed swarm of drones," *IEEE Access*, vol. 8, pp. 7117–7130, 2020.

[20] X. Wang, V. Yadav, and S. N. Balakrishnan, "Cooperative UAV formation flying with Obstacle/Collision avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 4, pp. 672–679, Jul. 2007.

[21] L. Zhu, X. Cheng, and F.-G. Yuan, "A 3D collision avoidance strategy for UAV with physical constraints," *Measurement*, vol. 77, pp. 40–49, Jan. 2016.

[22] J. Dufek and R. Murphy, "Visual pose estimation of rescue unmanned surface vehicle from unmanned aerial system," *Frontiers Robot. AI*, vol. 6, p. 42, May 2019. [Online]. Available: https://www.frontiersin.org/article/10.3389/frobt.2019.00042

[23] X. Xiang, C. Yu, L. Lapierre, J. Zhang, and Q. Zhang, "Survey on Fuzzy-Logic-Based guidance and control of marine surface vehicles and underwater vehicles," *Int. J. Fuzzy Syst.*, vol. 20, no. 2, pp. 572–586, Feb. 2018.

[24] D. R. Parhi and M. K. Singh, "Navigational strategies of mobile robots: A review," *Int. J. Autom. Control*, vol. 3, nos. 2–3, p. 114–134, 2009.

[25] J. P. L. S. de Almeida, R. T. Nakashima, F. Neves-Jr, and L. V. R. de Arruda, "Bio-inspired on-line path planner for cooperative exploration of unknown environment by a multi-robot system," *Robot. Auto. Syst.*, vol. 112, pp. 32–48, 2019.

[26] J. Alcala-Fdez and J. M. Alonso, "A survey of fuzzy systems software: Taxonomy, current research trends, and prospects," *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 1, pp. 40–56, Feb. 2016.

[27] F. Santoso, M. A. Garratt, S. G. Anavatti, and I. Petersen, "Robust hybrid nonlinear control systems for the dynamics of a quadcopter drone," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Jun. 4, 2018, doi: 10.1109/TSMC.2018.2836922.

[28] Z. Peng, S. Yang, G. Wen, A. Rahmani, and Y. Yu, "Adaptive distributed formation control for multiple nonholonomic wheeled mobile robots," *Neurocomputing*, vol. 173, pp. 1485–1494, Jan. 2016.

[29] H. Xiao, Z. Li, and C. L. P. Chen, "Formation control of Leader–Follower mobile robots' systems using model predictive control based on neural-dynamic optimization," *IEEE Trans. Ind. Electron.*, vol. 63, no. 9, pp. 5752–5762, Sep. 2016.

[30] X. Zhu, C. Bian, Y. Chen, and S. Chen, "A low latency clustering method for large-scale drone swarms," *IEEE Access*, vol. 7, pp. 186260–186267, 2019.

[31] L. Bian, W. Sun, and T. Sun, "Trajectory following and improved differential evolution solution for rapid forming of UAV formation," *IEEE Access*, vol. 7, pp. 169599–169613, 2019.

[32] L.-X. Wang, *A Course in Fuzzy Systems and Control*. Upper Saddle River, NJ, USA: Prentice-Hall, 1997.

[33] R. P. Paul, *Robot Manipulators: Mathematics, Programming, and Control: The Computer Control of Robot Manipulators*. Cambridge, MA, USA: MIT Press, 1981.

[34] W. Khalil and E. Dombre, *Modeling, Identification & Control of Robots*. London, U.K.: Butterworth, 2004.

[35] A. S. de Oliveira, E. R. De Pieri, U. F. Moreno, and D. Martins, "A new approach to singularity-free inverse kinematics using dual-quaternionic error chains in the Davies method," *Robotica*, vol. 34, no. 4, pp. 942–956, Apr. 2016.

[36] MESA IMAGING. (2011). *SR4000/SR4500 User Manual, Version 3.0 Ed.* [Online]. Available: http://downloads.mesa-imaging.ch/dlm.phpfname=customer/Customer_CD/SR4000_SR4500-_Manual.pdf

[37] E. Rohmer, S. P. N. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 1321–1326.

MARCO ANTONIO SIMOES TEIXEIRA received the M.Sc. degree in electrical and computer engineering from the Federal University of Technology—Parana (UTFPR), Brazil, in 2017, where he is currently pursuing the Ph.D. degree in electrical and computer engineering. His research interests comprise mobile robots in the areas of perception and intelligent systems, computer vision, navigation, mapping, and applied AI.

FLAVIO NEVES-JR received the B.Sc. and M.Sc. degrees in electrical engineering the from Federal University of Technology—Parana, Brazil, in 1987 and 1989, respectively, and the Ph.D. degree in electrical engineering from University Paul Sabatier (LAAS), France, in 1998. Since 1992, he has been with the Federal University of Technology—Parana, and actually, he is a Full Professor. His research interests comprise hardware and software to instrumentation and automation.

ANIS KOUBAA (Member, IEEE) is currently a Professor of computer science and the Leader of the Robotics and Internet of Things Research Laboratory, Prince Sultan University. He is also a Senior Researcher with CISTER and ISEP-IPP, Porto, Portugal, and a Research and Development Consultant with Gaitech Robotics, China. His current research deals with providing solutions toward the integration of robots and drones into the Internet of Things (IoT) and clouds, in the context of cloud robotics. His research interests also include robot operating systems (ROSs), robotic software engineering, wireless communication for the IoT, real-time communication, safety and security for cloud robotics, intelligent algorithms' design for mobile robots, and multirobot task allocation. He is a Senior Fellow of the Higher Education Academy (HEA), U.K. He has been the Chair of the ACM Chapter in Saudi Arabia, since 2014.

LUCIA VALERIA RAMOS DE ARRUDA (Member, IEEE) received the degree in electrical engineering from the Federal University of Ceara, Brazil, in 1985, the M.Sc. degree from the Graduate School of Electrical Engineering, Campinas State University (FEE/UNICAMP), Brazil, in 1988, and the Ph.D. degree in electrical engineering from the University of Nice-Sophia Antipolis, France, in 1992. Since 1995, she has been with the Federal University of Technology—Parana, and actually, she is a Full Professor. Her research interests comprise soft computing methods to model and control of dynamic systems.

ANDRE SCHNEIDER DE OLIVEIRA (Member, IEEE) received the M.Sc. degree in mechanical engineering, focused in force control of rigid manipulators, from the Federal University of Santa Catarina (UFSC), in 2007, and the Ph.D. degree in engineering of automation and systems, in 2011, with thesis focused on differential kinematics through dualquaternions for vehicle-manipulator systems. He is currently an Adjunct Professor with the Federal University of Technology—Parana (UTFPR). He is a member of the Advanced Laboratory of Robotics and Embedded Systems (LASER) and the Automation and Advanced Control System Laboratory (LASCA). His research interests include the areas of robotics, mechatronics, and automation with special focus in navigation and localization of mobile robots; autonomous and intelligent systems; perception and environment identification; cognition and deliberative decisions; and human-interaction and navigation control.

5 INTELLIGENT 3D PERCEPTION SYSTEM FOR SEMANTIC DESCRIPTION AND DYNAMIC INTERACTION

This chapter presents the paper published in the *Sensors* journal (ISSN: 1424-8220). The data of the paper is shown in Table 3.

Table 3 – Data of the paper *Intelligent 3D perception system for semantic description and dynamic interaction.*

Authors	Teixeira, M.A.S.; Nogueira, R.C.M.; Dalmedico, N.; Santos, H.B.; Arruda, L.V.R.; Neves-Jr, F.; Pipa, D.R.; Ramos, J.E.; Oliveira, A.S.
Title	*Intelligent 3D Perception System for Semantic Description and Dynamic Interaction*
Journal	*Sensors*
ISSN	1424-8220
DOI	https://doi.org/10.3390/s19173764
Publication date	August 30, 2019

Source: Own authorship.

This paper's motivation is that the data from the traditional sensors used in mobile robots, in most cases, are not enough for decision making, requiring the processing and fusion of sensors. For example, to know an object's position in a 3D environment, it is necessary to recognize the object in an image and then merge it with the depth data.

It is worth mentioning that the sensors fulfill the function for which they designed, providing raw data, but the data needs to be processed to generate information. Thus, this paper aims to propose a new sensing approach, where the data provided by the sensors will be processed, and useful information will be returned for the robot to use for decision making.

Regarding the theme of this thesis, this paper fulfills the third specific objective. The approach proposed in this paper was developed based on the sensing problems found in chapters 3 and 4, where studies of traditional sensing techniques were carried out.

sensors

MDPI

Article

Intelligent 3D Perception System for Semantic Description and Dynamic Interaction

Marco Antonio Simoes Teixeira [1,*], Rafael de Castro Martins Nogueira [1], Nicolas Dalmedico [1], Higor Barbosa Santos [1], Lucia Valeria Ramos de Arruda [1], Flavio Neves-Jr [1], Daniel Rodrigues Pipa [1], Julio Endress Ramos [2] and Andre Schneider de Oliveira [1]

[1] Graduate School of Electrical Engineering and Computer Science (CPGEI), Federal University of Technology-Paraná (UTFPR), Avenida 7 de Setembro 3165, Curitiba 80230-901, Brazil
[2] CENPES, Rio de Janeiro 21941-915, Brazil
* Correspondence: marcoteixeira@alunos.utfpr.edu.br

Received: 30 June 2019; Accepted: 13 August 2019; Published: 30 August 2019

Abstract: This work proposes a novel semantic perception system based on computer vision and machine learning techniques. The main goal is to identify objects in the environment and extract their characteristics, allowing a dynamic interaction with the environment. The system is composed of a GPU processing source and a 3D vision sensor that provides RGB image and PointCloud data. The perception system is structured in three steps: Lexical Analysis, Syntax Analysis and finally an Analysis of Anticipation. The Lexical Analysis detects the actual position of the objects (or tokens) in the environment, through the combination of RGB image and PointCloud, surveying their characteristics. All information extracted from the tokens will be used to retrieve relevant features such as object velocity, acceleration and direction during the Syntax Analysis step. The anticipation step predicts future behaviors for these dynamic objects, promoting an interaction with them in terms of collisions, pull, and push actions. As a result, the proposed perception source can assign relevant information to mobile robots, not only about distances as traditional sensors, but about other environment characteristics and object behaviors. This novel perception source introduces a new class of skills to mobile robots. Experimental results obtained with a real robot are presented, showing the proposed perception source efficacy and potential.

Keywords: perception; pointCloud; prediction and object recognition

1. Introduction

Nowadays, mobile robotics is one of the most prominent research areas [1,2]. Robots with the ability to freely move through an environment, knowing their positions and able to sense the world can execute intelligent tasks. Such robots may develop tasks, for example, in industries [3–5] and in rescue tasks [6–8], or replacing humans in hazardous work.

Such robots need to perceive the environment around them to identify obstacles and execute actions without collisions. That is, the robot needs to see the world before making some decision. For this, it uses sensors that allow for converting real-world data into digital information that can be used by robots and promote actions [9,10].

There are numerous types of sensors for mobile robotics. Some are popular due to their low cost and easy data manipulation, such as color image (RGB image) provided by cameras. These sensors can be used to identify Augmented Reality Tags (AR-Tags) [11,12], among other computational vision tasks [13–15]. Despite its worldwide use, the RGB sensor provides a simple image of the environment, without any specific characteristic that allows smart actions.

Intelligent behaviors are not achieved without a detailed understanding of the environment that allows a reliable mapping and safe obstacle avoidance. The depth sense is primordial to obtain a three-dimensional insight and execute instantaneous reactions in the presence of obstacles or planned deviations. RGB sensors provide a cloud of points (or PointClouds), which can collect spatial information from the environment. They are used in numerous works in mobile robotics [16–18], mainly towards obstacles avoidance and environment mapping.

However, this type of sensor provides only the notion of distance between the robot and nearby objects without identifying them. This lack of information causes the robot to consider all objects (dynamic, static, animate, or inanimate ones) in the same way. In summary, all objects are understood as a static obstacle in a scene simplification. In fact, the environment is commonly composed of a lot of moving objects, and a static sensing compromises the robot navigation since planning must consider instantaneous information that changes over time.

For example, consider a person walking inside the same room where a robot must navigate from point A to B. This robot has a depth sensor which sees all objects in the environment (including the person) and can plan the path to achieve point B avoiding all obstacles. Nevertheless, the person crosses the robot's path, and a new path is computed to avoid collision. This process can be eternal if the person continues to move and the robot always chooses the same side to implement an avoidance maneuver. To prevent such situation, It is primordial that more detailed information about the environment (for example, person motion forecast) is available to enhance intelligent behaviors, such as [19–21].

In addition to RGB sensors and depth sensors, other sensors can be used to add extra information to robots, such as temperature and localization, among others. The usefulness of this additional information requires a correlation analysis with primary sensors or simply a data fusion procedure that enhances a robot's understanding and or improves specific sensing. In fact, the use of a group of sensors does not effectively increase the robot knowledge about the environment, but only provides several independent information sources.

The transformation of raw information from the sensors into useful information for robots requires the execution of several procedures based on different techniques. Such procedures compose the so-called perception system that helps robots perform a dynamic interaction with scene elements to make inferences about features of the environment.

For instance, some papers use data from the RGB sensor to identify objects, and others combine such data with methods to segment objects in environments with 3D sensors [22,23]. These works manipulate raw sensor data to extract some useful information. The problem with this approach is that it should be done to each robot and a specific task, without any generalization.

There have been several object recognition techniques developed in recent years [24–26]. Deep learning techniques stand out as the best choice due to the best accuracy for object detection tasks [27–32]. Among them, one that stands out is YOLO (You Only Look Once) [33] due to its high accuracy and quickness in object recognition tasks. Such characteristics make YOLO a potential tool for being integrated into a sensor allowing a more detailed information catch.

This work proposes a novel intelligent perception source that introduces a new class of mobile robots skills through a detailed sensing of environment components. The approach rigorously decomposes RGB-D camera data through three analysis: lexical, syntax, and anticipation. Thus, decomposition is based on Semantic Description and Dynamic Interaction (SD2I). Mobile robots with SD2I perception will identify objects from the environment and understand the dynamic behavior of time-variant elements, allowing intelligent interaction.

This paper is organized as follows. A brief description of the used 3D perception sources is carried out in Section 2. The proposed perception system is developed in Section 3. Section 4 brings practical experiments and results and a discussion about the efficacy of the proposed system. Section 5 concludes the paper presenting an overview of the work.

2. Overview of Used 3D Perception Sources

The approach developed in this paper applies Semantic Description and Dynamic Interaction (SD2I) to collect information from the environment through RGB-D sensors and convert them into useful and intelligent information for a robot. In addition, we also present a compatible embedded hardware supporting such approach. An illustration of the proposed system, called the Intelligent 3D Perception (I3P) system, can be seen in Figure 1.

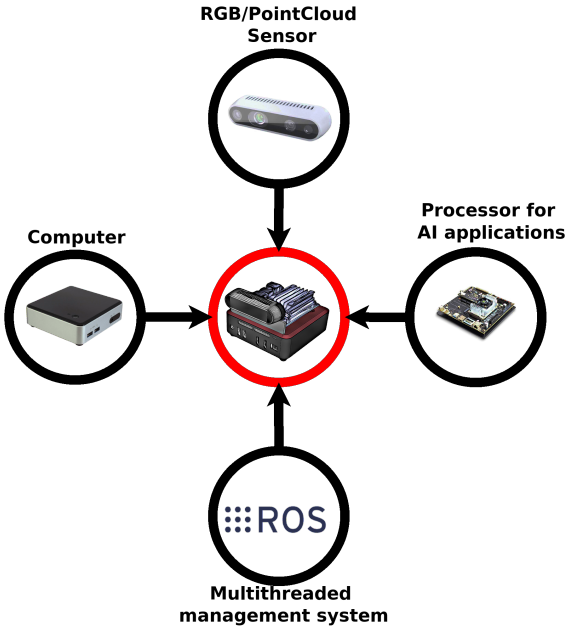


Figure 1. Intelligent 3D Perception (I3P) system.

The SD2I methodology is based on an embedded system with a 3D sensor. The PointCloud is acquired through an RGB-D sensor, which can be Intel RealSense [34], Kinect v1 or Kinect v2 [35], among others.

I3P system uses YOLO (You Only Look Once) to identify the objects by the RGB image from the source of perception. The third version of YOLO is available since 2018 [33]. In addition to the standard version, there is the "tiny-YOLO" version, which makes it possible to identify the same objects with lower precision. The main advantage of "tiny-YOLO" is a reduced number of layers, which makes it ideal to be embedded. Moreover, YOLO has showed good results when used with several different datasets, such as COCO [36], PASCAL VOC [37], and the KITTY dataset [38]. In this way, YOLO becomes suitable for object recognition in any previously cited databases.

A robot embedded device using YOLOv3 must have processing capacity to run a Convolutional Neural Network (CNN) [29]. Such devices make use of GPUs to execute these tasks; some examples are Jetson Tx1, Jetson Tx2, Jetson Nano or Jetson Xavier [39]. For the I3P system, this device is corresponding to *Processor for Ai application* in Figure 1.

The I3P required processor must be capable of collecting information from Points Cloud and from the Processor for the AI application, processing and returning information in a time period useful to the robot. A mini computer is chosen due to its processing power since the information delay due to an RGB-D sensor affects the run time of the proposed strategy. A suggested piece of equipment is Intel Nuc I5 (Santa Clara, CA, USA) [40] or similar.

The perception sensor is a distributed system composed of two processing sources and a 3D sensor. The I3P system management is based on an *Robot Operating System* (ROS) [41] to gather all information coming from different sources. ROS is a framework with tools that allow the use of parallel programming and distributed systems. Thus, the I3P system can operate with various sources of

processing, and perception, as well as providing a method for the robot to read the information from the sensor in a comfortable and agile way.

All components presented herein compose the hardware supporting the application of the approach proposed by this paper.This will be further detailed in Section 3, where the developed methodology for I3P system will be presented in detail.

3. Semantic Description and Dynamic Interaction (SD2I)

This paper presents a new intelligent perception source with a high abstraction level to mobile robots. It is called Semantic Description and Dynamic Interaction (SD2I), and it is shown in Figure 2. The main goal is the introduction of a new class of skills to mobile robots, allowing their interaction with a large group of time-varying objects from a crowded environment. The proposed methodology is conceived to be embedded into the Intelligent 3D Perception (I3P) system described above or another similar powerfully processor.

The SD2I methodology comprises three distinct steps to process and extract the environment features. The first step is the *Lexical Analysis*. In this step, the environment objects are identified and their attributes such as kind, height, width, confidence, and position are extracted. The second part, *Syntax Analysis*, consists of an element (Token) analysis, in which all objects' particular features like speed, acceleration, mass, and strength are estimated. Finally, the *Anticipation Analysis* uses the estimated parameters to infer future actions of dynamic objects. These analyses will be detailed in the following subsections.

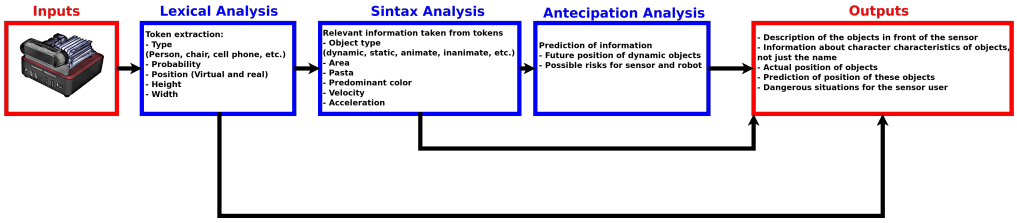


Figure 2. An overview of *Semantic Description and Dynamic Interaction* (SD2I) methodology.

3.1. Lexical Analysis

Lexical analysis is responsible for extracting information from the environment, generating objects (called *Tokens*) with specific characteristics. Therefore, each Token will contain various pieces of information about a single object. Figure 3 presents a diagram showing the steps of the lexical analysis.

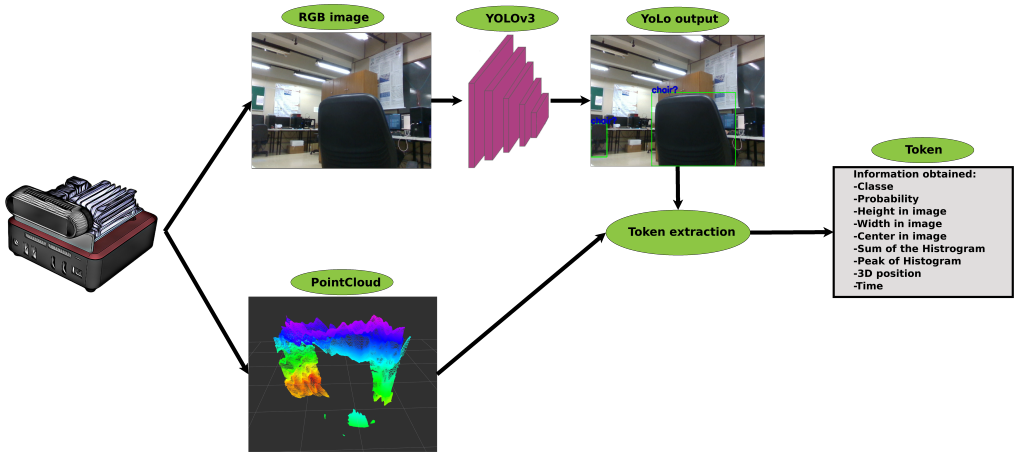


Figure 3. An overview of Lexical Analysis.

From Figure 3, all extracted information is associated with a Token. Such information are obtained from the RGB sensor (class, probability, center in pixel, width in pixel, height in pixel, sum of the histogram and peak of histogram) and the others come from the Depth sensor, as a spatial position of the object in the sensor frame. YOLOv3 is used for Token identification. For this, YOLOv3 has been trained with the COCO database [36] so that 80 different objects can be identified. The YOLOv3 convolutional neural network (CNN-YOLO) provides the name of the object, class, probability, and a box around the object from which the width and height are obtained. Figure 4 shows an example of data provided by the network from an image, and the strategy used to calculate height and width information.

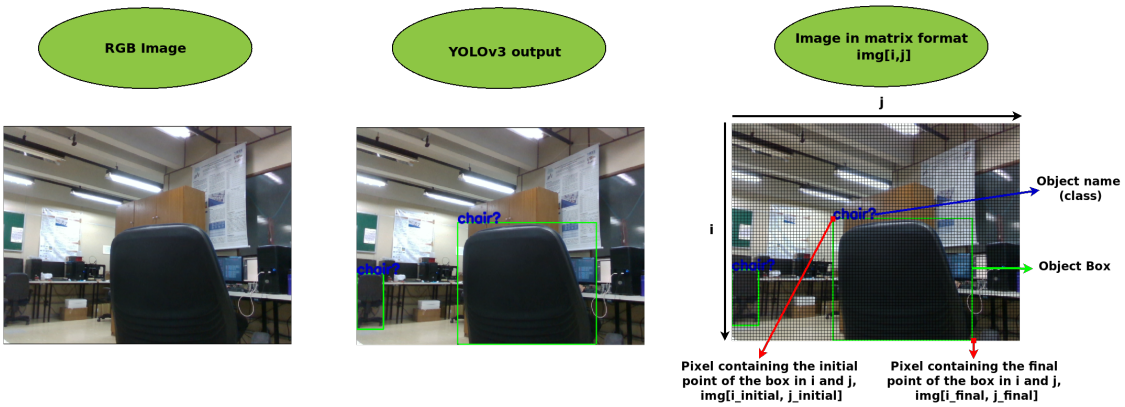


Figure 4. Extraction of the tokens from the image.

The class of Tokens are obtained as the output of CNN-YOLO without any pre-processing step. The other data need some processing for their extraction. In the lexical analysis, the objective is only to extract the relevant data; all relevant information will be obtained from this data in the next syntax analysis step. Therefore, Token extraction must be carried out with as little computational effort as possible.

From an object image, a subtraction is made between the i_final minus the $i\_initial$ pixels, as shown in Equation (1), in order to obtain the object's height information in pixel. The same procedure, using j (Equation (2)), is used to get the width. In order to obtain the central pixel of the object in the image, $center\_img[x, y]$, Equation (3) can be used. It is worth mentioning that the obtained position refers to the object in the image, not its actual location in the world, which will be obtained later. Therefore, some characteristics referring to the object and its position in the RGB image can be evaluated, for instance, height, width, and center:

$$height\_img = i\_final - i\_initial, \tag{1}$$

$$width\_img = j\_final - j\_initial, \tag{2}$$

$$center\_img = [(i\_initial + (height/2), j\_initial + (width/2)]. \tag{3}$$

The predominant gray tone from an image can be captured by a histogram [42,43]. Thus, such histogram is added as an object information and its peak is saved into the Token. This information may be interesting to identify if the object is light or dark, for example. Other information taken from the histogram is the sum of all values. This information can be used to identify the same object at different time states. Only the object being observed is cut out of the image to obtain the histogram for each element. After cutting the object, the image is converted to grayscale, and then its histogram is generated. Figure 5 gives an example, where the complete image is presented, the cut object is highlighted, and its histogram identified peak is computed.

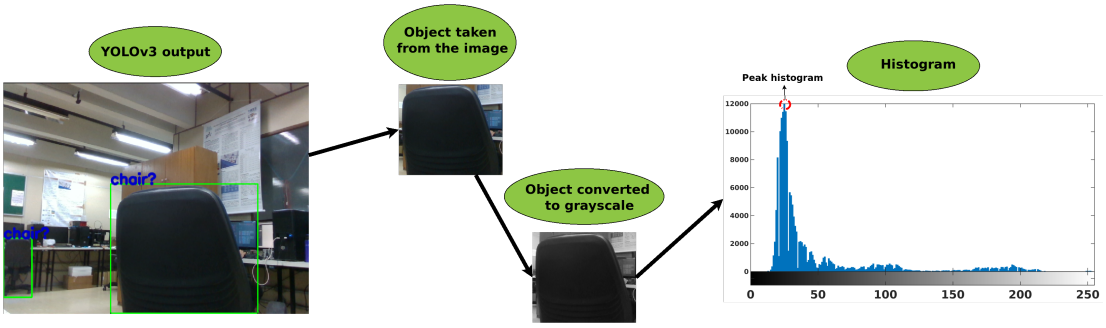


Figure 5. Analysis of histogram.

The 3D depth sensor associates each pixel of the image ($img[i, j]$) with a 3D point ($pc[x, y, z]$) of its point cloud. It is thus possible to identify the actual position of the object in the world, from the position of the object in the image. The proposed strategy consists of averaging the points relative to the object, resulting in an estimate of its position. However, the 3D sensor does not always provide the distance of all pixels; when, for different reasons, such information can not be read, the corresponding data position $img[i, j]$ are $pc[NaN, NaN, NaN]$ or some other invalid value. The method to avoid obtaining erroneous values consists of averaging the pixels that are in the box of the object, ignoring the values that are not valid. A second error can also happen when the box is larger than the object, generating 3D points that do not correspond to the object in question. This error is avoided with the use of only 60% of the box for averaging, ignoring the borders. The calculation of the new box consists of adding to the center of the object (*center_img*) 30% for more and for less of the height (*height_img*) for the values of i, and 30% for more and for less of the width (*width_img*) for the values of j.

Algorithm 1 presents the proposed strategy for identifying the object's 3D position, where the inputs are the point cloud provided by the depth sensor, with a resolution of 640x640, and each position contains a 3D point [X, Y, Z], named $pc[640, 480, 3]$. The second entry consists of the center of the object in the image ($center\_img[i, j]$). The third and fourth entries, respectively, are the object's height and width, in pixels, provided by the image, called *height_img* and *width_img*. The computation is made by averaging in X, Y, and Z of the points in the center of the object's box, reduced 40%. The output of the algorithm, a single 3D position $[X, Y, Z]$, is found containing the position of the object considered by the proposed strategy, named $p\_center[x, y, z]$.

This strategy is applied to all objects identified in the image, extracting the Tokens for all objects identified by YOLOv3. To review, Tokens are object name (class), obtained by YOLOv3; height of the object in the image (*height_img*) is obtained by Equation (1), width of the object in the image (*weidth_img*) is obtained by Equation (2), central pixel of the object in the image (*center_img*) is obtained by Equation (3), sum of histogram (*sum_histogram*) and histogram peak (*peak_histogram*) are as seen in Figure 5 and real position of the 3D object (*p_center*) is obtained by Algorithm (3). The next steps convert the Tokens into smart information.

Algorithm 1: Identification of the object's 3D center.

Input: PointCloud obtained by the 3D sensor, with dimension 640x480x3. $pc[640, 480, 3]$.
Input: Center of the object in the image. $center\_img[i, j]$
Input: Height of the object in the RGB image, in pixels. $height\_img$
Input: Width of the object in the RGB image, in pixels. $width\_img$
Output: 3D point corresponding to the position of the object in relation to the source of perception, $p\_center[x, y, z]$

$p\_center.x \leftarrow 0$ `/* Starts center x with zero */`
$p\_center.y \leftarrow 0$ `/* Starts center y with zero */`
$p\_center.z \leftarrow 0$ `/* Starts center z with zero */`

$counter \leftarrow 0$ `/* Counter used to calculate the average */`
$x \leftarrow 0$ `/* Variable used to save a sum of all x */`
$y \leftarrow 0$ `/* Variable used to save a sum of all y */`
$z \leftarrow 0$ `/* Variable used to save a sum of all z */`
for $i \leftarrow (center\_img.i - (height\_img * 0.30)) : 1 : (center\_img.i + (height\_img * 0.30))$ **do**
  for $j \leftarrow (center\_img.j - (width\_img * 0.30)) : 1 : (center\_img.j + (width\_img * 0.30))$ **do**
    `/* notNull is used to check if the value is valid, not null or inf or NaN */`
    if $notNull(pc[i, j, 0])$ *and* $notNull(pc[i, j, 1])$ *and* $notNull(pc[i, j, 2])$ **then**
      $x \leftarrow x + pc[i, j, 0]$
      $y \leftarrow y + pc[i, j, 1]$
      $z \leftarrow z + pc[i, j, 2]$
      $counter \leftarrow (counter + 1)$
    end
  end
end
if $counter! = 0$ **then**
  $p\_center.x \leftarrow (x/counter)$
  $p\_center.y \leftarrow (y/counter)$
  $p\_center.z \leftarrow (z/counter)$
end
return $p\_center[x, y, z]$

3.2. Syntax Analysis

Syntax analysis aims to convert information from lexical analysis into useful information for the robot. At the end of this analysis, the object will be classified as dynamic or non-dynamic, animate or inanimate, and its speed, direction, and acceleration will be calculated if the element is time-variant. Figure 6 gives an overview of the Analysis syntax.

The objects will be divided into categories, as static or dynamic objects and animate or inanimate objects. When the objects are classified as static or dynamic objects, the robot can make use of this information to know if there is a risk of an object moving around. Knowing if the object is animate or inanimate, the robot can use this information to measure its actions, to avoid acting dangerously and risking its integrity. Table 1 presents examples of objects that belong to dynamic category and life category (animate); other objects that don't belong to any of the two types are static or inanimate. A list of such objects can be accessed in [36].

The computation of dynamic objects' velocity, direction, and acceleration consist of collecting the Tokens at two different time moments. The variable t and $t - 1$ will be used to respectively indicate present and past into the vector of token $token\_vector[time][token\_id]$.

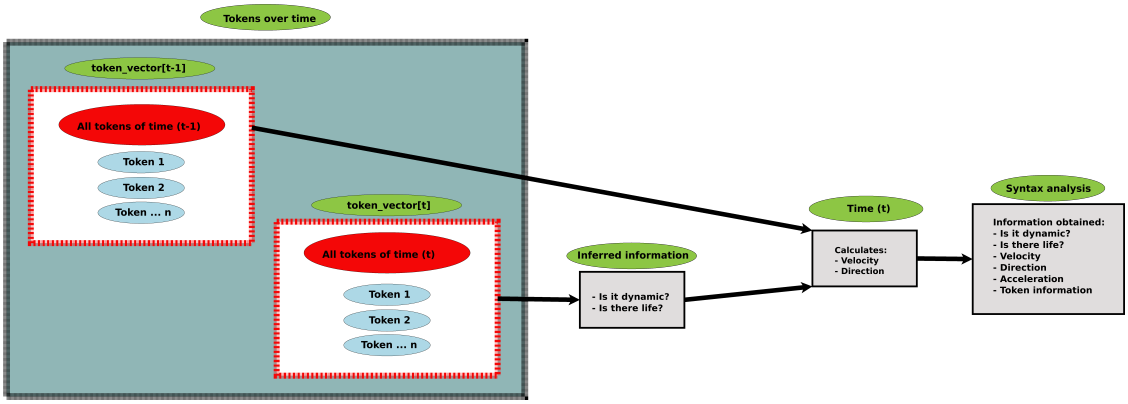


Figure 6. Overview of syntax analysis.

Table 1. Inferred knowledge in objects.

	Object Class
Is it dynamic?	person, bicycle, car, motorbike, aeroplane, bus, train, truck, boat, bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe, frisbee, snowboard, sports ball, skateboard, surfboard tennis racket, chair
Is there life?	person, bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe

To identify the same token at different time periods, a comparison between the data from a previous($t-1$) moment and the current one(t) is necessary. The compared information are the token's class (*class*), the peak of the histogram (*peak_histogram*) and Euclidean distance between the central points (*p_center*). First, they are compared in different moments (t and $t-1$) to verify if they belong to the same class. The second step is to use the histogram peak, where the objects are separated by gray tones, so a light grayscale dynamic object in the $t-1$ is compared only with the light grayscale dynamic object at the instant t. Finally, the Euclidean distance is used as the last criterion, considering that the tokens are generated at a small-time difference, something between 0.1 and 0.5 s, and a Euclidean distance of 0.7 m is considered as the maximum possible value for the displacement of an dynamic object of this time interval. This way, if two objects have the same class, the same shade of gray, and separated by a valid (small) value of a Euclidean distance between them, then they are the same object.

Algorithm 2 presents the procedure to identify the same object at two different instances of time. There are two vector inputs to the algorithm, where the first one contains all the present-instance tokens saved, while the other has all the past-instance tokens, the vectors are called $token\_vector[t].[token\_id]$ and $token\_vector[t-1].[token\_id]$, where t refers to the present instance time and t $-$ 1 to the past instance. The time difference depends on the processing capacity of the sensor, ranging from 0.1 to 0.5 s. In the algorithm processing, a comparison between all objects of both vectors is made. Where it is verified, if both objects are dynamic, then if they belong to the same class; next, if the difference between the histogram peak is less than 30, and, finally, if the distance between their centers is less than or equal to 70 cm. If all the requirements are met, the position of the two tokens is saved in a third vector, called $same\_token[position].[id\_(t), id\_(t-1)]$. In the future, these positions will be used to get the information of a Token to calculate the direction, velocity, and acceleration.

Algorithm 2: Identify the same object in two instants of time.

Input: Vector with all tokens at time t. $token\_vector[t].[token\_id]$.
Input: Vector with all tokens at time $t-1$. $token\_vector[t-1].[token\_id]$
Output: Vector with the id of the same token in the two instants of time.
$same\_token[position].[id\_(t), id\_(t-1)]$

```
/* Acceptable histogram difference */
```
$difference\_histogram\_peak = 30$
```
/* Acceptable euclidean distance */
```
$euclidean\_difference = 0.7$

$counter = 0$

```
/* Traverses the vector containing all the tokens of the present time (t) */
```
for $i \leftarrow (0:1:size(token\_vector[t])$ **do**
  `/* Traverses the vector containing all the tokens of the past time (t-1) */`
  for $j \leftarrow (0:1:size(token\_vector[t-1])$ **do**
    `/* Checks whether both objects are dynamic */`
    if $(token\_vector[t].[i].isDynamic == True)$ *and*
     $(token\_vector[t-1].[i].isDynamic == True)$ **then**
      `/* Checks whether both objects belong to the same class */`
      if $token\_vector[t].[i].class == token\_vector[t-1].[i].class$ **then**
        `/* Check the difference of the histogram peak */`
        if $(token\_vector[t].[i].peak\_histogram - token\_vector[t-1].[i].peak\_histogram) <= difference\_histogram\_peak$ **then**
          `/* Checks the difference of the Euclidean distance */`
          if $calculate\_euclidean\_distance(token\_vector[t].[i].p\_center, token\_vector[t-1].[i].p\_center) < euclidean\_difference$ **then**
            `/* Assigns to vector same_token the index values of the same object in time (t) and time (t-1) */`
            $same\_token[counter].id\_(t) == i$
            $same\_token[counter].id\_(t-1) == j$
            $counter = counter + 1$
          end
        end
      end
    end
  end
end
```
/* Returns the vector containing the index of the combined object in the two
   times.  */
```
return $same\_token[position].[id\_(t), id\_(t-1)]$

The direction of a dynamic object can be computed by the function atan2 that provides the angular difference between the two points [44]. Then, an angular deviation of 180 degrees is added to this difference, generating the course of the dynamic object. Algorithm 3 presents the complete computation of object direction. As input to the algorithm, there are the two vectors with all tokens saved, in the present instance (t) and in the past instance ($t-1$), with the respective names $token\_vector[t].[token\_id]$ and $token\_vector[t-1].[token\_id]$. The third input concerns the output of Algorithm 2, where the identification of the same tokens is stored, the name of this entry is $same\_token[position].[id\_(t), id\_(t-1)]$. The processing consists in obtaining the angular difference

between the two points representing the same token at different times so that it is possible to obtain the direction of the token. As output, a variable called as is created where the dynamic token direction, velocity, and acceleration data are stored. In this algorithm, the direction is saved in the $Syntax\_Analysis[id].direction$ field.

Algorithm 3: Calculate the direction of the dynamic object.

Input: Vector with all tokens at time t. $token\_vector[t].[token\_id]$.
Input: Vector with all tokens at time $t-1$. $token\_vector[t-1].[token\_id]$
Input: Vector containing the index of the combined object in the two times, $same\_token[position].[id\_(t), id\_(t-1)]$
`/* Traverses the vector containing the index of the combined object in the two times */`
for $i \leftarrow (0:1:size(same\_token)$ **do**
  `/* Difference in the x-axis between the two points */`
  $delta\_x \leftarrow token\_vector[t].[same\_token[i].id\_(t)].p\_center.x - token\_vector[t-1].[same\_token[i].id\_(t)].p\_center.x$
  `/* Difference in the y-axis between the two points */`
  $delta\_y \leftarrow token\_vector[t].[same\_token[i].id\_(t)].p\_center.y - token\_vector[t1].[same\_token[i].id\_(t)].p\_center.y$
  `/* Calculate the angle between the two points */`
  $angle\_between \leftarrow antan2(delta\_y, delta\_x)$
  `/* Adds 180 degrees to the angle, and saves in the variable responsible for the Syntax Analysis data. */`
  $Syntax\_Analysis[same\_token[i].id\_(t)].direction \leftarrow angle\_between + 3.14159$
end
return $Syntax\_Analysis$

The ratio between the variation of position and the variation of time is used to calculate the velocity of the dynamic object at time t. This feature is shown in meters, and the time in seconds, so the speed obtained is in meters per second (m/s). The Algorithm 4 presents the method applied to all the dynamic points identified by the Algorithm 2. The inputs to the algorithm are the same as those in Algorithm 3, while their output is the addition of speed of the moving token, called $Syntax\_Analysis[id].direction$. The algorithm processing consists of taking the time difference between the two tokens, the Euclidean distance traveled, and finally calculating the velocity by dividing the distance by time.

The acceleration is calculated using the speed variation divided by time. To calculate the acceleration, it is necessary that the object in question already has speed. Therefore, the velocity is first calculated, and, in the next cycle, its acceleration is calculated. Algorithm 5 brings the acceleration calculation, where the entries are the same as in Algorithms 3 and 4. The processing consists of first checking if the tokens are assigned any speed in the $Syntax\_Analysis[id].velocity$ field. Thus, the difference between the velocities of the tokens is calculated, and the time difference is obtained. Finally, acceleration is calculated by dividing the velocity variation by the time variation and saved in the $Syntax\_Analysis[id\_(t)].acceleration$ variable.

Algorithm 4: Calculate the velocity of the dynamic object.

Input: Vector with all tokens at time t. $token\_vector[t].[token\_id]$.
Input: Vector with all tokens at time $t-1$. $token\_vector[t-1].[token\_id]$
Input: Vector containing the index of the combined object in the two times, $same\_token[position].[id\_(t), id\_(t-1)]$
`/* Traverses the vector containing the index of the combined object in the two times */`
for $i \leftarrow (0:1:size(same\_token)$ **do**
  `/* Time difference between the two tokens */`
  $delta\_time \leftarrow$ $token\_vector[t].[same\_token[i].id\_(t)].time - token\_vector[t-1].[same\_token[i].id\_(t)].time$
  `/* Distance traveled by the object */`
  $delta\_distance \leftarrow euclidean\_distance(token\_vector[t][same\_token[i].id\_(t)].p\_center,$ $token\_vector[t-1].[same\_token[i].id\_(t-1)].p\_center)$
  `/* Calculate the velocity */`
  $velocity \leftarrow delta\_distance/delta\_time$
  `/* Save the object's velocity in the variable responsible for the Syntax Analysis data */`
  $Syntax\_Analysis[same\_token[i].id\_(t)].velocity \leftarrow velocity$
end
return *Syntax_Analysis*

Algorithm 5: Calculate the acceleration of the dynamic object.

Input: Vector with all tokens at time t. $token\_vector[t].[token\_id]$.
Input: Vector with all tokens at time $t-1$. $token\_vector[t-1].[token\_id]$
Input: Vector containing the index of the combined object in the two times, $same\_token[position].[id\_(t), id\_(t-1)]$
`/* Traverses the vector containing the index of the combined object in the two times */`
for $i \leftarrow (0:1:size(same\_token)$ **do**
  `/* Checking whether the dynamic object has speeds */`
  if $token\_vector[t].[same\_token[i].id\_(t)].velocity! = 0$ *and* $token\_vector[t-1].[same\_token[i].id\_(t-1)].velocity! = 0$ **then**
    `/* Time difference between the two tokens */`
    $delta\_time \leftarrow token\_vector[t].[same\_token[i].id\_(t)].time - token\_vector[t-1].[same\_token[i].id\_(t)].time$
    `/* velocity difference between the two tokens */`
    $delta\_velocity \leftarrow token\_vector[t].[same\_token[i].id\_(t)].velocity - token\_vector[t-1].[same\_token[i].id\_(t)].velocity$ `/* Calculate the acceleration */`
    $acceleration \leftarrow delta\_velocity/delta\_time$
    `/* Save the object's acceleration in the variable responsible for the Syntax Analysis data */`
    $Syntax\_Analysis[same\_token[i].id\_(t)].acceleration \leftarrow acceleration$
  end
end
return *Syntax_Analysis*

At the end of the syntax analysis, there is some new information of the token—for instance, stating if the object is dynamic or not, and whether it has a life or not (Table 1). The is also the calculation of the object's direction (Algorithm 3), velocity (Algorithm 4) and acceleration (Algorithm 5). The next step consists of a strategy of anticipation, where the future position of the object will be calculated, based on speed and acceleration, to take preventive actions.

3.3. Anticipation Analysis

The future position of dynamic objects is estimated through the data obtained in syntax analysis, in the next SD2I step, called anticipation analysis. The prediction of future behavior of time-varying elements is primordial to the robot attaining reliable path planning [45]. Moreover, for safety reasons, unnecessary calculations must be prevented during trajectory planning or maneuvers to avoid collisions with objects that are moving toward the robot. Figure 7 presents a diagram of the proposed strategy.

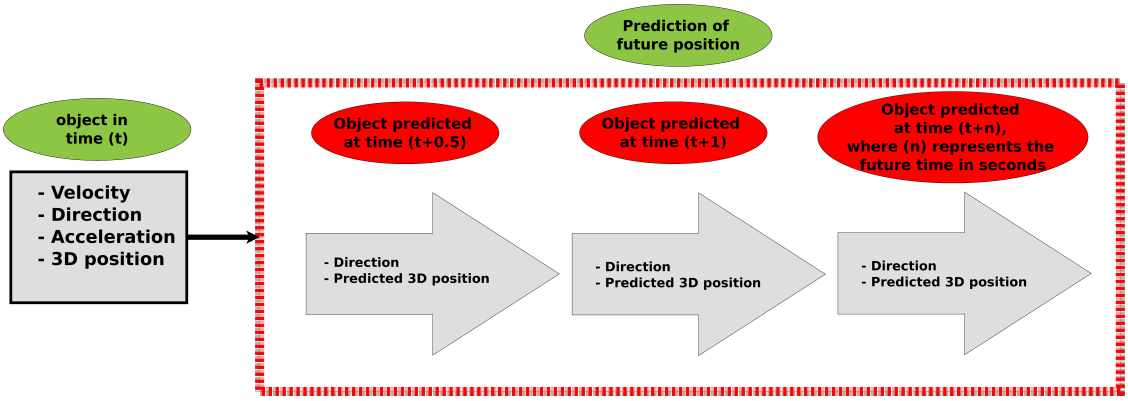


Figure 7. Diagram of the anticipation strategy.

The object motion estimation is based on acceleration, speed, and direction of the object. It is always assumed that the element maintains its direction. The Intelligent 3D Perception is designed to predict behavior at two future times: 0.5 and 1 s ahead. Equation (4) presents the calculation used to define the object displacement in time, where *time* must be replaced by the time to be predicted. The variable *Syntax_Analysis* possesses the direction, velocity, and acceleration of the object:

$$displacement = Syntax\_Analysis.velocity * time + \frac{Syntax\_Analysis.acceleration * time^2}{2}. \quad (4)$$

The predicted position is calculated by adding the displacement to the current object position, as shown in Equation (5). This process is carried out for two different time frames, being $time = 0.5$ and $time = 1$:

$$\begin{aligned} future\_position &= current\_position + displacement \\ future\_direction &= current\_direction. \end{aligned} \quad (5)$$

The next section will present the experiments and results. An explanation about the user interface detailing the displayed information is also presented.

4. Experiments and Results

The data provided by the proposed perception source are published in the format of ROS topics [41], where the robot can access them in an agile and easy way. This information allows the robot to make decisions and take action in navigation time. The robot accessing all the information of each Token is assured. The available information is: class, probability, center of the object in the

image, center of the object in 3D, width in pixels, height in pixels, histogram peak, histogram sum, speed, direction, acceleration, whether the object is dynamic and if the object has life or not. This information is returned directly to the robot, without the need for any additional processing.

To facilitate information understanding, a graphic strategy is developed. Data can also be displayed in graphical format by using Interactive Markers [46]. This way, it is possible to view the objects in a 3D environment; additionally clicking on the object will display all collected information related to it.

A first experiment in which the perception source was pointed at a shelf, as in Figure 8, is carried out to validate the Lexical Analysis. Figure shows the RGB image output added to YOLOv3, (YOLOv3 output). The output of the perception source *Intelligent 3D Perception* (I3P) is presented in two ways: a top view and a frontal view. In these views, each object position is checked against the source of perception (Algorithm 1). The objects are represented by colored spheres, where yellow represents the laptop, red the sofa, green the books, orange the bottles and blue the suitcase. The source of perception I3E is represented by a red robot. In the figure, the Point Cloud with color (RGB-D data) is also displayed. All information obtained from each object can be viewed by clicking on its respective sphere. The image *I3P output, (information display)* provides an example, where all information obtained from the book object on the right is displayed. The same can be done with any identified object.

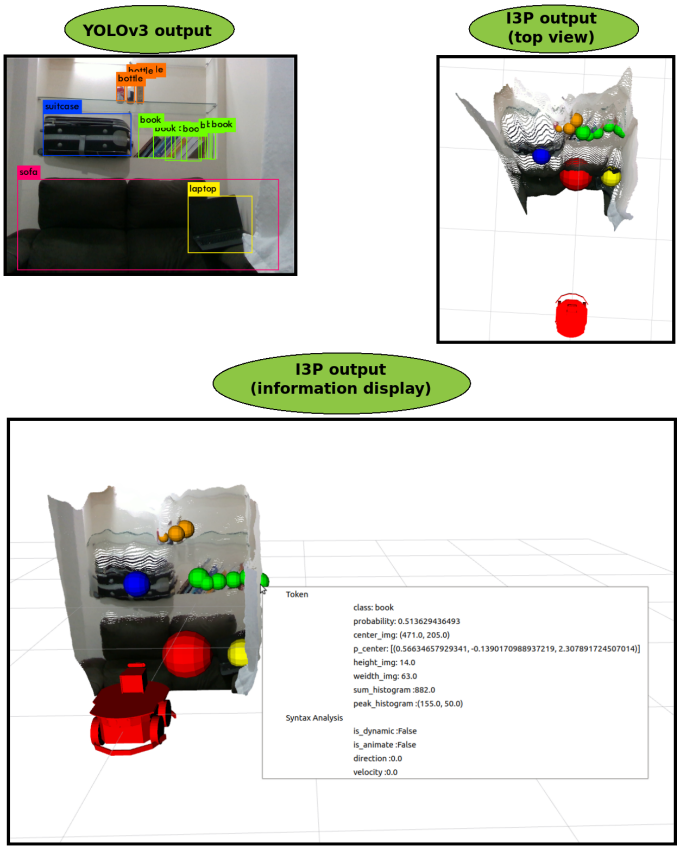


Figure 8. Result of the application of the Lexical Analysis.

The Syntax Analysis is concerned with obtaining information from data provided by the lexical analysis. First, the object behavior is inferred. If it is a dynamic and or animated object (Table 1), then all the Tokens are saved in two instants of different times and the procedure to identify the same object in these two instants is applied (Algorithm 2). Once this is done, it is possible to calculate its direction, velocity, and acceleration based on the displacement of the object (Algorithms 3–5). Figure 9 shows the result of the syntax analysis, when two people are moving in front of the robot. As soon as the dynamic object is identified by I3P, it is converted to a vector, which points to the direction in which the object is moving. The figure shows the path developed by the object (pink trail) so that it is possible to infer its direction, as well as validate the proposed strategy with multiple dynamic objects of the same class, one of the situations with the highest degree of complexity. In this Figure, the refrigerator is represented by a red sphere, the dining table by a light green sphere, the chair by a blue sphere, the bottle by an orange sphere, and person is represented in pink.

The Anticipation Analysis consists of predicting the future position of the moving object by its direction, speed, and acceleration. Figure 10 shows the working strategy, where the purple vector represents the future position of the object in 1 s. This time frame can be changed to identify the future position of the object; the Equations (4) and (5) are used. A second visual strategy has been developed where a black sphere is created around the perception source if the future position of a moving object may come crashing into the robot.

The proposed strategy was implemented and proved to be satisfactory for use in mobile robots. A specific sensor to provide intelligent information slows down the development of intelligent robotic systems and prevents accidents if the information provided by the sensor is used correctly, fulfilling the main objective of this paper.

Accuracy and Precision

The error of the proposed strategy of lexical analysis corresponds to the object identification error of YOLOv3 plus the distance error provided by the depth sensor. The YOLOv3 has an accuracy of 35.4 mAP (mean Average Precision). The position error of the 3D object refers to the sensor error and may vary according to the employed perception source. The paper [47] presents the data from the RealSense D435 3D sensor, which was used by this work, which shows an error of 0.707 mm for an object at 1 m. Based on the data provided by the work, an object identified at 1 m by the strategy proposed by this paper would have an accuracy of 35.4 mAP on the object identification and an error of 0.707 mm on its position.

One of the problems that can occur when calculating velocity and acceleration is not having the same object in two moments. This issue is caused by YOLOv3 not identifying the object in a specific image frame, or the proposed strategy not recognizing the same dynamic object. Ten experiments were performed to quantify the accuracy, where a dynamic object moved around nonstop within the sensor range. During the experiment period, all Tokens data were saved. Tokens that have velocity, direction, and acceleration data are considered valid. That did not track the same objected are considered stopped were defined as errors for obtaining I3P sensor data. The results are presented in Table 2, containing the duration of each experiment, the amount of total tokens captured, the calculated tokens per second, the number of tokens with velocity, direction, and acceleration data, number of tokens with calculation errors as well as the percentage of the error. The I3P sensor error in the experiments was 4.835% on the average.

Anticipation Analysis is valid for different tasks, such as high preservation by avoiding collision with an object moving towards the robot, or for trajectory calculation. A dynamic object was taken towards the robot in a total of 10 experiments, aiming to validate the analysis in a quantitative way. Data were collected from their actual position, and position established by the Anticipation Analysis, and then the Euclidean distance from both locations to the robot was performed. An average of the variation of the two places was shown, where the obtained result was of 84 cm. On average, the robot was able to anticipate an object's movement towards it from up to 84.79 cm. The smallest distance

difference between the tests was 34.43 cm and the largest 2.77 m. This distance varies with the speed of the object.

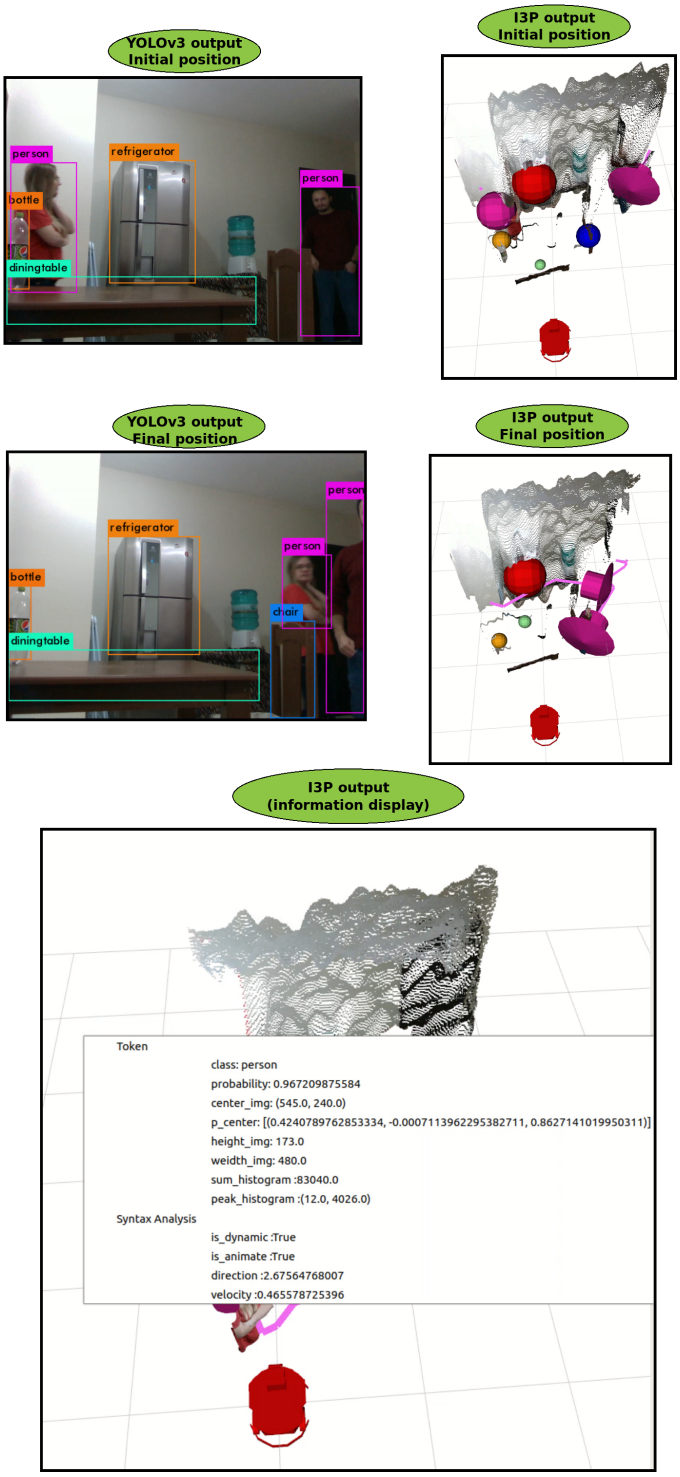


Figure 9. Result of the application of the Syntax Analysis.

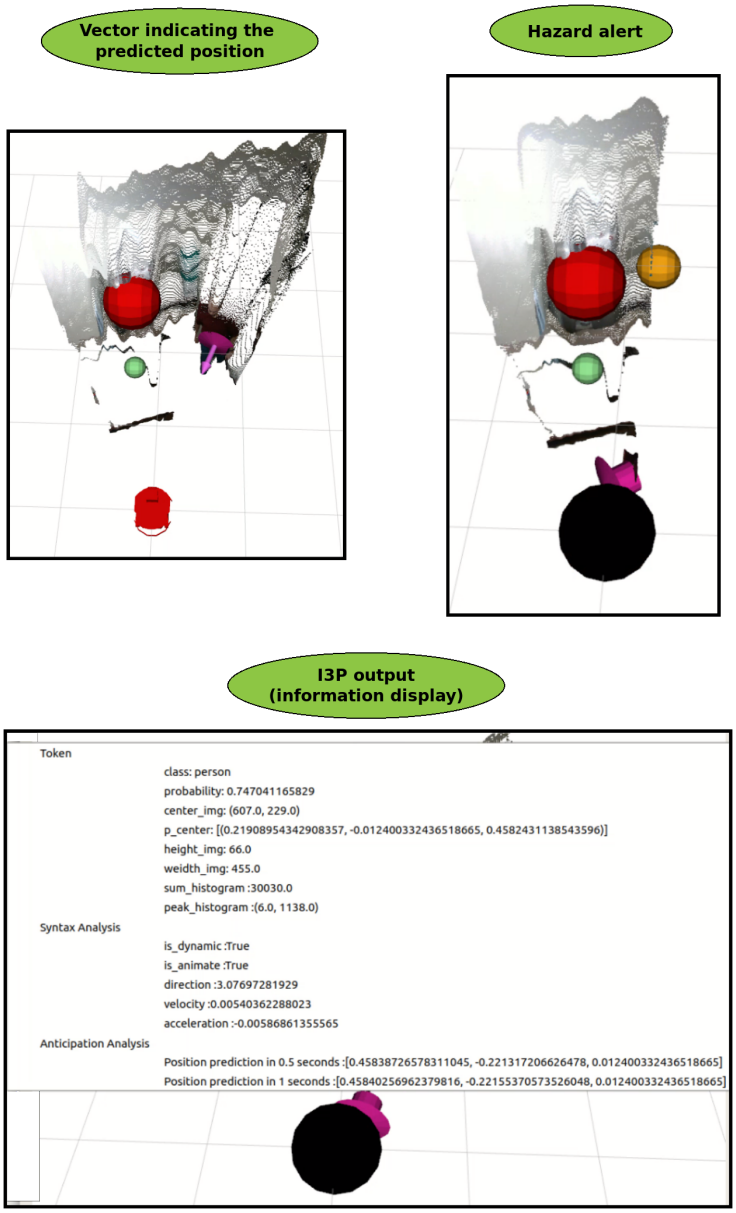


Figure 10. Result of the application of the Anticipation Analysis.

Table 2. Errors in the accuracy experiments.

	Time	Tokens	Tokens Per Second	Tokens with Speed/Direction/Acceleration	Tokens with Calculation Error	Error Percentage
	32.19	100	3.107	94	6	6.000
	46.37	247	5.327	235	12	4.858
	38.74	128	3.304	122	6	4.688
	69.58	227	3.262	215	12	5.286
	35.18	111	3.155	110	1	0.901
	58.44	185	3.166	180	5	2.703
	51.98	233	4.482	221	12	5.150
	76.31	355	4.652	311	44	12.394
	40.39	130	3.219	126	4	3.077
	47.96	152	3.169	147	5	3.289
Average	**49.71**	**186.8**	**3.684**	**186.8**	**10.7**	**4.835**

5. Conclusions

The paper has presented a novel source of perception, capable of identifying objects in 3D, and extracting as much information as possible, such as name, position in the real world in relation to the source of perception, predominant gray tone, speed, direction and acceleration, in addition to predicting its future position. The proposed approach is carried out in three steps. A *Lexical Analysis* was performed, where the RGB image from the sensor was used to identify the objects through YoLoV3. From the image, data such as the class of the object, center of the object in the image, Box around the object, probability of hit and histogram peak were identified. Moreover, the Lexical Analysis also implements the computation of the object position in the real world, from its position in the image, using the cloud of points provided by the depth sensor. All these data were stored in a Token. Each Token contains information about each object identified in the scene. The error of the sensor is highly dependent on YOLOv3 accuracy, which is 35.4 mean average precision. The 3D position error from the sensor, which is specified by the manufacturer by up to 2%.

The second step consisted of a *Syntax Analysis*, where two Tokens were obtained at different time instants, called (t) for the current instant, and (t-1) for the previous instant. A procedure was then developed to identify the same Token at both instants of time. Having the same Token in both instances of time, it is possible to calculate object properties such as speed, direction, and acceleration, based on the known data of each Token. This procedure is validated through the analysis of error from obtaining the velocity, direction, and acceleration data of the dynamic objects by the strategy proposed by this paper resulting in an absolute error of 4.835%.

Finally, with velocity, direction, and acceleration, an *Anticipation Analysis* is developed, where the position of the moving object at two future instants of time was predicted. These data are important for a mobile robot in order to avoid future collisions with a moving object. In the experiments performed, it was possible to verify that an object towards the robot with an average distance of 84.79 cm, having particular cases of identifying at a distance of up to 2.77 m, makes this sensor better suited to provide information for avoidance behaviors.

At the end of these three analyses, the proposed perception source provided the robot with the ability to identify objects, know their position in the world, calculate their speed and acceleration, predict their future position and present the obtained data in an intelligent way, alerting the robot and the viewer about collision hazards.

The sensor has some particular issues to be improved, such as the instantaneous lack of calculation the speed, direction, and acceleration of dynamic objects in single frames. This problem does not pose a risk to the robot because a frame lasts, on average, 0.3367 s, and the error happens in isolated cases in very tiny time intervals. As future efforts, it is possible to refine the strategy to work with a more significant amount of time intervals, thus preventing such errors from occurring.

The use of an intelligent sensor can prevent the loss of material and processing resources. By predicting the future position of a dynamic object, it is possible to develop a high preservation strategy to avoid collisions, for example. Focusing on processing and performance of the robot, it is possible to avoid the recalculation of trajectory, considering busy positions where dynamic objects have a chance to occupy. These are just a few examples of how the proposed strategy can contribute to the advances of research and development in mobile robotics.

Author Contributions: Conceptualization, M.A.S.T., A.S.d.O., L.V.R.d.A., D.R.P. and J.E.R.; Methodology, M.A.S.T., A.S.d.O., L.V.R.d.A., F.N.-J., J.E.R. and D.R.P.; Validation, N.D., H.B.S., R.D.C.M.N., D.R.P. and J.E.R.; Resources, L.V.R.d.A., F.N.-J., D.R.P. and J.E.R.; Funding acquisition, L.V.R.d.A., F.N.-J., D.R.P., J.E.R. and A.S.d.O.; Writing—original draft preparation, M.A.S.T. and A.S.d.O.; Writing—review and editing, R.D.C.M.N., L.V.R.d.A., F.N.-J., A.S.d.O., D.R.P. and J.E.R.; all of the authors have participated in revising the intellectual content of this article.

Acknowledgments: The authors thank the National Counsel of Technological and Scientific Development of Brazil (CNPq), Coordination for the Improvement of Higher Level People (CAPES) and the Brazilian Ministry of Science, Technology, Innovation and Communication (MCTIC). The authors would like to greatly thank NVIDIA Corporation (Santa Clara, CA, USA) for the donation of equipment and PETROBRAS for project funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Siegwart, R.; Nourbakhsh, I.R.; Scaramuzza, D.; Arkin, R.C. *Introduction to Autonomous Mobile Robots*; MIT Press: Cambridge, MA, USA, 2011.
2. Jones, J.L.; Seiger, B.A.; Flynn, A.M. *Mobile Robots: Inspiration to Implementation*; AK Peters/CRC Press: Boca Raton, FL, USA, 1998.
3. Lu, Y. Industry 4.0: A survey on technologies, applications and open research issues. *J. Ind. Inf. Integr.* **2017**, *6*, 1–10.
4. Bahrin, M.A.K.; Othman, M.F.; Azli, N.N.; Talib, M.F. Industry 4.0: A review on industrial automation and robotic. *J. Teknol.* **2016**, *78*, 137–143.
5. Benotsmane, R.; Kovács, G.; Dudás, L. Economic, Social Impacts and Operation of Smart Factories in Industry 4.0 Focusing on Simulation and Artificial Intelligence of Collaborating Robots. *Soc. Sci.* **2019**, *8*, 143. doi:10.3390/socsci8050143.
6. Cardona, G.A.; Calderon, J.M. Robot Swarm Navigation and Victim Detection Using Rendezvous Consensus in Search and Rescue Operations. *Appl. Sci.* **2019**, *9*, 1702. doi:10.3390/app9081702.
7. Murphy, R.R. Human-robot interaction in rescue robotics. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **2004**, *34*, 138–153.
8. Davis, M.; Sahin, F. HOG feature human detection system. In Proceedings of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Budapest, Hungary, 9–16 October 2016; pp. 002878–002883.
9. Everett, H. *Sensors for Mobile Robots*; AK Peters/CRC Press: Boca Raton, FL, USA, 1995.
10. Kehoe, B.; Patil, S.; Abbeel, P.; Goldberg, K. A survey of research on cloud robotics and automation. *IEEE Trans. Autom. Sci. Eng.* **2015**, *12*, 398–409.
11. Fiala, M. ARTag, a fiducial marker system using digital techniques. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 2, pp. 590–596.
12. Fiala, M. Designing highly reliable fiducial markers. *IEEE Trans. Pattern Anal. Mach. Intell.* **2009**, *32*, 1317–1324.
13. Cui, J.; Zhang, J.; Sun, G.; Zheng, B. Extraction and Research of Crop Feature Points Based on Computer Vision. *Sensors* **2019**, *19*, 2553. doi:10.3390/s19112553.
14. Tan, D.S.; Lin, J.M.; Lai, Y.C.; Ilao, J.; Hua, K.L. Depth Map Upsampling via Multi-Modal Generative Adversarial Network. *Sensors* **2019**, *19*, 1587. doi:10.3390/s19071587.
15. Tan, D.S.; Yao, C.Y.; Ruiz, C.; Hua, K.L. Single-Image Depth Inference Using Generative Adversarial Networks. *Sensors* **2019**, *19*, 1708. doi:10.3390/s19071708.
16. Teixeira, M.A.S.; Santos, H.B.; de Oliveira, A.S.; Arruda, L.V.; Neves, F. Robots Perception Through 3D Point Cloud Sensors. In *Robot Operating System (ROS)*; Springer: New York, NY, USA, 2017; pp. 525–561.
17. Jokela, M.; Kutila, M.; Pyykönen, P. Testing and Validation of Automotive Point-Cloud Sensors in Adverse Weather Conditions. *Appl. Sci.* **2019**, *9*, 2341. doi:10.3390/app9112341.
18. Xu, H.; Chen, G.; Wang, Z.; Sun, L.; Su, F. RGB-D-Based Pose Estimation of Workpieces with Semantic Segmentation and Point Cloud Registration. *Sensors* **2019**, *19*, 1873. doi:10.3390/s19081873.
19. Yen, S.H.; Tang, P.C.; Lin, Y.C.; Lin, C.Y. Development of a Virtual Force Sensor for a Low-Cost Collaborative Robot and Applications to Safety Control. *Sensors* **2019**, *19*, 2603. doi:10.3390/s19112603.
20. Shin, M.; Paik, W.; Kim, B.; Hwang, S. An IoT Platform with Monitoring Robot Applying CNN-Based Context-Aware Learning. *Sensors* **2019**, *19*, 2525. doi:10.3390/s19112525.
21. He, W.; Li, Z.; Chen, C.P. A survey of human-centered intelligent robots: issues and challenges. *IEEE/CAA J. Autom. Sin.* **2017**, *4*, 602–609.
22. Rusu, R.B.; Cousins, S. Point cloud library (pcl). In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1–4.
23. Pomerleau, F.; Colas, F.; Siegwart, R.; others. A review of point cloud registration algorithms for mobile robotics. *Found. Trends Robot.* **2015**, *4*, 1–104.
24. Viola, P.; Jones, M.J. Robust real-time face detection. *Int. J. Comput. Vis.* **2004**, *57*, 137–154.

25. Liu, C.; Chang, F.; Chen, Z.; Liu, D. Fast traffic sign recognition via high-contrast region extraction and extended sparse representation. *IEEE Trans. Intell. Transp. Syst.* **2015**, *17*, 79–92.
26. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the International Conference on Computer Vision & Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 886–893.
27. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; IEEE: New York, NY, USA, 2014; pp. 580–587.
28. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Washington, DC, USA, 7–13 December 2015; pp. 1440–1448.
29. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv* **2015**, arXiv:1506.01497v3.
30. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: New York, NY, USA, 2016; pp. 21–37.
31. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
32. Min, W.; Cui, H.; Han, Q.; Zou, F. A Scene Recognition and Semantic Analysis Approach to Unhealthy Sitting Posture Detection during Screen-Reading. *Sensors* **2018**, *18*, 3119. doi:10.3390/s18093119.
33. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
34. Keselman, L.; Woodfill, J.I.; Grunnet-Jepsen, A.; Bhowmik, A. Intel (r) realsense (tm) stereoscopic depth cameras. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017; pp. 1267–1276.
35. Pagliari, D.; Pinto, L. Calibration of kinect for xbox one and comparison between the two generations of microsoft sensors. *Sensors* **2015**, *15*, 27569–27589.
36. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: New York, NY, USA, 2014; pp. 740–755.
37. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338.
38. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The kitti vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361.
39. Nvidia. AUTONOMOUS MACHINES. Available online: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/ (accessed on 22 June 2019).
40. Intel. INTEL® NUC. Available online: https://www.intel.com/content/www/us/en/products/boards-kits/nuc.html (accessed on 22 June 2019).
41. Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An open-source Robot Operating System. In Proceedings of the IEEE International Conference on Robotics and Automation: Workshop on Open Source Software (ICRA), Kobe, Japan, 12–17 May 2009.
42. Kapur, J.N.; Sahoo, P.K.; Wong, A.K. A new method for gray-level picture thresholding using the entropy of the histogram. *Comput. Vis. Graph. Image Process.* **1985**, *29*, 273–285.
43. Macedo, J.; Marques, L.; Costa, E. A Comparative Study of Bio-Inspired Odour Source Localisation Strategies from the State-Action Perspective. *Sensors* **2019**, *19*, 2231.
44. Luo, Z.; Ding, J.; Zhao, L.; Wu, M. An Enhanced Non-Coherent Pre-Filter Design for Tracking Error Estimation in GNSS Receivers. *Sensors* **2017**, *17*, 2668. doi:10.3390/s17112668.
45. Teixeira, M.A.S.; Dalmedico, N.; de Oliveira, A.S.; de Arruda, L.V.R.; Neves-Jr, F. A pose prediction approach to mobile objects in 2D costmaps. In Proceedings of the 2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR), Curitiba, Brazil, 8–11 November 2017; IEEE: Piscatway, NJ, USA, 2017; pp. 1–6, doi:10.1109/SBR-LARS-R.2017.8215314.

46. Gossow, D.; Leeper, A.; Hershberger, D.; Ciocarlie, M. Interactive markers: 3-d user interfaces for ros applications [ros topics]. *IEEE Robot. Autom. Mag.* **2011**, *18*, 14–15.
47. Carfagni, M.; Furferi, R.; Governi, L.; Santarelli, C.; Servi, M.; Uccheddu, F.; Volpe, Y. Metrological and Critical Characterization of the Intel D415 Stereo Depth Camera. *Sensors* **2019**, *19*, 489. doi:10.3390/s19030489.

6 DEEPSPATIAL: INTELLIGENT SPATIAL SENSOR TO PERCEPTION OF THINGS

This chapter presents the paper published in the *IEEE Sensors* journal (ISSN: 1558-1748). The data of the paper is shown in Table 4.

Table 4 – Data of the paper *DeepSpatial: Intelligent Spatial Sensor to Perception of Things.*

Authors	TEIXEIRA, Marco Antonio Simoes; NEVES-, FLAVIO ; KOUBAA, ANIS; DE ARRUDA, LUCIA VALERIA RAMOS ; DE OLIVEIRA, ANDRE SCHNEIDER
Title	*DeepSpatial: Intelligent Spatial Sensor to Perception of Things*
Journal	*IEEE Sensors*
ISSN	1558-1748
DOI	10.1109/JSEN.2020.3035355
Publication date	04 November 2020

Source: Own authorship.

The paper presented in chapter 5 showed good results validating the proposed strategy but left some problems open that need to be improved to be coupled to the robot. It is possible to mention two main problems, egomotion, and the need to embark the entire approach in a compact device so that it can be attached to the robot.

Thus, this paper aims to solve the egomotion problem, create compact equipment capable of executing the proposed approach, couple the equipment to the robot, and perform real experiments. The proposed equipment consists of an Intel Nuc NUC5i5RYH minicomputer, Nvidia Jetson Nano and the Intel RealSense D435i and Intel RealSense Tracking Camera T265 sensors. The Intel Nuc NUC5i5RYH and Nvidia Jetson Nano equipment's main configurations can be seen in the Table 5.

Table 5 – Components used in the development of the paper *DeepSpatial: Intelligent Spatial Sensor to Perception of Things.*

Component	Description	Specification
Intel Nuc NUC5i5RYH		
CPU	Intel® Core™ i5-5250U	Cores: 2, Threads: 4, Frequency: 1.60 GHz
RAM memory	DDR3L-1600 1.35V SO-DIMM	8 gb
Nvidia Jetson Nano		
CPU	ARM® A57	Cores: 4, Frequency: 1.43 GHz
GPU	NVIDIA Maxwell	128-core
RAM memory	64-bit LPDDR4 25.6 GB/s	2 gb

Source: Own authorship.

This paper fulfills this thesis's last specific objective, validating the new approach proposed in real experiments. As a result, the approach was embedded in compact equipment, and

the equipment was coupled to the robot. The license for this article with permission for use in this thesis is in Appendix B.

3966 IEEE SENSORS JOURNAL, VOL. 21, NO. 4, FEBRUARY 15, 2021

DeepSpatial: Intelligent Spatial Sensor to Perception of Things

Marco Antonio Simões Teixeira, Flávio Neves-JR, Anis Koubaa, *Member, IEEE*, Lúcia Valéria Ramos de Arruda, *Member, IEEE*, and André Schneider de Oliveira, *Member, IEEE*

***Abstract*—This paper discusses a spatial sensor to identify and track objects in the environment. The sensor is composed of an RGB-D camera that provides point cloud and RGB images and an egomotion sensor able to identify its displacement in the environment. The proposed sensor also incorporates a data processing strategy developed by the authors to conferring to the sensor different skills. The adopted approach is based on four analysis steps: egomotive, lexical, syntax, and prediction analysis. As a result, the proposed sensor can identify objects in the environment, track these objects, calculate their direction, speed, and acceleration, and also predict their future positions. The on-line detector YOLO is used as a tool to identify objects, and its output is combined with the point cloud information to obtain the spatial location of each identified object. The sensor can operate with higher precision and a lower update rate, using YOLOv2, or with a higher update rate, and a smaller accuracy using YOLOv3-tiny. The object tracking, egomotion, and collision prediction skills are tested and validated using a mobile robot having a precise speed control. The presented results show that the proposed sensor (hardware + software) achieves a satisfactory accuracy and usage rate, powering its use to mobile robotic. This paper's contribution is developing an algorithm for identifying, tracking, and predicting the future position of objects embedded in a compact hardware. Thus, the contribution of this paper is to convert raw data from traditional sensors into useful information.**

***Index Terms*—Spatial sensor, egomotion, YOLO, mobile robot.**

I. Introduction

ADVANCESin sensing techniques and technologies have allowed the development of small and useful sensors capable of providing a large amount of data. A class of such sensors is the RGB-D type that provides spatial information about the environment. The distance information is linked to pixels of the image such that each pixel can be represented by its coordinates X, Y, Z in a Cartesian plane, relative to the center of the sensor. This operation generates a data point cloud.

Manuscript received October 2, 2020; accepted October 29, 2020. Date of publication November 4, 2020; date of current version January 15, 2021. This work was supported in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Brazil, under Grant 001. The associate editor coordinating the review of this article and approving it for publication was Dr. Amitava Chatterjee. *(Corresponding author: Marco Antonio Simões Teixeira.)*

Marco Antonio Simões Teixeira, Flávio Neves-JR, Lúcia Valéria Ramos de Arruda, and André Schneider de Oliveira are with the Graduate School of Electrical Engineering and Computer Science (CPGEI), Universidade Tecnológica Federal do Paraná (UTFPR), Curitiba 80230-901, Brazil (e-mail: marcoteixeira@alunos.utfpr.edu.br; neves@utfpr.edu.br; lvrarruda@utfpr.edu.br; andreoliveira@utfpr.edu.br).

Anis Koubaa is with the Robotics and Internet-of-Things Unit (RIOTU), Prince Sultan University (PSU), Riyadh 11586, Saudi Arabia, and also with the CISTER/INESC and ISEP-IPP, 4200-135 Porto, Portugal (e-mail: akoubaa@psu.edu.sa).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/JSEN.2020.3035355

These RGB-D sensors are used for a multitude of applications, such as facial recognition [1], [2], object measurement and classification in industries [3] among several other applications. Specially in mobile robotics, RGB-D sensors are usually used for navigation tasks and environment mapping [4], [5]. Moreover, they can also be used for robust tasks, such as object tracking [6], [7], and identification of the robot's position in the environment [8], [9]. However, the successful accomplishment of all these activities involves reliable data processing beyond the mere data capture by the RGB-D sensor.

Indeed, RGB-D sensors provide only distance data associated with an RGB image. This raw data is not enough to support any decision made by a robot, or to identify the person passing in front of the sensor, for example. Thus, It is necessary to process this data to generate useful information, such as an occupation map for mobile robots, or identify a person's face for a security system.

In general, computer vision techniques are applied to RGB images to extract such useful knowledge. For example, deep learning methods allow us to perform advanced tasks such as identification of objects or people, identification of anomalies, among many others [10]–[14]. One of the techniques that stands out in objects' recognition from RGB images is YOLO (You only look once) [15], [16]. This technique does not need robust hardware for online running and promotes a good

trade-off between processing power and precision of results. Besides, the used computer vision technique must also be able to deal with the point cloud data corresponding to the pixel coordinates generated by the RGB-D sensors [17]–[19].

In this paper, an embedded intelligent sensor, named *Intelligent Spatial Sensor to Perception of Things* (DeepSpatial sensor), is developed. The proposed sensor is an arrangement of different perception sources, which are merged to produce concise and reliable information. An RGB-D sensor is used to obtain a cloud of points containing the distance of the objects around it and also giving spatial notion of the environment. An egomotion sensor is used to identify sensor displacement and provide its linear and angular velocity. All pre-processing and merge-processing are carried out on an embedded CPU. The YOLO is used as a tool to object identification, and information matching for different perception sources, being processed over an embedded GPU that supports deep learning techniques.

The work [20] presents a comparison between CNN object detection techniques being performed in embedded systems, such as Jetson TX1, TX2, and Xavier. YoLo is the algorithm with the highest update rate in embedded systems, according to the authors. Thus, YoLo was chosen as an object detection tool. In the results section (sec::Performance), a comparison is presented between the different versions of YoLo running on Nvidia Jetson Nano. Our main contribution lies in integrating some well-known sensing hardware into a single perception system able to identify objects in the environment, track them, and predict their future positions.

Some works discuss similar approaches for sensing systems, also presenting solutions for objects tracking in the spatial environment using YOLO [21]–[24]. However, these works are not concerned with the technique's materialization into an embedded solution, with the identification and prediction of detected objects motion, and with the reference movement (i.e., the sensor motion by itself). As proposed in this paper, these techniques also use known deep learning tools for the development of a new sensing strategy. However, different from these cited works that only use computer vision techniques to track objects in the environment, the sensor herein proposed is capable of tracking objects using the Point Cloud and it also provides spatial object disposition based on the distance between objects in the environment, predicting the future positions of such objects.

There also are some papers concerned with the prediction of the objects' trajectory, as [25], [26]. Especially in [25], a time difference strategy similar to that presented in this work is used. However, in these works, the authors are not concerned with the acceleration and possible collision calculation between dynamic objects and the object being tracked, as it will be done in this paper.

Finally, our recent paper [27] presents a software strategy very similar to that developed herein. However, this recent paper is not worrying about the sensor displacement by itself in the environment and about the processing capacity or hardware requirements. In summary, this work aims to develop a perception system (in terms of hardware and software) using deep learning techniques as a tool for object identification and tracking, predicting their future position, and identifying the relative sensor displacement in the environment. The result will be a compact equipment capable of carrying out all the proposed actions.

II. DeepSpatial Sensor

This paper aims to develop an intelligent sensor to identify objects in the environment, track them, and predict their future positions. The sensor is called *Intelligent Spatial Sensor to Perception of Things* or simply DeepSpatial sensor. The DeepSpatial development is presented through two steps. First, the proposed sensor architecture and used hardware will be presented in section II-A. Thus the software procedures implementing the sensing intelligent approach is discussed in section II-B.

A. DeepSpatial Hardware

The proposed sensor hardware has four components. The first is a small computer implementing three tasks: data processing, information exchange with the user (a mobile robot in this paper), and Wireless network creation and management.

The chosen computer is the Intel Nuc NUC5i5RYH due to its processing power, small size, and low battery consumption. An Nvidia Jetson Nano board is the second component. It is used to run computer vision procedures for object identification in the environment (YoLo). This graphics processor is ideal for performing tasks in parallels, such as deep learning techniques and other Artificial Intelligence (AI) applications.

The other two hardware components are sensing elements. The Intel RealSense D435i sensor is used to perceive the environment. This component has an RGB camera to collect images of the environment and infrared sensors to obtain spatial information. This information is used to identify objects and their positions around the sensor. The Intel RealSense Tracking Camera T265 is used to capture the DeepSpatial displacement. This camera measures its movement allowing to infer information such as speed and travel direction of DeepSpatial, for example.

The communication between all components of DeepSpatial sensor is implemented through Ethernet and USB interfaces, as shown in Figure 1. A direct connection between the NUC computer and the Intel RealSense D435i via a USB 3.0 is established. The same occurs with the Intel RealSense Tracking Camera T265 sensor. The communication between Jetson Nano and the NUC computer is carried out via an Ethernet network. Finally, the NUC computer creates a wireless network allowing access to information from DeepSpatial sensor and communication with other equipment, such as the mobile robot.

The entire proposed strategy runs entirely on the DeepSpatial sensor. External equipment, such as a computer, can collect sensor data, but it is not necessary to employ it. This work aims to propose a novel embedded and independent equipment to spatial perception.

Finally, the integration and management of all hardware components is carried out through "Robot Operating System"

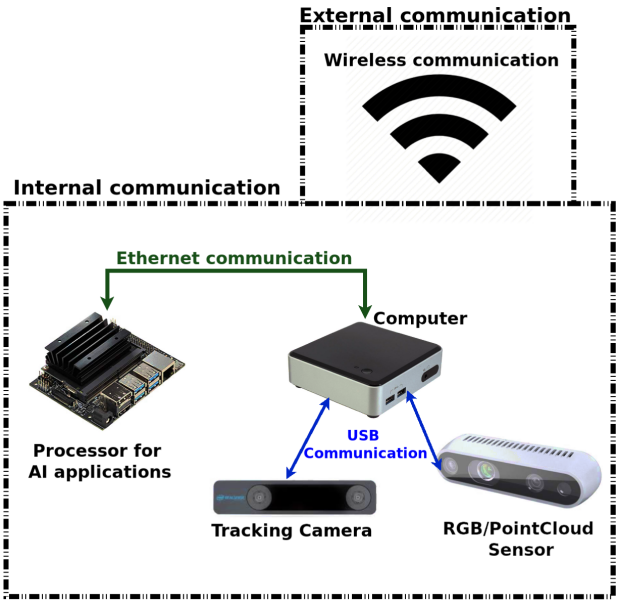


Fig. 1. Representation of communication between DeepSpatial sensor components. For AI processing, it is using an Nvidia Jetson Nano. The tracking sensor is an Intel RealSense Tracking Camera T265, and the RGB/Pointcloud sensor is an Intel RealSense D435i.

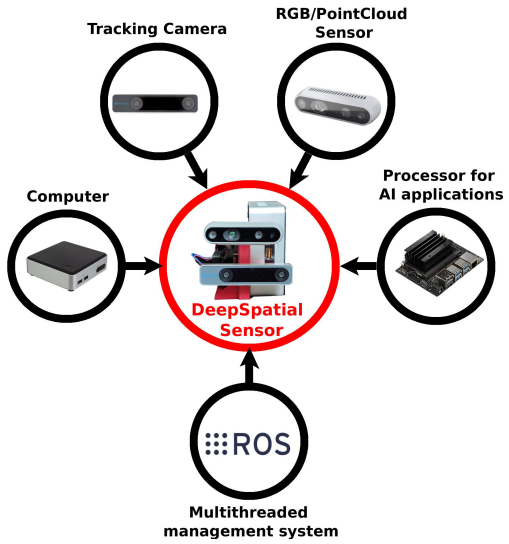


Fig. 2. All hardware components of the DeepSpatial sensor. For AI processing, it is using an Nvidia Jetson Nano. The tracking sensor is an Intel RealSense Tracking Camera T265, and the RGB/Pointcloud sensor is an Intel RealSense D435i.

(ROS) [28], [29] as can be seen in Figure 2. The center of this figure shows the DeepSpatial view, including all hardware components. It is worthwhile to note that the specific hardware components above cited are not mandatory. They can be replaced by any similar component with the same processing and sensing capacities and playing the same role.

B. DeepSpatial Software

The *Intelligent Spatial Sensor to Perception of Things* is able to identify objects in the environment, classify them into static and dynamic types, and track all of them. These tasks are accomplished through four analyzes: egomotion, lexical, syntax, and prediction analysis. The developed software strategy adopts a follow-up approach, where each analysis provides information to the next ones. However, it is also possible to collect all data and independently perform the analysis, according to the desired use of the DeepSpatial sensor. The software aspects will be presented and discussed in the following sections.

1) Egomotion Analysis: One of the main problems in the development of the DeepSpatial sensor is its spatial displacement. As the robot moves, errors can be generated in the calculation of the movement of the identified objects, since movement direction is based on the spatial displacement of the object concerning the sensor. In this way, a static object can have a misrepresentation of movement.

The Intel RealSense Tracking Camera T265 sensor is adopted to correct the influence of the sensor displacement. This equipment can support visual odometry techniques, like those presented in [30]–[32], which allow calculating the displacement of the sensor in the environment (egomotion). In this way, it is possible to obtain the linear speed (LVS) and angular speed (AVS) of the DeepSpatial sensor. This information will be used in the next Syntax analysis to compensate errors due to DeepSpatial displacement in the environment when the direction, speed, and acceleration of the identified objects are calculated.

2) Lexical Analysis: The lexical analysis aims to survey the characteristics of objects in the environment. The objects identified by the Lexical Analysis will be stored in tokens.The token's attributes are related to the object characteristics. Such attributes are object's class (C_t), probability (PRO_t), object's center in the image (C_{timg}), object's height in the image (H_{img}), object's width in the image (W_{img}), 3D position of the object in the real environment (P_t) and the object's life time (T_t).

- *Object's class and probability* These information are generated by YOLO, which can identify objects in an image. For each object detected by YOLO, a box is created around the object and an object name (class) is given. A degree of probability (probability) of the detected object belongs to the given class is also added. Both attributes (class and probability) are directly taken from YOLO. The image is collected by the D435i sensor, inputted to YOLO, processed and the YOLO answer is stored.
- *Center, height, and width of the object in the image* These data refer to the shape of the object identified in the image. This shape is computed from the object's box provided by YOLO. The number of pixels forming the height (H_{timg}) and the width (W_timg) of the object's box are computed, as also the position in the image of the box's center (C_timg). These measures correspond to the height and width of the object in pixels and to the position of the center of the object in the image. These data are used to extract future information and can be employed by the sensor user to carry other specific tasks.
- *3D position of the object* The position of the object in the real world is one of the most critical information to be

captured. The previous attributes (center of the object in the image, height and width in pixels) and the information from the Intel RealSense D435i sensor are used to identify the object's position in the world. D435i sensor provides a cloud of points where each pixel of the image has an associated spatial point, given by spatial coordinates [x, y, z] indicating the distance among the pixel and the center of the D435i sensor. Thus, each pixel [i, j] has a distance data x, y, z in the point cloud (PC [i, j, x, y, z]).

• *Life time* that is the time values when the object is detected by YOLO. Time is running continuously, starting at zero when the DeepSpatial is turned on.

The object's localization is estimated through an average of the points belonging to the box of the identified object. As YOLO does not perform segmentation of the object, the average is carried out only with 20% of the box's height and width pixels, thus only the center of the identified object is obtained, ensuring that the points used for the average of its 3D position are referring to the object. Equations [1,2,3,4] present the calculation used to obtain the 3D position of each object identified by the sensor, where the variable PC [i, j, x, y, z] refers to all 3D points identified by the RGB-D sensor, being i and j their dimensions in a 2D matrix, and (x, y, z) are the distance data. Moreover equation 5 computes the limits over the coordinate axes that are used in the summation. In this equation, CH_{start} and CH_{end} correspond to the height ranges used in the RGB-D data, and CW_{start} and CW_{end} limit the width ranges. These intervals correspond to 10% of the size of the width and height taken from the center of the identified object. Finally, the spatial position of the object is a 3D point (P_t) having the positions (x, y, z) of the identified object. To cut only the center of the box, its size is multiplied by 0.1 (10%) (Equations 23 and 4) of its height and width from the center. In this way, the box is converted to 20% of its total height, and 20% of its total width.

$$P_t = (P_t[x], P_t[y], P_t[z]) \tag{1}$$

where

$$P_t[x] = \frac{1}{CH_{end}} \frac{1}{CW_{end}} \sum_{i=CH_{start}}^{CH_{end}} \sum_{j=CW_{start}}^{CW_{end}} PC[i, j, x] \tag{2}$$

$$P_t[y] = \frac{1}{CH_{end}} \frac{1}{CW_{end}} \sum_{i=CH_{start}}^{CH_{end}} \sum_{j=CW_{start}}^{CW_{end}} PC[i, j, y] \tag{3}$$

$$P_t[z] = \frac{1}{CH_{end}} \frac{1}{CW_{end}} \sum_{i=CH_{start}}^{CH_{end}} \sum_{j=CW_{start}}^{CW_{end}} PC[i, j, z] \tag{4}$$

and

$$\begin{aligned} CH_{end} &= (C_{timg} + (H_{timg} * 0.1)) \\ CH_{start} &= (C_{timg} - (H_{timg} * 0.1)) \\ CW_{end} &= (C_{timg} + (w_{timg} * 0.1)) \\ CW_{start} &= (C_{timg} - (w_{timg} * 0.1)) \end{aligned} \tag{5}$$

3) Syntax Analysis: Syntax analysis is responsible for converting the data of each token into useful information. The information generated in this step is the speed, direction, and acceleration of each identified dynamic object. An object is considered dynamic or not, according to the pre-defined classification given in Table I. This table also indicates which dynamic objects are alive. Thus, at the end of this analysis, each identified object will have information such as dynamic or static, alive or not, velocity, direction, and acceleration if they are in motion.

TABLE I
INFERRED KNOWLEDGE IN OBJECTS

	Object class
Is it dynamic?	person, bicycle, car, motorbike, aeroplane, bus, train, truck, boat, bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe, frisbee, snowboard, sports ball, skateboard, surfboard tennis racket, chair
Is it alive?	person, bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe

• *Identification of the same token* The displacement of the object in space allows the computation of speed, direction, and acceleration. Thus, it is necessary to have the position of the same object in two instants of time, to calculate these variables. The strategy to identify the same object in two moments is given by a simple comparison between all tokens of two objects at two instants of time,present time $(T_{present})$ and past time (T_{past}). In this way, the identification of the same token in both times is made by comparing all objects from time $T_{present}$ with objects from time T_{pass}. First, it is compared whether the two tokens are dynamic and belong to the same class, if it is true, the Euclidean distance between the real position of the objects, identified through the point cloud, is compared. If the Euclidean distance is bigger than 0.15, we assume that it is not the same object. This value was defined empirically, after proving to be enough to not lose the movement of an object and to prevent different grouping tokens.

• *Velocity Calculation* eThe object's speed, direction, and acceleration are based on the same token identification in two moments.. The speed calculation is given by the displacement of the object in the environment $(\delta Distance)$ divided by time $(\delta Time)$. The distance is obtained by calculating the Euclidean distance between $(P_t[x, y, z]_{T_{pass}})$ and $(P_t[x, y, z]_{T_{present}})$. Time is obtained by the difference between $T_t T_{present}$ and $T_t T_{pass}$, where $T_{present}$ refers to the time of the last set of tokens collected, and T_{pass} refers to the tokens previously received. Finally, the sensor speed, obtained by the egomotion analysis, is subtracted from the speed of the token. In this way, the calculation of the linear token velocity (LV_t) is given by the equation 6.

$$LV_t = \frac{\delta Distance}{\delta Time} - LVS \tag{6}$$

where LVS is the linear speed of the sensor.

• *Direction calculation* The direction is used to check the object's trajectory in the environment. This calculation is done by measuring the angle between the same token in two moments using the function $atan2$ [33]. The direction is calculated in 2D, thus only $(P_t[x, y]_{T_{pass}})$ and $(P_t[x, y]_{T_{present}})$

are used, despising the z information for each identified object. The $atan2$ function returns the angle between the two points in radian. This information is added to the angular speed of the DeepSpatial sensor obtained in the analysis of egomotion. The direction (D_t) calculation is given by the equation 7.

$$D_t = atan2((P_t[y]_{T_{present}} - P_t[y]_{T_{pass}}), (P_t[x]_{T_{present}} - P_t[x]_{T_{pass}}) - AVS \quad (7)$$

where AVS is the angular speed of the sensor.

• *Acceleration calculation* The acceleration of an object is due to the speed difference during two instants of time divided by time. In this way, it is only possible to calculate the acceleration of objects that already have speed. Therefore, to perform the acceleration calculation (ACC_t), it is first checked whether the token has speed at two instants of time, if so, equation 8 is computed, which corresponds to the speed variation (δVel) by the time variation ($Time$).

$$ACC_t = \frac{LV_tT_{present} - LV_tT_{pass}}{\delta Time} \quad (8)$$

4) Prediction Analysis: Prediction analysis uses egomotion data, lexical, and syntax analyses, to infer the future position of the object. The prediction of the next object, position is computed through the speed, acceleration, and direction of a dynamic object. These future positions can be used to prevent potential collisions between dynamic objects and the DeepSpatial sensor.

The future position of dynamic objects at a different time in the future (T_p) is computed based on acceleration information. If an object has an acceleration value, equation 9 is used to identify the object's displacement in space (Ob_d). If the object has no acceleration, its speed, considered as constant, is used, multiplying T_p by LV_t.

$$Ob_d = LV_t * T_p + ACC_t * \frac{T_p^2}{2} \quad (9)$$

After calculating the displacement of the object in the environment, it is possible to calculate its future position after T_p seconds. Having the object's spatial position ($P_t[x, y, z]$) its displacement in the environment (Ob_d) and its direction (D_t), it is possible to calculate its new position in the environment ($P_p[x, y]$) as presented in Equation 10. The displacement of the object on the Z-axis is not considered.

$$P_p[x] = P_t[x] + \sin(Ob_d + 90) * Ob_d; \quad P_p[y] = P_t[y] + \cos(Ob_d + 90) * Ob_d; \quad (10)$$

As all collected data are related to the DeepSpatial sensor center, the calculation of the future position of the sensor is applied using the equation 9 considering, instead of ACC_t, the linear speed of the sensor LVS. The points used to represent the initial position of the sensor are [0,0,0] because the sensor is considered the origin of the coordinate plane. The new predicted position of the sensor in T_p time is defined as Sp_p.

• *Collision prediction* Having the future position of all the identified dynamic objects and the DeepSpatial sensor position, at time T_p, it is possible to predict possible collision paths. The collision predaiction is carried out by calculating the Euclidean distance between the future position of the DeepSpatial sensor, and all future positions of the dynamic objects identified in the environment. If this distance is less than a predetermined threshold, it means that the sensor and the dynamic object will be very close to each other in the future, signaling a possible collision.

By default, the future position of identified objects is continually calculated with T_p taking values of 1, 3, and 5 seconds. After the next location of all objects is obtained, the collision prediction is also carried out for each instant of time. If possible collisions are predicted, an alert is published, so that the DeepSpatial sensor user can take the appropriate actions.

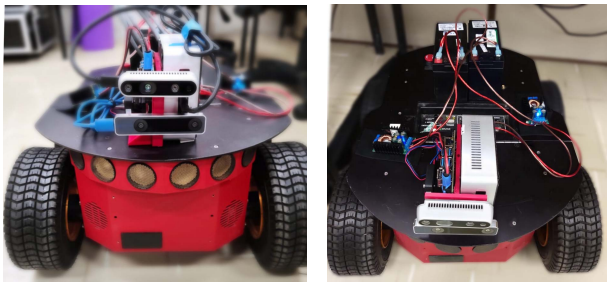


Fig. 3. Sensor attached to the Pioneer P3-AT robot. Left, front view of the robot. Right, top view of the robot, where it is possible to view the batteries for powering the sensor.

III. Results and Discussions

This section aims to present the results obtained with the developed *Intelligent Spatial Sensor to Perception of Things*. A first experiment is presented in order to analyze as the operating data are collected and processed by DeepSpatial sensor. Then, an experiment with the DeepSpatial sensor embedded in a mobile robot is carried out. The mobile robot has linear and angular speed control, making it possible to perform a comparison between the speeds obtained by the robot and by the proposed DeepSpatial sensor.

The mobile robot Pioneer P3-AT was used (Figure 3). This robot is compatible with ROS, and it has been connected to the DeepSpatial sensor. From the wireless network created by the DeepSpatial sensor, it was possible to collect data from the DeepSpatial sensor and send commands to the robot. The robot has linear and angular speed control, besides encoders used to calculate these speeds logically. All the described experiments were carried out with the DeepSpatial sensor embedded to the robot, powered by batteries and communicating through the wireless network created by the DeepSpatial sensor.

A. Knowledge Extraction From Collected Data

All information processed by the DeepSpatial sensor can be visually obtained throughout a user interface. Thus it is possible to view the identified objects, their positions around the sensor, their predicted positions, and the possible collision paths. This information is also available in a textual form, through a topic from ROS. In this way, the information can be

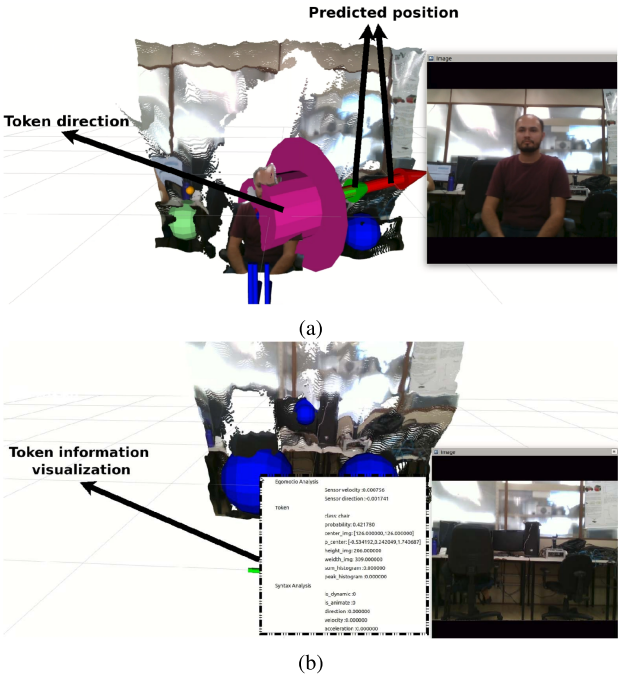


Fig. 4. Graphical display provided by the DeepSpatial sensor. (a) Representation of a dynamic object, its movement and future positions. (b) Token information.

directly read from the DeepSpatial sensor in order to support actions and decisions, such as stopping the robot or deflect it if a future collision is predicted, or look for a specific object in the environment and others. This DeepSpatial sensor opens up a wide range of options in the field of mobile robotics.

Figure 4 and Figure 5 show the information provided by the DeepSpatial sensor. In Figure 4, the sensor is directly connected to a monitor, where the information is displayed. First, the image is processed by YOLO, then identified object information such as object's position in the world is calculated based on data coming from the RGB-D view. The appearance of dynamic objects changes from sphere to an arrow, pointing to the calculated direction for object displacement. The information about predicted positions are also displayed as arrows, pointing to the possible future positions of the object. All future positions can be visible or it is possible to filter them for 1, 3, and 5 seconds. Finally, when a possible collision between objects is inferred, a black sphere is generated around the possible collision locus.

B. Performance Analysis

All experiments are carried out at a frequency of 10 Hertz. After 30 minutes from the beginning of the operation, the DeepSpatial sensor CPU (Intel Nuc) is operating at 53.4% and using only 10.42% of memory. The CPU is handily running; however, the entire system is limited by YOLO's update rate. If YOLO operates at 2 Hertz, the whole system will work at the same frequency. There are different versions of YOLO, the last being YOLOv3. A smaller version, but with less precision, is the YOLOv3-tiny, it operates at a higher

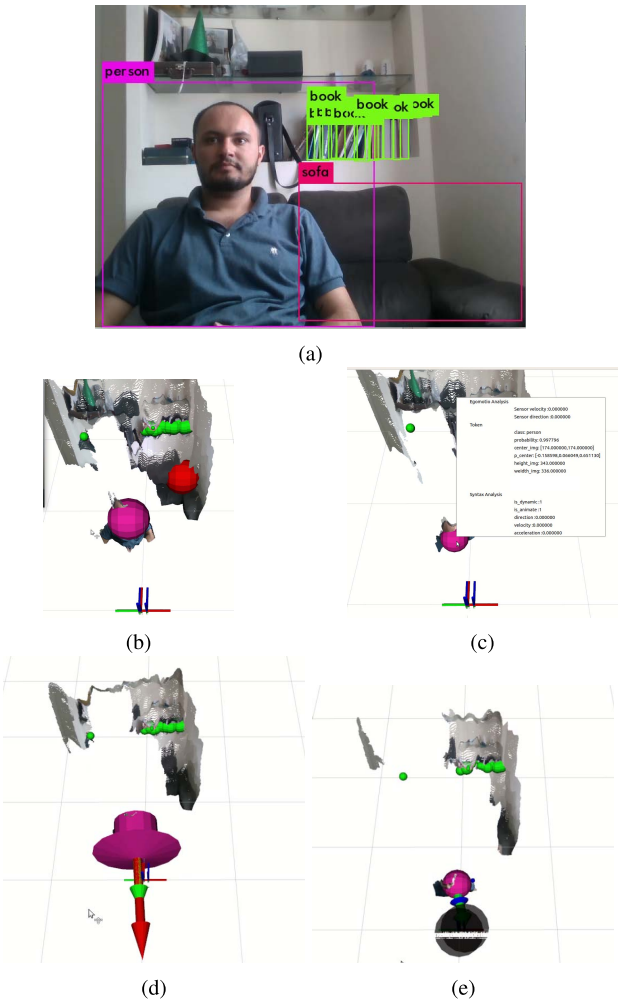


Fig. 5. Graphical display provided by the DeepSpatial sensor. The grid is represented in meters. (a) YOLOv3-tiny. (b) 3D position of the objects identified in the environment. (c) Information about a specific object. (d) A vector pointing to the predicted positions in 1 (blue), 3 (green) and 5 (red) seconds. (e) Possible collision warning.

frequency but with a lower mean Average Precision (mAP). Figure 6 presents two bar plots, where the first one shows the operating frequencies of the YOLO version running on the DeepSpatial sensor, and the second shows the mAP of all versions, according to its developer [15], [16]. YOLOv3 has the highest mAP, but its update rate is minor (1.40). YOLOv2 has a good mAP and an acceptable refresh rate in some situations. YOLOv3-tiny offers a reasonable update rate and an adequate mAP. GPU usage remains 99% regardless of the chosen version of YOLO.

C. Egomotion

The DeepSpatial sensor's ability to capture and calculate its displacement is evaluated in the next experiments in which the DeepSpatial sensor is connected to the mobile robot. Visually, Figure 7 presents a representation of the robot in motion, and stopping in front of a person. In Figure 7.a and 7.c, it is possible to observe that the robot calculated its future

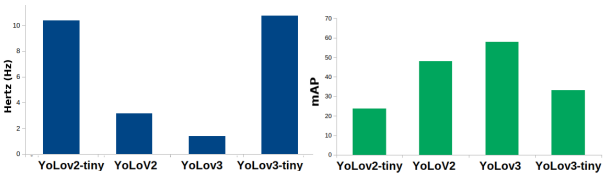


Fig. 6. On the left, frequency of operation in Hertz. On the right, Mean Average Precision (mAP).

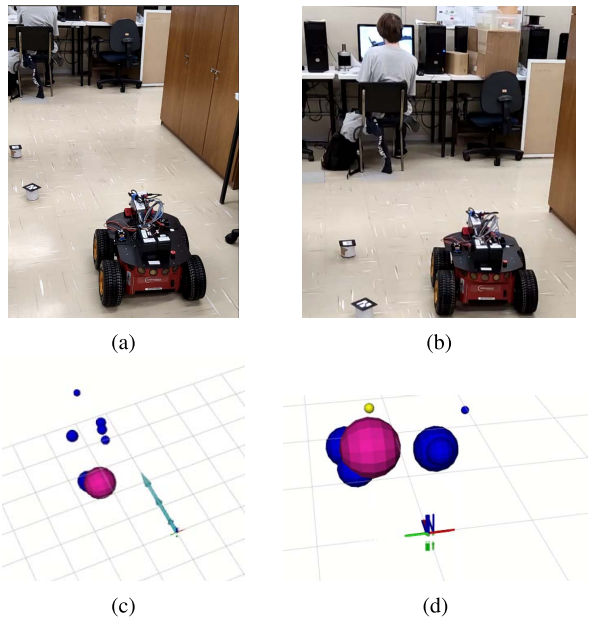


Fig. 7. DeepSpatial sensor embedded into the robot and all sensor information are collected over the wireless network. The grid is represented in meters. (a) Robot in motion. (b) Robot stopped. (c) Information calculated during robot motion. (d) Information calculated during robot stop.

position, but It did not consider the person as a moving object, this is because it compensated for its speed angular and linear with the calculated velocities for the dynamic object. In Figure 7.b and 7.d, future movements are not calculated for the robot, nor for the dynamic object, as both are still stoped.

The spatial motion capture is compared with odometry computed by the mobile robot. The robot is a standard mobile platform that estimates its relative displacement through a fusion of encoder odometry and inertial movement sensor. This estimation is susceptible to errors because it is based on dead-reckoning, with error accumulation. The results are presented in Figure 8, where "commands" represents the speed reference sent to the robot controller, "robot" represents the speed calculated by the robot's encoder and DeepSpatial sensor represents the speed calculated by the sensor DeepSpatial sensor.

The average error for the linear velocity calculated by the sensor was around 0.04 m/s, and for the angular velocity, it was around 0.06 m/s when compared to the speed obtained by the robot's encoder. Figure 9 presents a boxplot of the difference between speed data from both DeepSpatial sensor and the robot encoder. This error does not significantly affect

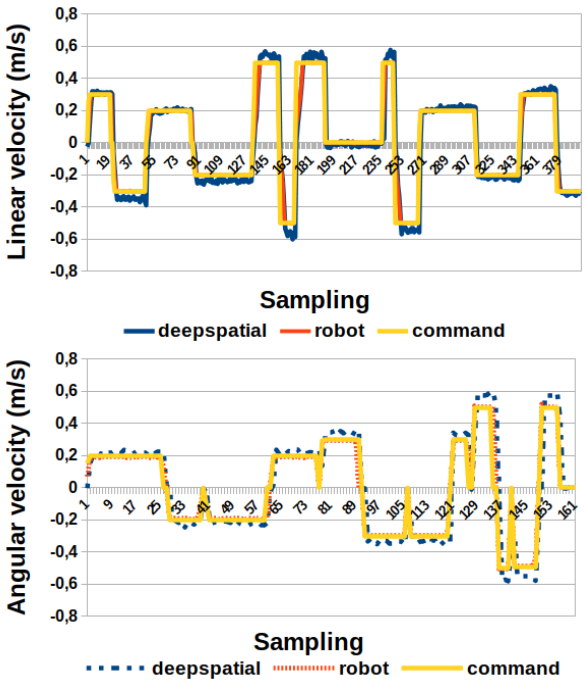


Fig. 8. Representation of linear and angular velocities measured during the experiment. commands are the velocity references sent to the robot controller. Robot is the speeds calculated by the robot's encoder. DeepSpatial sensor is the speeds calculated by the DeepSpatial sensor.

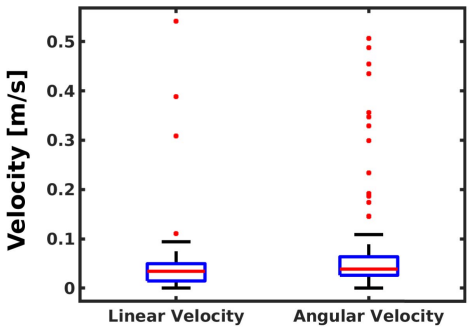


Fig. 9. Boxplot of the error between the speeds calculated by the DeepSpatial sensor and by the robot.

the calculation of the future speed for identified dynamic objects. For example, an object has been identified at 1 meter from the robot and it moves in $\delta t 1$, that is, 1 second in the future, it will be identified at 0.96 meters, causing the robot to detect the collision in advance.

D. Object Tracking

This experiment aims to analyze the behavior of the proposed strategy and the Intel RealSense Tracking Camera T265 tracking sensor. Specific information about its sensor can be obtained at [34], [35].

One of the main difficulties in calculating the future position, speed, acceleration, and direction of an object, is to identify the same object in two moments. The strategy developed in this work uses only the Euclidean distance between the

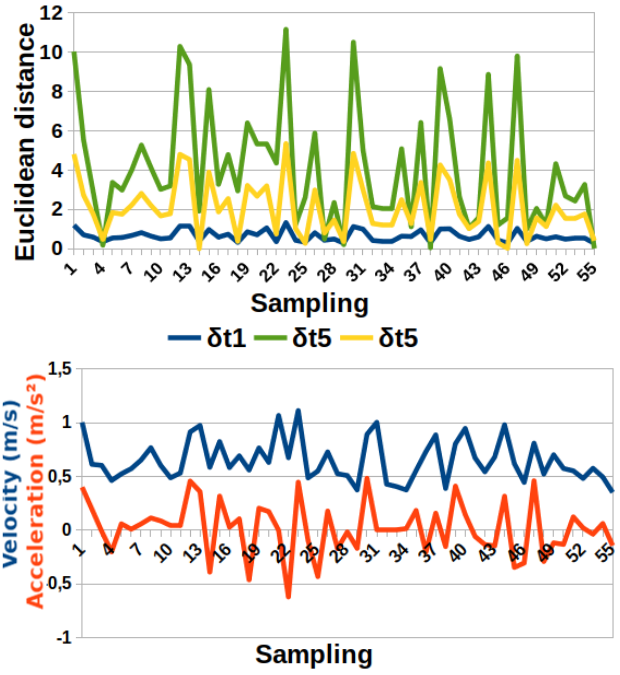


Fig. 10. Top figure: Euclidean distance between the current object position, and its predicted positions in 1, 3 and 5 seconds. Bottom figure: velocity and acceleration calculated for the object.Left Plot of the distance between the current position of the object, and its predicted positions in 1, 3 and 5 seconds.Right plots of velocity and acceleration calculated for the object.

position of dynamic objects in two moments. If this distance is less than 0.15 and more significant than 0.035, and the objects belong to the same class, then they are considered the same object. As the update frequency of the sensor is high, it is allowed to use a shorter distance, since the object does not move much between the two-time instants. The distance value greater than 0.035 is used to avoid false calculations, resulting from small movements of the object.

Figure 10 shows the tracking of a moving person. On the left plot, the Euclidean distance between the person's current position and his predictions of future positions is shown, on the right plot, the calculated acceleration and speed profiles are presented. When both acceleration and speed are high, the future status of the object is calculated at a greater distance, as shown at position 24 of the plot. When we have a high speed, and low acceleration, the object's next location is considered to be less since the object is decelerating. In some cases, with a negative acceleration value, the object is deemed to stop in the future, position 4 of the plot.

The validation of dynamic object tracking by the DeepSpatial sensor is carried out by an experiment with the worst possible scenario: two dynamic objects of the same class are side by side. In this way, the proposed tracking algorithm must differentiate the objects to carry their tracking. The strategy used for this action was presented in the section II-B.3. During the experiment, two people walk side by side, and the Euclidean distance between these two positions obtained by the DeepSpatial sensor must be monitored. Figure 12 shows

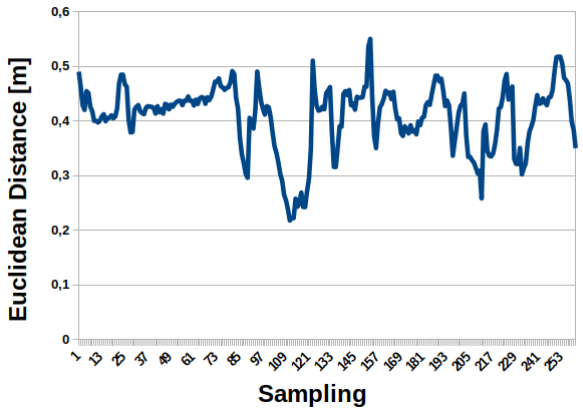


Fig. 11. Graph representing the distance between the two people during the experiment. People walked back and forth, side by side.

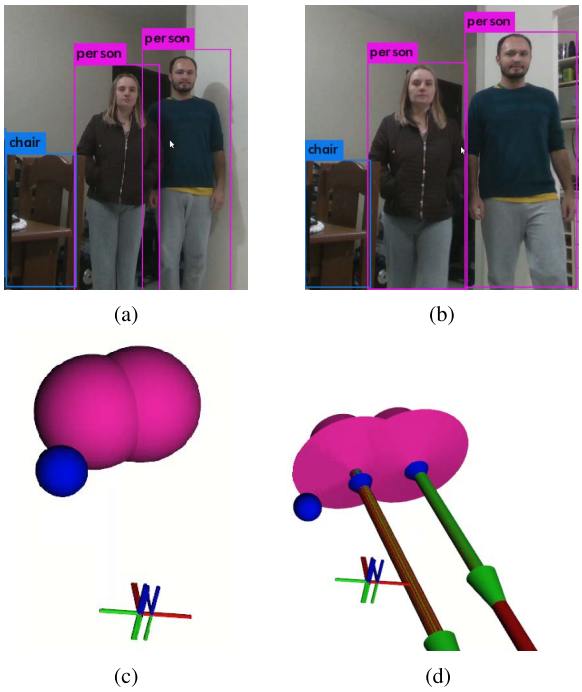
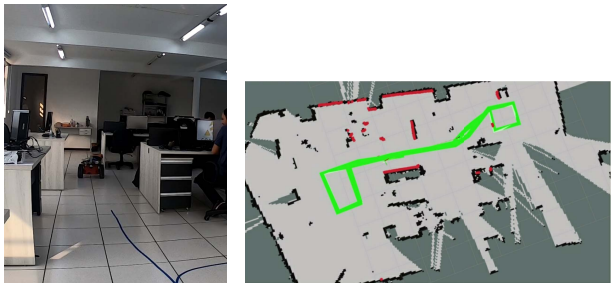


Fig. 12. DeepSpatial sensor view aimed at two people side by side. (a) People are standing still. (b) People are on the move. (c) Sensor output, people are standing still. (d) Sensor output, people in motion, with their future positions calculated.

the two people standing side by side, and then moving, where it is possible to observe the calculation of the future positions for both dynamic objects. Figure 11 presents the Euclidean distance between the objects (people) during the experiment. It is worth mentioning that during the entire monitoring, both people were correctly identified and tracked, validating the proposed tracking algorithm.

IV. Application Example

The DeepSpatial sensor will be demonstrated in an example task To validate the approach proposed by this article. The equipment will be integrated into an autonomous navigation strategy and used as safety equipment to prevent accidents.



(a) (b)

Fig. 13. Pioneer P3-At robot performing autonomous navigation. (a) Robot in the test environment, being an office. (b) Map of the environment, the trajectory to be covered in green, and the robot's position on the map.

The Pioneer P3-AT robot is assigned to perform autonomous navigation from point to point. This is a traditional strategy for autonomous cargo transport robots [36]–[38] and, thus, validate the sensor during the execution of a conventional task in the area of mobile robotics. The robot's navigation circuit will be repetitive, so the robot will always navigate the same environment.

The DeepSpatial sensor will send environmental information to the robot, such as the need to perform an emergency stop. The robot will read the sensor's information in the token format and then decide to *stop* or *continue* according to the information provided by the DeepSpatial sensor. Some rules will be created for the robot's action based on the sensor's statement, which is presented below.

- If the prediction analysis identifies a collision in the present position or predicted positions in the future (1,3 and 5 seconds), the robot must save the navigation data for analysis.
- If the objective is in motion, but is not towards the robot, and is at a distance greater than 0.5 meters, the robot must continue its trajectory, and thus avoid an unnecessary stop.

In this way, the robot will perform navigation from point to point, repeating the points, and stopping when some of the conditions mentioned previously are reached. The distance between the object and robot at the time of stop will be stored, and the speeds of the object and robot, to evaluate the performance.

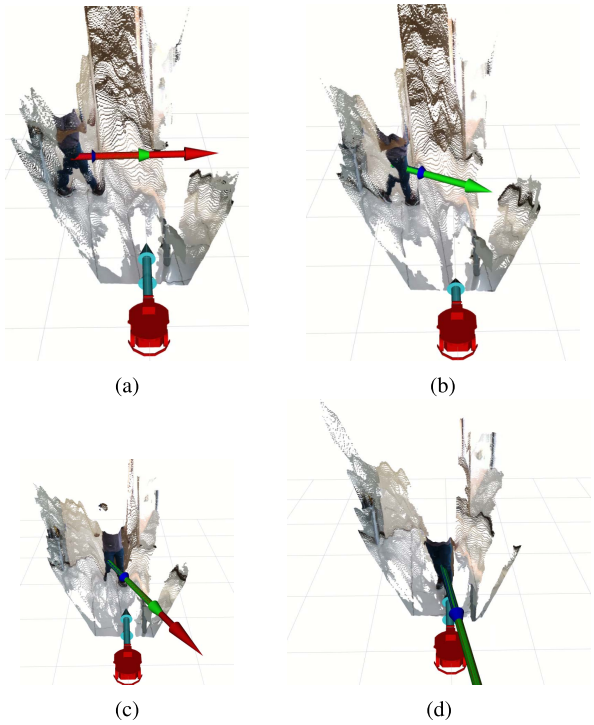
The robot sailed for 1 hour in an office, and 60 possible collisions were identified, being a possible collision in the present time, 17 potential collisions in one second, 26 collisions predicted in 3 seconds, and 16 predicted collisions for 5 seconds in the future, according to with the prediction analysis developed in this paper.

Figure 13 shows the robot navigating the defined circuit,where the robot makes a map of the environment, and then runs the SLAM. The navigation and localization technique is not interesting for this work, being used only and exclusively to validate DeepSpatial in a real application.

During navigation, the DeepSpatial sensor was turned on, and when it identified a possible collision, it wrote down the information. Table II presents an average containing the distance between the robot and the Token at the time of the collision is detected, both in the present time (0) and in the future (1.3 and 5 seconds). The robot's actual distance and the predicted distance to the object are also shown in the table. It is worth mentioning that the higher the object's speed and

TABLE II
COLLISION DATA DETECTED BY THE DEEPSPATIAL SENSOR DURING THE AUTONOMOUS NAVIGATION OF THE ROBOT. A TOTAL OF 60 COLLISIONS WERE IDENTIFIED, ONE AT TIME 0, 17 AT TIME 1, 26 AT TIME 3, AND 16 AT TIME 5

Future time (seconds)		Distance (Token future) (Robot prediction)	Distance (Token future) (Robot real)	Robot speed	Object speed
0		**0,476 (Average)**	**0,476 (Average)**	**0,211 (Average)**	**0,391 (Average)**
	Min:	0,476	0,476	0,211	0,391
	Max:	0,476	0,476	0,211	0,391
1		**0,222 (Average)**	**0,304 (Average)**	**0,189 (Average)**	**0,901 (Average)**
	Min:	0,038	0,037	0,123	0,301
	Max:	0,487	1,040	0,231	1,366
3		**0,338 (Average)**	**0,705 (Average)**	**0,174 (Average)**	**0,461 (Average)**
	Min:	0,026	0,188	0,093	0,014
	Max:	0,497	1,163	0,254	0,951
5		**0,298 (Average)**	**0,852 (Average)**	**0,181 (Average)**	**0,469 (Average)**
	Min:	0,149	0,300	0,094	-0,017
	Max:	0,499	1,501	0,232	1,226



(a) (b) (c) (d)

Fig. 14. Image provided by DeepSpatial during its operation. The identification, tracking, and prediction of the position of a person on the movement is presented. (a) The person is perpendicular to the sensor. (b, c) The person is performing a turning action. (d) The person is moving towards the sensor.

the robot, the greater the distance from the predicted collision, as the forecast will consider that the object is accelerating.

As DeepSpatial calculates the identified objects' speed and direction, it considers that a collision will only happen if the object is in its direction, thus avoiding unnecessary stops and accidents, anticipating a stop or slow down. Figure 14 presents an identified trajectory of a person, where first, he is going in a direction perpendicular to the robot. Then he performs a contour action and goes towards the robot.

This section presented the software and hardware of the DeepSpatial sensor proposed by this work, also discusses the sensor's advantages in a traditional application in mobile robotics and sensing.

V. Conclusion

This work has developed an embedded sensor, composed of a set of components, where the raw data of each component is collected and gathered, generating useful information for several applications. With the proposed sensor, it is possible to develop an intelligent equipment capable of identifying dynamic objects and tracking them, in addition it also provides information such as, for example, a bottle is on a table, which can be used by a household robot, for example. In this way, the development of such intelligent equipment can be concerned with treating the information from the DeepSpatial sensor and not trying to collect them from the environment.

The sensor processing runs on Intel Nuc NUC5i5RYH, and it is observed that after 30 minutes of uninterrupted use, the computer remained with only 53.4 % of its processing capacity and 10.42 % of its occupied memory. The Jetson Nano was used to perform object detection. When using YOLO to identify objects, Jetson Nano used 99% of its GPU. However, this use does not represent a risk during execution, since it has a CPU, leaving the GPU dedicated to YOLO. In terms of hardware, the sensor proved to be satisfactory, having no problems at run-time, always running online.

The logical approach is organized in some steps, *egomotion analysis*, *lexical analysis*, *syntax analysis*, and *prediction analysis*. In egomotion analysis, the Intel RealSense Tracking Camera T265 is used to identify the sensor's movement. The ability to identify movements was verified, where its presentedresults have attained an average error concerning the data obtained by the robot of 0.04 at linear speed, and 0.06 for angular speed. Then the lexical analysis is performed, where all the information of the object is collected, using the YOLO and the RGB-D depth sensor. In syntax analysis, the data collected from the objects is used to calculate their displacement in the environment, direction, speed, and acceleration. In prediction analysis, a prediction of the future position of all dynamic objects is carried out. This prediction is able to prevent a possible collision between two dynamic objects in the environment.

The experiment results have showed that the DeepSpatial sensor performance was satisfactory. Its limited frequency of operation is directly linked to YOLO. However, a new setting can be done since we choose between using YOLOv2 for greater accuracy, and operating at a low rate, or losing efficiency using the YOLOv3-tiny and operating at a frequency of 10 hertz. In future works, the replacement of either Jetson Nano or YOLO will be considered to seek a reasonable rate with a better accuracy on the identified objects. Finally, the sensor was proposed and used in a real application. Thus, this article proposed not only creating the sensor, in terms of hardware and software, but also brought examples of application.

Acknowledgment

The authors gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

[1] L. Jiang, J. Zhang, and B. Deng, "Robust RGB-D face recognition using attribute-aware loss," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 10, pp. 2552–2566, Oct. 2020.

[2] L. Ren, J. Lu, J. Feng, and J. Zhou, "Multi-modal uniform deep learning for RGB-D person re-identification," *Pattern Recognit.*, vol. 72, pp. 446–457, Dec. 2017.

[3] R. Méndez Perez, F. A. Cheein, and J. R. Rosell-Polo, "Flexible system of multiple RGB-D sensors for measuring and classifying fruits in agri-food industry," *Comput. Electron. Agricult.*, vol. 139, pp. 231–242, Jun. 2017.

[4] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[5] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving RGB-D SLAM in dynamic environments: A motion removal approach," *Robot. Auto. Syst.*, vol. 89, pp. 110–122, Mar. 2017.

[6] Y. Liu, X.-Y. Jing, J. Nie, H. Gao, J. Liu, and G.-P. Jiang, "Context-aware three-dimensional mean-shift with occlusion handling for robust object tracking in RGB-D videos," *IEEE Trans. Multimedia*, vol. 21, no. 3, pp. 664–677, Mar. 2019.

[7] M. Jiang, Z. Pan, and Z. Tang, "Visual object tracking based on cross-modality Gaussian-Bernoulli deep Boltzmann machines with RGB-D sensors," *Sensors*, vol. 17, no. 12, p. 121, Jan. 2017.

[8] P. F. Proença and Y. Gao, "Probabilistic RGB-D odometry based on points, lines and planes under depth uncertainty," *Robot. Auto. Syst.*, vol. 104, pp. 25–39, Jun. 2018.

[9] P. F. Proenca and Y. Gao, "SPLODE: Semi-probabilistic point and line odometry with depth estimation from RGB-D camera motion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 1594–1601.

[10] T. Wilaiprasitporn, A. Ditthapron, K. Matchaparn, T. Tongbuasirilai, N. Banluesombatkul, and E. Chuangsuwanich, "Affective EEG-based person identification using the deep learning approach," *IEEE Trans. Cognit. Develop. Syst.*, vol. 12, no. 3, pp. 486–496, Sep. 2020.

[11] A. Zhavoronkov *et al.*, "Deep learning enables rapid identification of potent DDR1 kinase inhibitors," *Nature Biotechnol.*, vol. 37, no. 9, pp. 1038–1040, Sep. 2019.

[12] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," 2019, *arXiv:1901.03407*. [Online]. Available: http://arxiv.org/abs/1901.03407

[13] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, "Car detection using unmanned aerial vehicles: Comparison between faster R-CNN and YOLOV3," in *Proc. 1st Int. Conf. Unmanned Vehicle Syst.-Oman*, 2019, pp. 1–6.

[14] B. Benjdira, Y. Bazi, A. Koubaa, and K. Ouni, "Unsupervised domain adaptation using generative adversarial networks for semantic segmentation of aerial images," *Remote Sens.*, vol. 11, no. 11, p. 1369, Jun. 2019.

[15] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767

[16] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7263–7271.

[17] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, Nov. 2019.

[18] X. Liu, M. Yan, and J. Bohg, "MeteorNet: Deep learning on dynamic 3D point cloud sequences," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9246–9255.
[19] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 770–779.
[20] S. Hossain and D. J. Lee, "Deep learning-based real-time multiple-object detection and tracking from aerial imagery via a flying robot with GPU-based embedded devices," *Sensors*, vol. 19, no. 15, p. 3371, Jul. 2019.
[21] G. Ning *et al.*, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.
[22] X. Lv *et al.*, "A robust real-time detecting and tracking framework for multiple kinds of unmarked object," *Sensors*, vol. 20, no. 1, p. 2, Dec. 2019.
[23] M. Jiang, T. Hai, Z. Pan, H. Wang, Y. Jia, and C. Deng, "Multi-agent deep reinforcement learning for multi-object tracker," *IEEE Access*, vol. 7, pp. 32400–32407, 2019.
[24] Y. Yoon *et al.*, "Analyzing basketball movements and pass relationships using realtime object tracking techniques based on deep learning," *IEEE Access*, vol. 7, pp. 56564–56576, 2019.
[25] M. A. Simoes Teixeira, N. Dalmedico, A. S. de Oliveira, L. V. Ramos de Arruda, and F. Neves, "A pose prediction approach to mobile objects in 2D costmaps," in *Proc. Latin Amer. Robot. Symp. (LARS) Brazilian Symp. Robot. (SBR)*, Nov. 2017, pp. 1–6.
[26] J. Zhong, H. Sun, W. Cao, and Z. He, "Pedestrian motion trajectory prediction with stereo-based 3D deep pose estimation and trajectory learning," *IEEE Access*, vol. 8, pp. 23480–23486, 2020.
[27] M. Teixeira *et al.*, "Intelligent 3D perception system for semantic description and dynamic interaction," *Sensors*, vol. 19, no. 17, p. 3764, Aug. 2019.
[28] M. Quigley *et al.*, "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, Kobe, Japan, vol. 3, no. 3.2, 2009, p. 5.
[29] A. Koubaa, *Robot Operating Syst. (ROS): The Complete Reference*, vol. 4, 4th ed. Cham, Switzerland: Springer, 2020.
[30] S.-H. Tsao and S.-S. Jan, "Observability analysis and performance evaluation of EKF-based visual-inertial odometry with online intrinsic camera parameter calibration," *IEEE Sensors J.*, vol. 19, no. 7, pp. 2695–2703, Apr. 2019.
[31] W. Ci, Y. Huang, and X. Hu, "Stereo visual odometry based on motion decoupling and special feature screening for navigation of autonomous vehicles," *IEEE Sensors J.*, vol. 19, no. 18, pp. 8047–8056, Sep. 2019.
[32] M. Aladem and S. A. Rawashdeh, "A combined vision-based multiple object tracking and visual odometry system," *IEEE Sensors J.*, vol. 19, no. 23, pp. 11714–11720, Dec. 2019.
[33] Z. Luo, J. Ding, L. Zhao, and M. Wu, "An enhanced non-coherent pre-filter design for tracking error estimation in GNSS receivers," *Sensors*, vol. 17, no. 11, p. 2668, Nov. 2017.
[34] A. Alapetite, Z. Wang, J. P. Hansen, M. Zajączkowski, and M. Patalan, "Comparison of three off-the-shelf visual odometry systems," *Robotics*, vol. 9, no. 3, p. 56, 2020.
[35] O. Mise, R. Madison, and B. Haight, "A comparison of SWaP-limited, visual-inertial odometry systems for GPS-denied navigation," *Proc. SPIE*, vol. 11424, Apr. 2020, Art. no. 1142406.
[36] A. Saffiotti, "The uses of fuzzy logic in autonomous robot navigation," *Soft Comput. Fusion Found., Methodologies Appl.*, vol. 1, no. 4, pp. 180–197, Dec. 1997.
[37] E. Krell, A. Sheta, A. P. R. Balasubramanian, and S. A. King, "Collision-free autonomous robot navigation in unknown environments utilizing PSO for path planning," *J. Artif. Intell. Soft Comput. Res.*, vol. 9, no. 4, pp. 267–282, Oct. 2019.
[38] M. M. Ejaz, T. B. Tang, and C.-K. Lu, "Vision-based autonomous navigation approach for a tracked robot using deep reinforcement learning," *IEEE Sensors J.*, early access, Aug. 13, 2020, doi: 10.1109/JSEN.2020.3016299.

Marco Antonio Simões Teixeira received the M.Sc. degree in electrical and computer engineering from the Federal University of Technology at Parana (UTFPR), Parana, Brazil, in 2017, where he is currently Ph.D. degree in electrical and computer engineering. His research interests include mobile robots in the areas of perception and intelligent systems, computer vision, navigation, mapping, and applied AI.

Flávio Neves-JR received the B.Sc. and M.Sc. degrees in electrical engineer from the Federal University of Technology, Parana, Brazil, in 1987 and 1989, respectively, and the Ph.D. degree in electrical engineering from the University Paul Sabatier (LAAS), France, in 1998. Since 1992, he has been with the Federal University of Technology, Parana, where he is a Full Professor. His research interests include hardware and software to instrumentation and automation.

Anis Koubaa (Member, IEEE) is currently a Professor of Computer Science and the Leader of the Robotics and Internet of Things Research Laboratory, Prince Sultan University. He is also a Senior Researcher with CISTER and ISEP-IPP, Porto, Portugal. His current research interests include providing solutions toward the integration of robots and drones into the Internet of Things (IoT) and clouds, in the context of cloud robotics, deep learning, robot operating systems (ROS), cloud robotics, and unmanned aerial systems. He is also a Senior Fellow of the Higher Education Academy (HEA), U.K. He has been the Chair of the ACM Chapter in Saudi Arabia, since 2014.

Lúcia Valéria Ramos de Arruda (Member, IEEE) received the degree in electrical engineer from the Federal University of Ceara, Brazil, in 1985, the M.Sc. degree from the Graduate School of Electrical Engineering, Campinas State University (FEE/UNICAMP), Brazil, in 1988, and the Ph.D. degree in electrical engineering from the University of Nice-Sophia Antipolis, France, in 1992. Since 1995, she has been with the Federal University of Technology, Parana, where she is a Full Professor. Her research interests include soft computing methods to model and control of dynamic systems.

André Schneider de Oliveira (Member, IEEE) received the M.Sc. degree in mechanical engineering, focused in force control of rigid manipulators, from the Federal University of Santa Catarina (UFSC) in 2007, and the Ph.D. degree in engineering of automation and systems, in 2011, with thesis focused on differential kinematics through dualquaternions for vehicle-manipulator systems. He is currently an Adjunct Professor with the Federal University of Technology, Parana (UTFPR). He is a member of the Advanced Laboratory of Robotics and Embedded Systems (LASER) and Automation and Advanced Control System Laboratory (LASCA). His research interests include robotics, mechatronics, and automation with special focus in navigation and localization of mobile robots, autonomous and intelligent systems, perception and environment identification, cognition, deliberative decisions, human-interaction, and navigation control.

7 CONCLUSION AND FUTURE WORK

This thesis proposes a new sensing approach for mobile robots. Mobile robots need to see the environment around them to make decisions, such as avoiding obstacles. The sensors can also be used to develop mapping techniques, object identification, inspection, and many other activities. This thesis aimed to improve the data coming from traditional RGB-D sensors, which are images with a notion of depth, in order to develop a device capable of identifying objects and tracking them, they can be used by the robot for making the decisions. The thesis is written in the format of a collection of articles, thus, some chapters of the thesis are articles already published in journals.

Chapter 1 presents the problem of this thesis, with a brief description of mobile robots' need to perceive the environment around them. The general and specific objectives of the thesis were also presented. Chapter 2 presented a contextualization of the thesis, comparing the works developed with related works in the literature and associating each work with the general and specific objectives proposed.

Chapter 3 presents the work *Intelligent environment recognition and prediction for NDT inspection through autonomous climbing robot* (TEIXEIRA *et al.*, 2018), where a strategy for mapping Liquefied Petroleum Gas (LPG) storage tanks to a mobile inspection robot was developed. For the preparation of the map, knowledge of the inspection environment was inferred, such as the spacing between the weld beads. In this way, it was possible to predict the entire structure, by identifying parts of the tank, such as weld junctions. This work made use of several sources of perception, being 1D, 2D, and 3D. In this way, this work explored to the maximum of the use of traditional perception sensors, making evident the need for improvement in this area.

Chapter 4 presents the work *A Quadral-Fuzzy Control Approach to Flight Formation by a Fleet of Unmanned Aerial Vehicles* (TEIXEIRA *et al.*, 2020) where a drone formation strategy was proposed. The strategy aims to carry out the flight while maintaining a specific formation between 4 drones so that they can perform collaborative tasks, such as cargo transportation. The work uses a single RGB-D sensor to detect obstacles, and Fuzzy controllers to perform obstacle avoidance and position control. As a result, the strategy proved capable of maintaining formation even during obstacle avoidance's. This work fulfills the second specific objective of the thesis, where point cloud processing techniques are applied in aerial robots

so that the sensing techniques can be validated not only in terrestrial robots but also in aerial robots.

The strategy presented in the chapter 3 and 4 made use of several traditional sensors in mobile robotics, showing the need for innovation in this area. In this way, the work presented in the chapter 5 (TEIXEIRA *et al.*, 2019) was developed. Chapter 5 presents the work *Intelligent 3D Perception System for Semantic Description, and Dynamic Interaction* (TEIXEIRA *et al.*, 2019), where a new sensing strategy is proposed. In this work, a new approach to sensing from traditional sources of perception is proposed. In this way, the RGB-D perception sensor is used to generate new types of information, such as identifying objects in the environment and their spatial positions. It is used for object identification, computer vision techniques; in this work, the YoLo algorithm was used. This chapter fulfilled the third specific objective of the thesis, focusing on elaborating a new sensing proposal.

Chapter 6 presents the work *DeepSpatial: Intelligent Spatial Sensor to Perception of Things* (TEIXEIRA *et al.*, 2021), an improvement on the previous work, where the proposed strategy is embedded in a compact equipment that can be attached to the mobile robot. The proposed sensor comprises several pieces of equipment: a computer (an Nvidia Jetson board to perform the graphic processing), an RGB-D camera, and a camera to perform the visual odometry. With visual odometry, it is possible to calculate the robot's displacement in the environment and compensate for this displacement in the identified objects' position. As a result, the equipment proved to be capable of executing the technique without using much CPU processing, about 53.4 %. This work fulfilled the thesis's objective to develop a new source of perception for mobile robots, where intelligent information is provided, not just depth data.

This thesis addressed the study of sources of perception in mobile robots, starting with a preliminary study on traditional sensing strategies, such as the use of sensors to avoid obstacles and map obstacles, ending with the proposal of a new source of perception of the environment, proposing a new sensor capable of identifying objects and their spatial position, in addition to predicting their displacement in the environment. In this way, this thesis carried out a study on sensing techniques for mobile robots, elaborated a proposal for a new sensing approach, and developed and validated the strategy on embedded hardware, leading the author to browse through several science areas.

7.1 FUTURE WORK

Based on the studies developed in this thesis, it was possible to identify some points that can be improved in the future, both about the works presented here, as well as new works to be developed. Future works are presented in the list below.

1. Adding cargo to the flew in formation with Drones, and experiments in real environments. The work showed that it is possible to carry out cargo transport with multiple drones. However, the effect of cargo on the flight was not discussed. Real experiments need to be done to validate the strategy in a real environment;

2. Addition of the DeepSpatial sensor to the leading Drone during the flight. In this way, the obstacle avoidance maneuver can be performed beforehand, using the prediction of the identified obstacles' position;

3. Enhancement of the DeepSpatial sensor to be used in the inspection environment. The sensor currently identifies only objects through the YoLo network. To use the sensor in the inspection environment, the sensor can be adapted to identify critical environmental components, such as tubes and weld beads. This information can be used for location and other inspection actions;

4. The improvement of the object tracking strategy present in the DeepSpatial sensor, in its latest version, the strategy consists of verifying whether the objects belong to the same class, and if they belong, the Euclidean distance between them is calculated. Deep learning tracking algorithms can be used. However, it is necessary to assess whether these techniques do not impair the performance of the sensor;

5. The creation of a semantic map, where the position of the objects identified by the DeepSpatial sensor is used to develop an intelligent mapping strategy. The position of each object will be saved, and its movements will also be calculated continuously if the object is in motion;

6. Use and adapt the sensing techniques developed for mobile robots in other environments, such as industries and smart cities.

REFERENCES

ABDALLA, T. Y.; ABED, A. A.; AHMED, A. A. Mobile robot navigation using pso-optimized fuzzy artificial potential field with fuzzy control. **Journal of Intelligent & Fuzzy Systems**, IOS Press, v. 32, n. 6, p. 3893–3908, 2017.

ABIYEV, R. H.; GÜNSEL, I. S.; AKKAYA, N.; AYTAC, E.; ÇAĞMAN, A.; ABIZADA, S. Fuzzy control of omnidirectional robot. **Procedia computer science**, Elsevier, v. 120, p. 608–616, 2017.

ALONSO-MORA, J.; MONTIJANO, E.; NÄGELI, T.; HILLIGES, O.; SCHWAGER, M.; RUS, D. Distributed multi-robot formation control in dynamic environments. **Autonomous Robots**, Springer, v. 43, n. 5, p. 1079–1100, 2019.

BACIK, J.; DUROVSKY, F.; FEDOR, P.; PERDUKOVA, D. Autonomous flying with quadrocopter using fuzzy control and aruco markers. **Intelligent Service Robotics**, Springer, v. 10, n. 3, p. 185–194, 2017.

BOULCH, A.; SAUX, B.; AUDEBERT, N. Unstructured point cloud semantic labeling using deep segmentation networks. **3DOR**, v. 2, p. 7, 2017.

CAO, Y. U.; FUKUNAGA, A. S.; KAHNG, A. Cooperative mobile robotics: Antecedents and directions. **Autonomous robots**, Springer, v. 4, n. 1, p. 7–27, 1997.

CARULLO, A.; PARVIS, M. An ultrasonic sensor for distance measurement in automotive applications. **IEEE Sensors journal**, v. 1, n. 2, p. 143, 2001.

CHEN, C.; JAFARI, R.; KEHTARNAVAZ, N. A survey of depth and inertial sensor fusion for human action recognition. **Multimedia Tools and Applications**, Springer, v. 76, n. 3, p. 4405–4425, 2017.

DUDEK, G.; JENKIN, M. **Computational principles of mobile robotics**. : Cambridge university press, 2010.

ELFES, A. Using occupancy grids for mobile robot perception and navigation. **Computer**, IEEE, v. 22, n. 6, p. 46–57, 1989.

FORSTER, C.; ZHANG, Z.; GASSNER, M.; WERLBERGER, M.; SCARAMUZZA, D. Svo: Semidirect visual odometry for monocular and multicamera systems. **IEEE Transactions on Robotics**, IEEE, v. 33, n. 2, p. 249–265, 2016.

GHORPADE, D.; THAKARE, A. D.; DOIPHODE, S. Obstacle detection and avoidance algorithm for autonomous mobile robot using 2d lidar. *In*: **2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)**. 2017. p. 1–6.

GIORDANO, F.; MATTEI, G.; PARENTE, C.; PELUSO, F.; SANTAMARIA, R. Integrating sensors into a marine drone for bathymetric 3d surveys in shallow waters. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 16, n. 1, p. 41, 2016.

HAGEBEUKER, D. B.; MARKETING, P. A 3d time of flight camera for object detection. **PMD Technologies GmbH, Siegen**, 2007.

HAYATI, S. Hybrid position/force control of multi-arm cooperating robots. *In*: IEEE. **Proceedings. 1986 IEEE International Conference on Robotics and Automation**. 1986. v. 3, p. 82–89.

HORNUNG, A.; WURM, K. M.; BENNEWITZ, M.; STACHNISS, C.; BURGARD, W. Octomap: An efficient probabilistic 3d mapping framework based on octrees. **Autonomous robots**, Springer, v. 34, n. 3, p. 189–206, 2013.

HU, H.; GU, J.; ZHANG, Z.; DAI, J.; WEI, Y. Relation networks for object detection. *In*: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. 2018. p. 3588–3597.

HUANG, A. S.; BACHRACH, A.; HENRY, P.; KRAININ, M.; MATURANA, D.; FOX, D.; ROY, N. Visual odometry and mapping for autonomous flight using an rgb-d camera. *In*: **Robotics Research**. : Springer, 2017. p. 235–252.

HUANG, H. Fusion of modified bat algorithm soft computing and dynamic model hard computing to online self-adaptive fuzzy control of autonomous mobile robots. **IEEE Transactions on Industrial Informatics**, IEEE, v. 12, n. 3, p. 972–979, 2016.

JAFARI, O. H.; MITZEL, D.; LEIBE, B. Real-time rgb-d based people detection and tracking for mobile robots and head-worn cameras. *In*: IEEE. **2014 IEEE international conference on robotics and automation (ICRA)**. 2014. p. 5636–5643.

KNOOP, S.; VACEK, S.; DILLMANN, R. Sensor fusion for 3d human body tracking with an articulated 3d body model. *In*: IEEE. **Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.** 2006. p. 1686–1691.

KONOLIGE, K.; AUGENBRAUN, J.; DONALDSON, N.; FIEBIG, C.; SHAH, P. A low-cost laser distance sensor. *In*: IEEE. **2008 IEEE international conference on robotics and automation**. 2008. p. 3002–3008.

LI, G.; ST-ONGE, D.; PINCIROLI, C.; GASPARRI, A.; GARONE, E.; BELTRAME, G. Decentralized progressive shape formation with robot swarms. **Autonomous Robots**, Springer, v. 43, n. 6, p. 1505–1521, 2019.

LIANG, M.; YANG, B.; WANG, S.; URTASUN, R. Deep continuous fusion for multi-sensor 3d object detection. *In*: **Proceedings of the European Conference on Computer Vision (ECCV)**. 2018. p. 641–656.

LIU, L.; OUYANG, W.; WANG, X.; FIEGUTH, P.; CHEN, J.; LIU, X.; PIETIKÄINEN, M. Deep learning for generic object detection: A survey. **International journal of computer vision**, Springer, v. 128, n. 2, p. 261–318, 2020.

MEHREZ, M. W.; SPRODOWSKI, T.; WORTHMANN, K.; MANN, G. K.; GOSINE, R. G.; SAGAWA, J. K.; PANNEK, J. Occupancy grid based distributed mpc for mobile robots. *In*: IEEE. **2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. 2017. p. 4842–4847.

MOHANTY, V.; AGRAWAL, S.; DATTA, S.; GHOSH, A.; SHARMA, V. D.; CHAKRAVARTY, D. Deepvo: A deep learning approach for monocular visual odometry. **arXiv preprint arXiv:1611.06069**, 2016.

MOKHTARI, G.; ZHANG, Q.; HARGRAVE, C.; RALSTON, J. C. Non-wearable uwb sensor for human identification in smart home. **IEEE Sensors Journal**, v. 17, n. 11, p. 3332–3340, 2017.

MUEGGLER, E.; GALLEGO, G.; REBECQ, H.; SCARAMUZZA, D. Continuous-time visual-inertial odometry for event cameras. **IEEE Transactions on Robotics**, IEEE, v. 34, n. 6, p. 1425–1440, 2018.

MUEGGLER, E.; REBECQ, H.; GALLEGO, G.; DELBRUCK, T.; SCARAMUZZA, D. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. **The International Journal of Robotics Research**, SAGE Publications Sage UK: London, England, v. 36, n. 2, p. 142–149, 2017.

NEHMZOW, U. **Mobile robotics: a practical introduction**. : Springer Science & Business Media, 2012.

PALACÍN, J.; MARTÍNEZ, D.; RUBIES, E.; CLOTET, E. Mobile robot self-localization with 2d push-broom lidar in a 2d map. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 20, n. 9, p. 2500, 2020.

PARKER, L. E.; RUS, D.; SUKHATME, G. S. Multiple mobile robot systems. *In*: **Springer Handbook of Robotics**. : Springer, 2016. p. 1335–1384.

PEŁKA, M.; MAJEK, K.; RATAJCZAK, J.; BĘDKOWSKI, J.; MASŁOWSKI, A. Study of multi-sensor fusion for localization. *In*: IEEE. **2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)**. 2019. p. 1–2.

POLLARD, B.; TALLAPRAGADA, P. An aquatic robot propelled by an internal rotor. **IEEE/ASME Transactions on Mechatronics**, IEEE, v. 22, n. 2, p. 931–939, 2016.

REDMON, J..; FARHADI, A. Yolo9000: better, faster, stronger. *In*: **Proceedings of the IEEE conference on computer vision and pattern recognition**. 2017. p. 7263–7271.

REDMON, J.; FARHADI, A. Yolov3: An incremental improvement. **arXiv**, 2018.

ROCCHINI, C.; CIGNONI, P.; MONTANI, C.; PINGI, P.; SCOPIGNO, R. A low cost 3d scanner based on structured light. *In*: WILEY ONLINE LIBRARY. **Computer Graphics Forum**. 2001. v. 20, n. 3, p. 299–308.

RONAO, C. A.; CHO, S. Human activity recognition with smartphone sensors using deep learning neural networks. **Expert systems with applications**, Elsevier, v. 59, p. 235–244, 2016.

RUSU, R. B.; COUSINS, S. 3d is here: Point cloud library (pcl). *In*: IEEE. **2011 IEEE international conference on robotics and automation**. 2011. p. 1–4.

RYLL, M.; MUSCIO, G.; PIERRI, F.; CATALDI, E.; ANTONELLI, G.; CACCAVALE, F.; FRANCHI, A. 6d physical interaction with a fully actuated aerial robot. *In*: IEEE. **2017 IEEE International Conference on Robotics and Automation (ICRA)**. 2017. p. 5190–5195.

SABATINI, R.; GARDI, A.; RICHARDSON, M. Lidar obstacle warning and avoidance system for unmanned aircraft. **International Journal of Mechanical, Aerospace, Industrial and Mechatronics Engineering**, World Academy of Science, Engineering and Technology, v. 8, n. 4, p. 718–729, 2014.

SIMON, M.; AMENDE, K.; KRAUS, A.; HONER, J.; SAMANN, T.; KAULBERSCH, H.; MILZ, S.; GROSS, M. H. Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds. *In*: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops**. 2019. p. 0–0.

SIMONY, Martin; MILZY, Stefan; AMENDEY, Karl; GROSS, Horst-Michael. Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds. *In*: **Proceedings of the European Conference on Computer Vision (ECCV)**. 2018. p. 0–0.

SUN, L.; YAN, Z.; ZAGANIDIS, A.; ZHAO, C.; DUCKETT, T. Recurrent-octomap: Learning state-based map refinement for long-term semantic mapping with 3-d-lidar data. **IEEE Robotics and Automation Letters**, IEEE, v. 3, n. 4, p. 3749–3756, 2018.

TAN, Y. H.; SIDDALL, R.; KOVAC, M. Efficient aerial–aquatic locomotion with a single propulsion system. **IEEE Robotics and Automation Letters**, IEEE, v. 2, n. 3, p. 1304–1311, 2017.

TEIXEIRA, M. A. S.; DALMEDICO, N.; SANTOS, H. B.; OLIVEIRA, A. S.; ARRUDA, L. V. R.; NEVES, F. Enhancing robot capabilities of environmental perception through embedded gpu. *In*: **2017 VII Brazilian Symposium on Computing Systems Engineering (SBESC)**. 2017. p. 217–224.

TEIXEIRA, M. A. S.; NEVES-JR, F.; KOUBAA, A.; ARRUDA, L. V. R. De; OLIVEIRA, A. S. De. A quadral-fuzzy control approach to flight formation by a fleet of unmanned aerial vehicles. **IEEE Access**, v. 8, p. 64366–64381, 2020.

TEIXEIRA, M. A. S.; NEVES-JR, F.; KOUBAA, A.; ARRUDA, L. V. R.; OLIVEIRA, A. S. *et al.* Deepspatial: Intelligent spatial sensor to perception of things. **IEEE Sensors Journal**, v. 21, n. 4, p. 3966–3976, 2021.

TEIXEIRA, M. A. S.; NOGUEIRA, R. C. M.; DALMEDICO, N.; SANTOS, H. B.; ARRUDA, L. V. R.; NEVES-JR, F.; PIPA, D. R.; RAMOS, J. E.; OLIVEIRA, A. S. Intelligent 3d perception system for semantic description and dynamic interaction. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 19, n. 17, p. 3764, 2019.

TEIXEIRA, M. A. S.; SANTOS, H. B.; DALMEDICO, N.; ARRUDA, L. V. R.; NEVES-JR, F.; OLIVEIRA, A. S. Intelligent environment recognition and prediction for ndt inspection through autonomous climbing robot. **Journal of Intelligent & Robotic Systems**, Springer, v. 92, n. 2, p. 323–342, 2018.

THRUN, S. Learning occupancy grid maps with forward sensor models. **Autonomous robots**, Springer, v. 15, n. 2, p. 111–127, 2003.

WANG, P.; LI, W.; OGUNBONA, P.; WAN, J.; ESCALERA, S. Rgb-d-based human motion recognition with deep learning: A survey. **Computer Vision and Image Understanding**, Elsevier, v. 171, p. 118–139, 2018.

WANG, Y.; SHAN, M.; YUE, Y.; WANG, D. Vision-based flexible leader-follower formation tracking of multiple nonholonomic mobile robots in unknown obstacle environments. **IEEE Transactions on Control Systems Technology**, IEEE, 2019.

WURM, K. M.; HORNUNG, A.; BENNEWITZ, M.; STACHNISS, C.; BURGARD, W. Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems. *In*: **Proc. of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation**. 2010. v. 2.

YANG, B.; LUO, W.; URTASUN, R. Pixor: Real-time 3d object detection from point clouds. *In*: **Proceedings of the IEEE conference on Computer Vision and Pattern Recognition**. 2018. p. 7652–7660.

YEN, C.; CHENG, M. A study of fuzzy control with ant colony algorithm used in mobile robot for shortest path planning and obstacle avoidance. **Microsystem Technologies**, Springer, v. 24, n. 1, p. 125–135, 2018.

YU, H.; SHI, P.; LIM, C.; WANG, D. Formation control for multi-robot systems with collision avoidance. **International Journal of Control**, Taylor & Francis, v. 92, n. 10, p. 2223–2234, 2019.

ZHANG, Z.; REBECQ, H.; FORSTER, C.; SCARAMUZZA, D. Benefit of large field-of-view cameras for visual odometry. *In*: IEEE. **2016 IEEE International Conference on Robotics and Automation (ICRA)**. 2016. p. 801–808.

ZHAO, Z.; ZHENG, P.; XU, S.; WU, X. Object detection with deep learning: A review. **IEEE transactions on neural networks and learning systems**, IEEE, v. 30, n. 11, p. 3212–3232, 2019.

APPENDIX A – LICENSE REFERRING TO THE PAPER *INTELLIGENT ENVIRONMENT RECOGNITION AND PREDICTION FOR NDT INSPECTION THROUGH AUTONOMOUS CLIMBING ROBOT*

SPRINGER NATURE LICENSE TERMS AND CONDITIONS

Feb 02, 2021

This Agreement between UTFPR -- MARCO ANTONIO TEIXEIRA ("You") and Springer Nature ("Springer Nature") consists of your license details and the terms and conditions provided by Springer Nature and Copyright Clearance Center.

License Number	4999360302934
License date	Jan 31, 2021
Licensed Content Publisher	Springer Nature
Licensed Content Publication	Journal of Intelligent and Robotic Systems
Licensed Content Title	Intelligent environment recognition and prediction for NDT inspection through autonomous climbing robot
Licensed Content Author	Marco Antonio Simoes Teixeira et al
Licensed Content Date	Jan 19, 2018
Type of Use	Thesis/Dissertation
Requestor type	academic/university or research institute
Format	print and electronic
Portion	full article/chapter
Will you be translating?	no
Circulation/distribution	50000 or greater
Author of this Springer Nature content	yes
Title	MOBILE ROBOTS: A STUDY ON SENSING AND PERCEPTION SYSTEMS
Institution name	Universidade Tecnológica Federal do Paraná (UTFPR) - Brazil
Expected presentation date	Jan 2021
Requestor Location	Marco A. S. Teixeira Av. Sete de Setembro, 3165 - Rebouças Ap 2102 Curitiba, other 80230-901 Brazil Attn: Universidade Tecnológica Federal do Paraná
Total	**0.00 USD**
Terms and Conditions	

Springer Nature Customer Service Centre GmbH
Terms and Conditions

This agreement sets out the terms and conditions of the licence (the **Licence**) between you and **Springer Nature Customer Service Centre GmbH** (the **Licensor**). By clicking 'accept' and completing the transaction for the material (**Licensed Material**), you also confirm your acceptance of these terms and conditions.

1. Grant of License

1. 1. The Licensor grants you a personal, non-exclusive, non-transferable, world-wide licence to reproduce the Licensed Material for the purpose specified in your order only. Licences are granted for the specific use requested in the order and for no other use, subject to the conditions below.

1. 2. The Licensor warrants that it has, to the best of its knowledge, the rights to license reuse of the Licensed Material. However, you should ensure that the material you are requesting is original to the Licensor and does not carry the copyright

of another entity (as credited in the published version).

1. 3. If the credit line on any part of the material you have requested indicates that it was reprinted or adapted with permission from another source, then you should also seek permission from that source to reuse the material.

2. Scope of Licence

2. 1. You may only use the Licensed Content in the manner and to the extent permitted by these Ts&Cs and any applicable laws.

2. 2. A separate licence may be required for any additional use of the Licensed Material, e.g. where a licence has been purchased for print only use, separate permission must be obtained for electronic re-use. Similarly, a licence is only valid in the language selected and does not apply for editions in other languages unless additional translation rights have been granted separately in the licence. Any content owned by third parties are expressly excluded from the licence.

2. 3. Similarly, rights for additional components such as custom editions and derivatives require additional permission and may be subject to an additional fee. Please apply to Journalpermissions@springernature.com/bookpermissions@springernature.com for these rights.

2. 4. Where permission has been granted **free of charge** for material in print, permission may also be granted for any electronic version of that work, provided that the material is incidental to your work as a whole and that the electronic version is essentially equivalent to, or substitutes for, the print version.

2. 5. An alternative scope of licence may apply to signatories of the STM Permissions Guidelines, as amended from time to time.

3. Duration of Licence

3. 1. A licence for is valid from the date of purchase ('Licence Date') at the end of the relevant period in the below table:

Scope of Licence	Duration of Licence
Post on a website	12 months
Presentations	12 months
Books and journals	Lifetime of the edition in the language purchased

4. Acknowledgement

4. 1. The Licensor's permission must be acknowledged next to the Licenced Material in print. In electronic form, this acknowledgement must be visible at the same time as the figures/tables/illustrations or abstract, and must be hyperlinked to the journal/book's homepage. Our required acknowledgement format is in the Appendix below.

5. Restrictions on use

5. 1. Use of the Licensed Material may be permitted for incidental promotional use and minor editing privileges e.g. minor adaptations of single figures, changes of format, colour and/or style where the adaptation is credited as set out in Appendix 1 below. Any other changes including but not limited to, cropping, adapting, omitting material that affect the meaning, intention or moral rights of the author are strictly prohibited.

5. 2. You must not use any Licensed Material as part of any design or trademark.

5. 3. Licensed Material may be used in Open Access Publications (OAP) before publication by Springer Nature, but any Licensed Material must be removed from OAP sites prior to final publication.

6. Ownership of Rights

6. 1. Licensed Material remains the property of either Licensor or the relevant third party and any rights not explicitly granted herein are expressly reserved.

7. Warranty

IN NO EVENT SHALL LICENSOR BE LIABLE TO YOU OR ANY OTHER PARTY OR ANY OTHER PERSON OR FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL OR INDIRECT DAMAGES, HOWEVER CAUSED, ARISING OUT OF OR IN CONNECTION WITH THE DOWNLOADING, VIEWING OR USE OF THE MATERIALS REGARDLESS OF THE FORM OF ACTION, WHETHER FOR BREACH OF CONTRACT, BREACH OF WARRANTY, TORT, NEGLIGENCE, INFRINGEMENT OR OTHERWISE (INCLUDING, WITHOUT LIMITATION, DAMAGES BASED ON LOSS OF PROFITS, DATA, FILES, USE, BUSINESS OPPORTUNITY OR CLAIMS OF THIRD PARTIES), AND
WHETHER OR NOT THE PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION SHALL APPLY NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY PROVIDED HEREIN.

8. Limitations

8. 1. ***BOOKS ONLY:***Where **'reuse in a dissertation/thesis'** has been selected the following terms apply: Print rights of the final author's accepted manuscript (for clarity, NOT the published version) for up to 100 copies, electronic rights for use only on a personal website or institutional repository as defined by the Sherpa guideline (www.sherpa.ac.uk/romeo/).

8. 2. For content reuse requests that qualify for permission under the STM Permissions Guidelines, which may be updated from time to time, the STM Permissions Guidelines supersede the terms and conditions contained in this licence.

9. Termination and Cancellation

9. 1. Licences will expire after the period shown in Clause 3 (above).

9. 2. Licensee reserves the right to terminate the Licence in the event that payment is not received in full or if there has been a breach of this agreement by you.

Appendix 1 — Acknowledgements:

For Journal Content:
Reprinted by permission from [**the Licensor**]: [**Journal Publisher** (e.g. Nature/Springer/Palgrave)] [**JOURNAL NAME**] [**REFERENCE CITATION** (Article name, Author(s) Name), [**COPYRIGHT**] (year of publication)

For **Advance Online Publication papers:**
Reprinted by permission from [**the Licensor**]: [**Journal Publisher** (e.g. Nature/Springer/Palgrave)] [**JOURNAL NAME**] [**REFERENCE CITATION** (Article name, Author(s) Name), [**COPYRIGHT**] (year of publication), advance online publication, day month year (doi: 10.1038/sj.[JOURNAL ACRONYM].)

For Adaptations/Translations:
Adapted/Translated by permission from [**the Licensor**]: [**Journal Publisher** (e.g. Nature/Springer/Palgrave)] [**JOURNAL NAME**] [**REFERENCE CITATION** (Article name, Author(s) Name), [**COPYRIGHT**] (year of publication)

Note: For any republication from the British Journal of Cancer, the following credit line style applies:

Reprinted/adapted/translated by permission from [**the Licensor**]: on behalf of Cancer Research UK: : [**Journal Publisher** (e.g. Nature/Springer/Palgrave)] [**JOURNAL NAME**] [**REFERENCE CITATION** (Article name, Author(s) Name), [**COPYRIGHT]** (year of publication)

For **Advance Online Publication** papers:
Reprinted by permission from The [**the Licensor**]: on behalf of Cancer Research UK: [**Journal Publisher** (e.g. Nature/Springer/Palgrave)] [**JOURNAL NAME**] [**REFERENCE CITATION** (Article name, Author(s) Name), [**COPYRIGHT**] (year of publication), advance online publication, day month year (doi: 10.1038/sj.[JOURNAL ACRONYM])

For Book content:
Reprinted/adapted by permission from [**the Licensor**]: [**Book Publisher** (e.g. Palgrave Macmillan, Springer etc) [**Book Title**] by [**Book author**(s)] [**COPYRIGHT**] (year of publication)

Other Conditions:

Version 1.3

Questions? customercare@copyright.com or +1-855-239-3415 (toll free in the US) or +1-978-646-2777.

APPENDIX B – LICENSE REFERRING TO THE PAPER *DEEPSPATIAL: INTELLIGENT SPATIAL SENSOR TO PERCEPTION OF THINGS*