Vinicius de Mello Lima

Obstacle Detection and Avoidance System for UAV's, based on neuro-fuzzy controller

DISSERTAÇÃO DE MESTRADO

DEPARTAMENTO DE ENGENHARIA ELÉTRICA
Programa de Pós-graduação em Engenharia Elétrica

Rio de Janeiro
September 2018

Pontifícia Universidade Católica
do Rio de Janeiro

Vinicius de Mello Lima

Obstacle Detection and Avoidance System for UAV's, based on neuro-fuzzy controller

Dissertação de Mestrado

Dissertation presented to the Programa de Pós–graduação em Engenharia Elétrica of PUC–Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Elétrica.

Advisor: Prof. Eduardo Costa e Silva

Rio de Janeiro
September 2018

Pontifícia Universidade Católica
do Rio de Janeiro

Vinicius de Mello Lima

Obstacle Detection and Avoidance System for UAV's, based on neuro-fuzzy controller

Dissertation presented to the Programa de Pós–graduação em Engenharia Elétrica of PUC–Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Elétrica. Approved by the undersigned Examination Committee.

Prof. Eduardo Costa e Silva
Advisor
Departamento de Engenharia Elétrica PUC-Rio

Prof. Carlos Roberto Hall Barbosa
Programa de Pós-Graduação em Metrologia – PUC-Rio

Prof. Leonardo Alfredo Forero Mendoza
UERJ

Prof. Karla Tereza Figueiredo Leite
UERJ

Prof. Márcio da Silveira Carvalho
Vice Dean of Graduate Studies
Centro Técnico Científico – Puc-Rio

Rio de Janeiro, September 13th, 2018

Vinicius de Mello Lima

Graduated in Electrical Engineering from Pontifical Catholic University of Rio de Janeiro at 2015. Scientific Initiation at Optoelectronic Lab's - CETUC developing electronic circuits and PCB's. Worked at TV Globo S.A. for nine years, Brazilian Navy for three years and actually at VML Technology. Interested in signal processing, artificial intelligence, UAV, optoelectronic, power converters, robotics.

Bibliographic data

Lima, Vinicius de Mello

Obstacle Detection and Avoidance System for UAV's, based on neuro-fuzzy controller / Vinicius de Mello Lima; advisor: Eduardo Costa e Silva. – 2018.
75 f: il. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica, 2018.
Inclui bibliografia

1. Engenharia Elétrica – Teses;. 2. Sistema de detecção e desvio de obstáculos;. 3. VANT;. 4. Controlador neuro-fuzzy;. 5. SONAR;. 6. LIDAR. I. Silva, Eduardo Costa. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. III. Título.

CDD: 621.3

Acknowledgments

To God that has given me strength to carry out this work.

To my advisor Prof. Eduardo Costa e Silva for believing me and in this work, for the time spent and partnership.

To my family and to my love for the support, caress, patience and encouragement at all times.

To my friends for the companionship, help and ideas.

“This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001”

Abstract

Lima, Vinicius de Mello; Silva, Eduardo Costa (Advisor). **Obstacle Detection and Avoidance System for UAV's, based on neuro-fuzzy controller**. Rio de Janeiro, 2018. 75p. Dissertação de mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

This dissertation presents the design and development of an obstacle detection and avoidance system for unmanned aerial vehicles, implemented by a neuro-fuzzy controller. In this context, this work presents a theoretical review of unmanned aerial vehicles, the applicable Brazilian legislation, obstacle detection methods, fuzzy logic and neural networks. The developed controller was implemented in order to mimic the actions taken by a human operator, aiming at avoiding obstacles found in the navigation path of the UAV. Inference rules were established based on consultation with specialists in the field and the weights adjusted by neural networks. The decision-making process takes into account information collected by a multichannel Lidar and ultrasonic sensors embedded in the UAV. In turn, the developed algorithm was embedded in a commercial flight controller. The complete quadricopter system is detailed, highlighting the key features of all sensors and the flight controller. The results of computational simulations and experimental tests are presented, discussed and compared, in order to evaluate the performance of the developed system.

Keywords

Electrical Engineering – Thesis; Obstacle Detection and Avoidance System; UAV; Neuro-Fuzzy Controller; SONAR; LIDAR;

Resumo

Lima, Vinicius de Mello; Silva, Eduardo Costa. **Sistema de Detecção e Desvio de Obstáculos para VANTs, baseado em Controlador Neuro-Fuzzy**. Rio de Janeiro, 2018. 75p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Esta dissertação apresenta o projeto e desenvolvimento de um sistema para detecção e desvio de obstáculos para veículos aéreos não tripulados (VANTs), implementado por um controlador neuro-fuzzy. Neste contexto, este trabalho apresenta uma revisão teórica sobre veículos aéreos não tripuláveis, legislação brasileira aplicável, métodos de detecção de obstáculos, lógica nebulosa e redes neurais. O controlador desenvolvido foi implementado de forma a imitar as ações realizadas por um operador humano, visando desviar de obstáculos encontrados no caminho de navegação do VANT. Regras de inferência são estabelecidas baseadas na consultoria de especialistas da área e os pesos ajustados pela rede neural. O processo de tomada de decisão ocorre levando em consideração as informações coletadas por um Lidar multicanal e sensores ultrassônicos embarcados no VANT. Por sua vez, o algoritmo desenvolvido foi incorporado em um controlador de vôo comercial. O sistema completo do quadricóptero é detalhado, destacando as principais características de todos os sensores e do controlador de vôo. Os resultados das simulações computacionais e testes experimentais são apresentados, discutidos e comparados, a fim de avaliar o desempenho do sistema desenvolvido.

Palavras-chave

Engenharia Elétrica – Teses; Sistema de detecção e desvio de obstáculos; VANT; Controlador neuro-fuzzy; SONAR; LIDAR

Table of contents

List of figures

List of tables

List of Abbreviations

ANAC	*Agência Nacional de Aviação Civil*
ANATEL	*Agência Nacional de Telecomunicações*
ANFIS	Adaptive Neuro-fuzzy Inference System
BP	Back-propagation
BRLOS	Beyond Radio Line of Sight
BVLOS	Beyond Visual Line of Sight
CAN	Controller Area Network
CANFIS	Coactive Neuro-fuzzy Inference System
CPU	Central Processing Unit
DAS	Detection and Avoidance System
DECEA	*Departamento de Controle do Espaço Aéreo*
DJI	Dà-Jiãng Innovations Science and Technology Co.
DSM	Digital Spectrum Modulation
ESD	Electrostatic Discharge
EVLOS	Extended Visual Line of Sight
FIS	Fuzzy Inference System
FMU	Flight Management Unit
FPU	Floating Ploint Unit
GPS	Global Positioning System
HDPR	Horizontal direct passive ranging
I/O	Input-Output
I2C	Inter-Integrated Circuit
INS	Inertial Navigation System
LED	Light Emitting Diode
LEDDAR	Light-emitting Diode Detection and Ranging

LIDAR	Light Detection and Ranging
LMS	Least-mean-square
PPM	Pulse Position Modulation
RAM	Random Access Memory
RBAC-E	*Regulamento Brasileiro da Aviação Civil Especial*
RLOS	Radio Line of Sight
RPA	Remotely Piloted Aircrafts
RPAS	Remotely Piloted Aircrafts Systems
RSSI	Received Signal Strength Indicator
RTOS	Real-time Operating System
SARPAS	*Sistema de Solicitação de Acesso ao Espaço Aéreo por RPAS*
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver-transmitter
UAV	Unmanned Aerial Vehicle
USB	Universal Serial Bus
VDPR	Vertical Direct Passive Ranging
VLOS	Visual Line of Sight

A robot must protect its own existence....

Isaac Asimov, *Runaround, 1942.*

1
Introduction

The technological development that has occurred over the last years in electronics, solid state devices, microprocessors, data storage and sensors, led to a significant increase of alternative solutions based on the use of UAVs (Unmanned Aerial Vehicle) in several activities: military, security, engineering inspections, filming, recreation, etc. This market trend points to large-scale use of UAVs, opening space for the development of research associated with performance optimization of these systems. In particular, flight safety is an extremely relevant aspect in this regard, since it contributes to the enhancement of the reliability of UAV based systems.

The costs and size of electronics are continuously decreasing, allowing to reach levels of innovation and discovery not yet foreseen. However, cultural and regulatory restrictions have been slowing down the large scale use of UAVs, so a lot has to be done in order to break paradigms and educate people about its use. In EUA, two areas stand out: public safety and precision agriculture. In Brazil, the use of UAVs in precision agriculture has shown a large increase in the last years. It is expected that research on new materials, propulsion systems, structural constructs and electronics will lead to even further technological breakthroughs on the next decades [1].

1.1
Context

The present work aims at contributing to the development of an auxiliary pilot system capable of safely identifing and avoiding obstacles [2]. Nowadays, in most cases, obstacle avoidance operations are carried out by a human pilot; and just a few commercial systems for detection and avoidance are already in use. Among them, the Phantom 4 (DJI) embedded controller can be highlighted, which is based on computer vision techniques. In this context, the present work focus at the development of an autonomous detection and avoidance system, using lidar and sonar sensors for detection and a neuro-fuzzy algorithm to define how to perform the obstacle avoidance [3][4].

1.2 Unmanned Aerial Vehicle

1.2.1 Definition

Unmanned Aerial Vehicles (UAV) are aircrafts that do not need a pilot on board. They are systems composed by several sub-systems, such as: aircraft, payload, control station, launch and recovery system (where applicable), communication and others [5]. Their first applications were on the military sector, followed by scientific and commercial applications. Their development and operation had rapidly expanded in the last 30 years. UAV can be divided in two groups: remotely piloted aircrafts (RPA) and autonomous aircrafts [6]; and UAV regulations address the issue differently for each case. For civil applications, RPAs are more commonly used and, on the other hand, autonomous aircrafts are widely used in military context. The range of applications spread across agriculture, geosciences, disaster management, industrial, photography, surveillance and others. It is important to mention that all UAV sub-systems are an integral part of the UAV system and they have equal importance. Figure 1.1 illustrates the integration of each major UAV sub-system, that are herein briefly explained:

- **Control Station**: could be based on the ground, on a ship or also in another aircraft. It is the control center of the operation and man-machine interface. From there, all commands and missions are uploaded through a communication system to the aircraft, and it receives all the information sent from the aircraft.
- **Payload**: the load carried by an aircraft in order to achieve an specific mission. Some sophisticated UAVs could carry different sensors as video and thermal camera, radar, lidar, and others.
- **Air vehicle**: The mission characteristics settle the required type and performance of the air vehicle. The principal task of the aircraft is to carry the mission payload and all required subsystems for the operation. Other important requirements are, but not limited to, airspeed, endurance, fuel and speed. Three designs are common: fixed-wing, rotary-wing and convertible aircraft.
- **Navigation System**: It continuously informs to the operator the position of the aircraft and vice-versa. It is essential for autonomous operation, and commonly based on inertial navigation system (INS) and global positioning system (GPS). Nowadays, other methods are also available as radio tracking and direct reckoning.

- **Launch, Recovery and Retrieval Equipment**: Launch equipment is required when the aircraft does not have vertical flight capability or access to a suitable runway. Conventionally, it is a ramp used to accelerate the aircraft on a trolley until it reaches sufficient airspeed to sustain airborne flight. Besides, a parachute, suitable landing zone or other method are required to bring back the aircraft. Additionally, heavy aircrafts may require other equipments for transportation and launching.
- **Communications**: The main function of a communication system is to provide data links between the control station and the aircraft. The transmission is usually done by radio frequency, but other alternatives could be used as laser beam and optical fibers. The communication system is responsible for transmitting: control commands to the aircraft and mounted payload; updated position to control station and vice versa; aircraft telemetry. The complexity, cost and power demand of the communication system is defined by the required range of operation, data security and bandwidth.
- **Interfaces**: Although some systems could operate as stand-alone, other have to read and write information from/to several subsystems. They have to use compatible protocols in order to ensure an adequate data path throughout the entire system.
- **Support equipment and transportation**: The mission planning should consider the required support equipment, such as tools, spare parts, manuals, etc.; because neglecting these aspects could compromise the mission. Besides, some missions require transportation equipment to: move the aircraft to specific launching zones, position control stations and recovery equipment and others.

Usually, the design of aircraft-based systems consists in three main stages: **conceptual phase**, **preliminary design** and **detailed design phase**.

At conceptual phase, market studies should be done in order to analyse the commercial viability of the project, such as operational costs, size of the market, regulatory issues, etc. During this phase, tests on aircraft models are performed to confirm theoretical aerodynamic calculations and to identify any inconsistencies. The obtained results are evaluated in order to define if the original conceptual design will proceed or not to the next phase. The preliminary design comes afterwards, where optimisations to increase overall performance of the system are made and normally an aircraft mock-up is manufactured to get a better idea about components assembling and maintenance of the system. These studies lead to a comprehensive definition

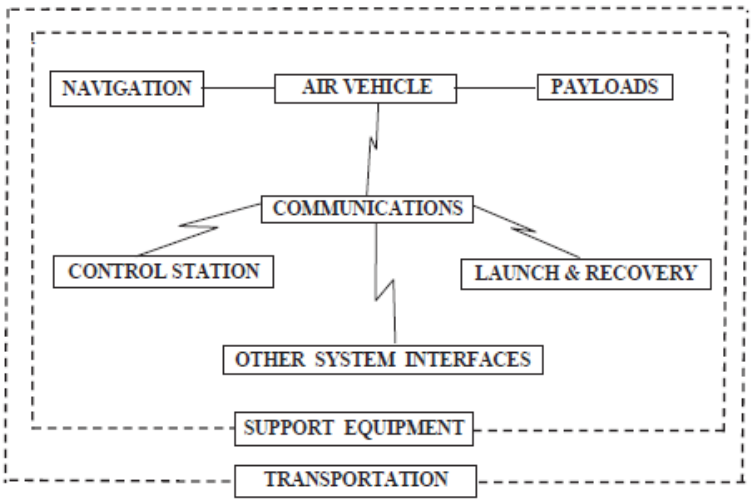

Figure 1.1: Structure of UAV System [5].

and specification of the system. The detailed design is the final stage of the process, being responsible for the detailed analysis of mechanical and electronics issues, production requirements, test equipments and operating / maintenance manuals.

1.2.2 Airframes

Due to the reduced risk compared with crewed aircraft, many different airframe configurations are available for UAVs. For convenience, they are grouped into three types, according to their take-off and landing methods:

- **HTOL**: horizontal take-off and landing;
- **VTOL**: vertical take-off and landing
- **Hybrid**: combine attributes of HTOL and VTOL.

HTOL Basically, they can be divided in three different types according to lift/mass balance, stability and control: tailplane-aft, tailplane forward or tailless. Most configurations have the power-plant at the rear of the fuselage, and the payload on the front with unobstructed view forward [5]. Figure 1.3 gives an example of this kind of configuration.

VTOL They are rotary-wing aircrafts, having a considerably more complex aerodynamics than fixed-wing aircrafts. Their lift is based on the principle of receiving air from above and accelerating it downwards. Many models could be seen in literature [5, 8]: Single rotor, Co-axial Rotor, Tandem Rotor, Tri-Rotor, Quad-Rotor, Hex-rotor and others. All of them have advantages and

Figure 1.2: Praxis Solar HTOL [7].

disadvantages and use complicated control algorithms. Figure 1.3 gives an example of this kind of configuration.

Figure 1.3: 3DR Solo [9].

Hybrid For long-range missions, HTOL are preferred, because they have longer flight autonomies and reach higher speeds, covering larger areas in less time. But the ability to take-off and land vertically and hovering are precious assets of VTOLs; especially under conditions of limited space to initiate and finish the mission. Hybrids find their space in this gap. Many types of hybrid aircrafts are presented in literature, as: Convertible rotor, tilt-wing-body, ducted fan and jet-life. Figure 1.4 gives an example of this kind of configuration.

1.2.3 Legislation

Technological improvements, increase in operational capabilities and more accessible prices made a "boom" on sales and use of UAV's. Coming with that, the necessity of regulations in order to guarantee public safety,

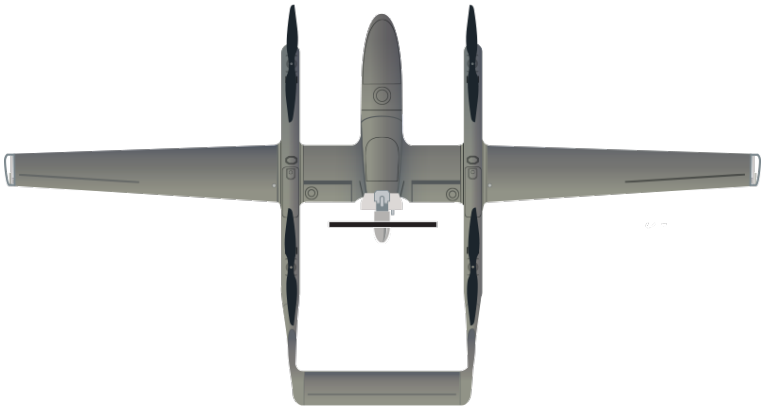

Figure 1.4: Latitude HQ-90B [10].

data protection and privacy takes place [6]. These factors, associated with legal, cultural and social landscapes, pose some impediments to the advance of this technology. At this part, a brief explanation about Brazilian UAV Legislation appears necessary. Basically, three organizations are responsible for UAV regulation: National Telecommunication Agency - ANATEL, National Civil Aviation Agency - ANAC, and Air Space Control Department - DECEA. Their area of actuation and main responsibility are briefly described:

ANATEL It was created on 1997 with the main goal to regulate telecommunication services in Brazil. Its area of actuation could be divided in four subgroups: customer service, certification and homologation of telecommunications products, regulatory authorization and competition solution[11]. A brief explanation of each subgroups comes next:

- Customer Service: makes possible the consumer to register complaints against telecommunications operators (mobile phone services, internet providers and others), and also against ANATEL services.
- Certification and Homologation of Telecommunications Products: analyses if telecommunication equipments are compliant with Brazilian standards of operation and identify them with a stamp.
- Regulatory Authorization: authorizes the use of radio frequencies and gives operation license for telecommunications stations.
- Competition Solution: deals with administrative solutions for companies or economic groups. These solutions are divided in three areas: prior consent,contractual homologation and conflicts resolution.

Once UAVs use data-link, remote monitoring, remote operation, video link and others, all of these RF-transmitters have to be certificated and homologated for operation in Brazil. This process aims to control the use of spectrum, level of interference between channels and to verify if the level of power irradiated is according to standards.

By resolution 242[12], ANATEL specifies all the rules and procedures to certificate and homologate products for telecommunications. All procedures are done using MOSAICO platform (specific system accessed through ANATEL website).

ANAC The Brazilian National Civil Aviation Agency is responsible by developing standards, certifing companies, promoting workshops, schools, supervising civil aviation professionals, airfields and airports, and overseeing the operations of aircraft, airlines, airports and professionals in the industry, focusing on safety and the quality of air transport[13]. The normative RBAC-E $n°94$, specifies the general rules, under ANAC competence, for RPAS . This normative is composed by eight sub-chapters and discusses topics such as:

- E94.1 - Applicability;
- E94.3 - Definitions;
- E94.5 - RPAS and RPA classification;
- E94.7 - Responsibilities of remote pilot;
- E94.9 - Remote pilot requisites
- E94.19 - Documents necessary for operation;
- E94.103 - Flight rules;
- E94.301 - Registration and trademarks;
- E94.701 - Contraventions.

According to RBAC-E $n°94$, any operation with RPAS have to take place at least 30 m (at horizontal level) away from persons that are not involved with the operation and autonomous flight are forbidden.

DECEA The Air Space Control Department (DECEA) is in charge of air space access control. It is a department inside the Aeronautics Command, which in turn belongs to Defence Ministry. Using normative ICA 100-40, it specifies the rules that RPAS have to follow in order to access Brazilian Air Space. Some points of this normative are herein highlighted [14].

At chapter two, some definitions and abbreviations are explained, such as:

- 2.1.6 - RPA is an unmanned aircraft without crew piloted by a remote station with non-recreational purpose;
- 2.1.15 - Detection and Avoidance means capability of seeing, perceiving or detecting conflicting traffic and other risks, making possible to take adequate decisions to avoid them;
- 2.1.29 - Autonomous operation is a operation that takes place without the pilot intervention during the flight;
- 2.1.30 - Visual Line of Sight - VLOS: is a operation where the pilot has a direct visual contact with the aircraft, without any external aid;
- 2.1.31 - Extended Visual Line of Sight - EVLOS: the pilot does not have the direct visual contact with the aircraft and then has to use auxiliary equipment or observers to conduct a safe fight;
- 2.1.32 - Beyond Visual Line of Sight - BVLOS: the pilot does not have the aircraft in his visual line of sight, even using an observer;
- 2.1.33 - Radio Line of Sight - RLOS: operation with direct radio link between aircraft and pilot;
- 2.1.34 - Beyond Radio Line of Sight - BRLOS: operation with indirect radio link between aircraft and pilot, i.e. using signal repeaters, satellites and others.
- 2.1.45 - Air Space Access Request System for RPAS - SARPAS: system created to facilitate the requests to access air space by the users.

Chapter three discusses about Brazilian Air Space structure; chapter seven describes pilot responsibilities; chapter eleven the access rules, and chapter twelve all authorization procedures. All the limits and principles of flight are explained, as well as penalties in case of rules transgressions.

1.3 Relevance and Objectives

For increasing flight security, with the increasingly higher number of civilian players coming to the UAV market, it is necessary to provide them with sufficient resources to avoid dangerous flight conditions and, at the same time, let the pilot / autopilot control the aircraft. Computational vision, nowadays, is still available in some aircrafts on the market, however with many limitations as maximum distance of detection, maximum speed of the aircraft, processing time and computational cost.

Many literature about using Lidar for obstacle detection [15], Fuzzy Inference System for control [16,17], underwater robot using AI based algorithms

[18], trajectory control using neural network [19] are available. In spite of that, a system using lidar and sonars for detection and neuro-fuzzy control for avoidance was not found. Hereupon, this work aims at developing an object detection and avoidance system, based on distance sensors (lidar and sonar) and computational intelligence algorithms (based on neuro-fuzzy control). The DAS should be capable to be used in outdoor and indoor environments, and works with flight speed higher than 5 m/s. Thus, the main activities towards that goal were:

- Studies about neuro-fuzzy systems and UAVs;
- Sensors selection and theory;
- Study of commercial flight controllers;
- Development of a test platform;
- Development of the neuro-fuzzy system;
- Integration between the developed system and the platform; and
- Simulated tests and results evaluation.

1.4 Dissertation Structure

Chapter 1 presented the introduction, definitions, applicable legislation, relevance and objective. Chapter 2 brings important aspects about Neural Networks, Fuzzy Logic, Detection and Estimation; topics that were used in the developed detection and avoidance algorithms. In this way, it intends to give the reader a general understanding about the subject. At chapter 3, all the characteristics of the developed prototype are described, including mechanical and electrical specifications. This chapter presents the development of the neuro-fuzzy controller and discusses the system integration with the platform. Chapter 4 presents and discusses the obtained results and chapter 5 presents the conclusions and ideas for future works. The appendices show all of the developed codes.

2
Theoretical Review

2.1
Neural Networks

2.1.1
Basic Concepts

The human brain is a highly complex, nonlinear and parallel computer; that may process certain computations many times faster than a digital computer. This has motivated many works on artificial neural networks in order to mimic a "nervous system" similar to the human brain, with its parallel processing, adaptation and generalization capabilities, including real-time learning. According [20], neural networks are:

> "*A neural network is a massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:*
>
> 1. *Knowledge is acquired by the network from its environment through a learning process.*
> 2. *Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge."*

The learning process is performed by a learning algorithm that modifies the synaptic weights of the network. Neural networks may change its own topology, so as neurons in a human brain may die and new synaptic connections may grow. Generalization at neural networks refers to the ability of producing reasonable outputs for inputs not seen during the learning process. Neural networks are used for the treatment of linear and nonlinear problems, input-output mapping, evidential response (pattern classification), contextual information, fault tolerance, neurobiological analogy and others [20].

The basic component of a neural network is the neuron, which is fundamental to process information. As shown in Figure 2.1, a neuron is composed by three basic elements:

- **Synaptic weights**: similar to biological neuron, a group of synapses receive the input signals. Each synapse is characterized by a weight w_{kj}, the notation representing a input signal x_j connected to neuron k. The input signals are multiplied by synaptic weights. Different from synapses in the brain, a synaptic weight in artificial neurons could swing from negative to positive values.
- **Adder**: for summing weighted input signals. This operation is a linear combiner.
- **Activation function**: to adjust the amplitude range of output signal y_k. Usually normalized from 0 to 1 or -1 to 1.

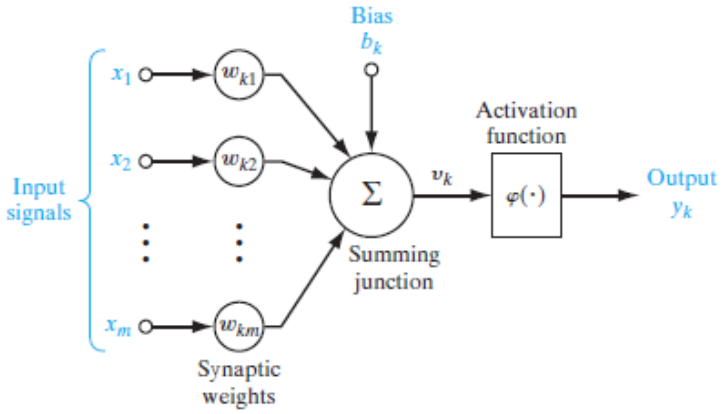

Figure 2.1: Nonlinear model of a neuron [20].

Besides, there is a external *bias* (b_k) used to increase or decrease the input of the activation function. Mathematically, the neuron could be described as given by equations ((2-1)) to ((2-3)), where x_1,..., x_m are inputs; w_{kj} are the respective synaptic weights and y_k the output signal[20].

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{2-1}$$

$$v_k = b_k + \sum_{j=1}^{m} w_{kj} x_j \tag{2-2}$$

$$y_k = \varphi(v_k) \tag{2-3}$$

The two main kinds of activation functions are [20]:

- **Threshold Function**: Also named as Heaviside function. In this model, the output of the neuron is equal to 1, when the input of the activation

function is greater than or equal to zero, and the output is 0 otherwise, as presented in equation (2-4).

$$\varphi_{(v_k)} = \begin{Bmatrix} 1 & if\ v_k \geq 0 \\ 0 & if\ v_k < 0 \end{Bmatrix} \tag{2-4}$$

– **Sigmoid Function**: The most common activation function used in artificial neural networks. It exhibits a great balance between linear and nonlinear behaviour. It is mathematically described by equation (2-5), where a is the slope of the function.

$$\varphi_{(v_k)} = \frac{1}{1 + exp(-av_k))} \tag{2-5}$$

2.1.2 Network Architectures and Knowledge Representation

The learning algorithm used to train the network should consider the structure of the neural network. There are three main classes of neural networks [20]:

– **Single-layer Feedforward Networks**: in this case, the input layer is connected directly to the neurons of the output layer, but the reverse never happens.

– **Multilayer Feedforward Networks**: this case differs from single-layer topology due to the presence of one or more hidden layers. The term "hidden" refers to the fact that this part of the network is not directly connected to the inputs and outputs. The hidden layers allow the network to solve more complex problems, because new neural interactions can be achieved. This kind of network is normally identified by the number of : inputs, hidden layers, outputs, and neurons at each layer.

– **Recurrent Networks**: they have at least one feedback loop, which represents the main difference between them and feedforward networks. These feedback loops improve the learning capability performance of the network.

The knowledge of neural networks refers to stored information by some agent to predict or respond to the outside world. The representation of this knowledge is highly diverse, which makes development of neural networks a challenge. According to [20]: *"...knowledge of the world consists of two kinds of information:*

– *The known world state, represented by facts about what is and what has been known; this form of knowledge is referred to as prior information;*

– *Observations (measurements) of the world, obtained by means of sensors designed to probe the environment, in which the neural network is supposed to operate. Ordinarily, these observations are inherently noisy, being subject to errors due to sensor noise and system imperfections. In any event, the observations so obtained provide the pool of information, from which the examples used to train the neural network are drawn."*

The knowledge of the world is organized as a set of input-output pairs, each pair consisting of an input signal and the corresponding desired response. The set of training data is presented to the neural network during its learning process. As the information inside a neural network is very complex, there are four general rules for knowledge representation:

1. Similar inputs from similar classes should produce similar representations and should be classified in the same class;

2. Items in different classes should have widely different representations;

3. A large number of neurons should be involved in the representation of an important feature;

4. Whenever available, prior information and invariances should be built into the design of the neural network.

As knowledge representation is directly related with network architecture, normally, desired answers are achieved by experimental study for a specific application of interest. The design considerations take places as an essential part of the structural learning loop.

2.1.3 Learning Process

As humans learn from a sort of ways, it is natural to suppose that neural networks can also learn from different manners. The learning process of a neural network can be divided in supervised and unsupervised learning.

Supervised Learning Lets consider a situation of a generic environment that a person knows very well and a blank neural network. At this environment, the network is exposed to different situations and for each one of them, the person tells the neural network how to react. That is, basically, the mechanism of supervised learning. For each input-output training pair, the network parameters are adjusted based on the error signal, which is the difference between the desired answer and actual response of the neural network. When

training is finished, the knowledge is stored as synaptic weights and the neural network may deal with the environment by itself. Some effort should be done to minimize the error. Section 2.1.4 describes some algorithms conventionally used to reduce the error. Figure 2.2 illustrates the supervised learning process.

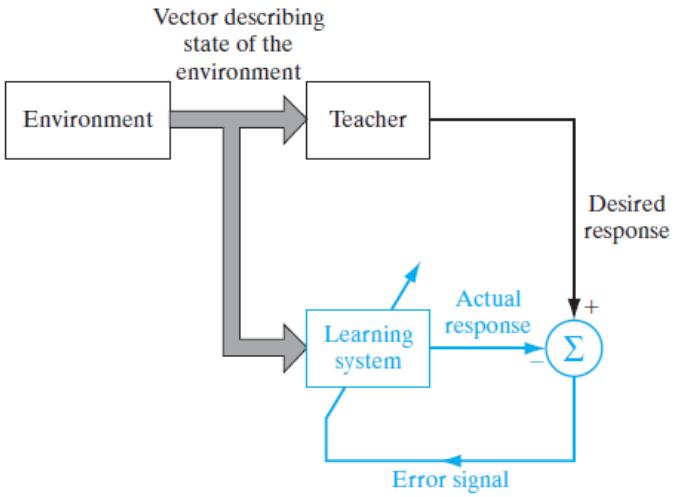

Figure 2.2: Supervised Learning Block Diagram [20].

Unsupervised Learning At this case, no one supervises or teaches the neural network during the learning process. The parameters of the network are optimized with respect to the data available. The purpose of this kind of algorithm is to discover patterns or features. The learning process consists of modifying the synaptic weights of all connections in response to inputs and a set of rules, until a final configuration is achieved. Figure 2.3 illustrates the unsupervised learning process.

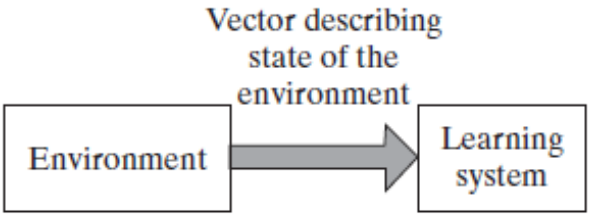

Figure 2.3: Unsupervised Learning Block Diagram [20].

2.1.4 Training Algorithm

2.1.4.1 Least-Mean-Square Algorithm

According to [20], LMS algorithm was inspired by Rosenblatt's perceptron and it was the first linear adaptive-filtering algorithm for solving problems such as prediction and communication-channel equalization. It uses a linear combiner and proved to be robust against external disturbances, computationally efficient and effective in performance. This algorithm is configured to minimize the instantaneous value of the cost function (2-6), where e(n) is the error signal measured at time n. Differentiating this function with respect to the weight vector results in (2-7). With the least-square filter, the LMS algorithm operates with a linear neuron, so the error signal could be expressed as (2-8). Doing some algebra and applying the gradient descent method, the LMS algorithm could be written as (2-9). LMS traces a random trajectory towards the goal, a small η (learning-rate parameter) improves the capacity of the algorithm to remember past data, however the convergence rate becomes slow. From a practical perspective, this algorithm is interesting because it does not require a model of the environment and is very simple.

$$\varepsilon(\hat{w}) = \frac{1}{2}e^2(n) \tag{2-6}$$

$$\frac{\partial\varepsilon(\hat{w})}{\partial\hat{w}} = e(n)\frac{\partial e(n)}{\partial w} \tag{2-7}$$

$$e(n) = d(n) - x^T(n)\hat{w}(n) \tag{2-8}$$

$$\hat{w}(n+1) = \hat{w}(n) + \eta x(n)e(n) \tag{2-9}$$

where:

e(n) = error signal measured at time n;

x(n) = input vector;

d(n) = desired response;

w(n) = weight vector;

$\hat{w}$(n) = instantaneous weight vector.

2.1.4.2 Back-Propagation Algorithm

With the advent of multilayer neural networks other solutions were created to deal with non-linear characteristics and increasingly larger search spaces. The LMS algorithm has some limitations due to its linear characteristics, so, aiming to overcome these, some algorithms were developed. Back-propagation (BP) algorithm was one of them, which is widely used due to its

capability to work with multilayer structures and to find solutions to non-linear problems. At BP, training have two phases: forward and backward. At forward phase, synaptic weights remain the same throughout the network and function signals are computed at neuron-by-neuron basis. At backward phase, the procedure begins at the output layer, passing the error signals backward through the network, layer by layer, towards the input, and recursively computing the local gradient for each neuron.

For the sake of simplicity, just a few parts of the algorithm will be explained and illustrated. Similar to LMS, BP applies a correction to the synaptic weight proportional to the partial derivative, as shown in (2-7). After some mathematical manipulations, the new equation for synaptic weight adjustment can be expressed as:

$$\Delta w_{ji}(n) = -\eta \delta_j(n) y_i(n) \tag{2-10}$$

The local gradient points towards the required changes in synaptic weights. It is given by:

$$\delta_j(n) = e_j(n)\varphi_j^{'}(v_j(n)) \tag{2-11}$$

where:
η = learning-rate parameter;
y_j = signal at output of neuron j at iteration n;
$e_j(n)$ = error signal for neuron j at time n;
$\varphi_j^{'}(v_j(n))$ = derivative of the associated activation function;
$v_j(n)$ = activation function (2.1.1).

In order to increase the learning rate and prevent learning process to finish in a local minimum, without occurring instability of the system, equation (2-10) could be modified to include a *momentum constant* (α) with a value from zero to one. The new equation is:

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta \delta_j(n) y_i(n) \tag{2-12}$$

In general, the convergence of BP algorithm cannot be analytically demonstrated; however, some criteria using Euclidean norm and average squared error are usually applied.

2.2 Fuzzy Logic

2.2.1 Basic Concepts

A high level of precision, normally, implies in high computational cost and takes significant time. Many problems admits a level of uncertainty and according to Professor Zadeh, by accepting some level of imprecision we can do a better job. In this scenario, Fuzzy Logic finds its place when smooth transitions and the nature of human concepts and thoughts are required [4,21].

Fuzzy systems are universal approximators, that can approximate the behaviour of complex systems that are not properly represented by analytic functions or numerical relations. In a very simplistic way, they map an input group to an output group. These groups may be linguistic propositions or other forms of fuzzy information.

2.2.1.1 Fuzzy Set

It is a set where the transition from "belongs" to "does not belong" is gradual, i. e. without a crisp boundary. This transition is characterized by membership functions that commonly use linguistic expressions as "this lake is deep". A fuzzy set is mathematically defined by equation (2-13), as an extension of a classical set. Its construction relies on identification of a suitable universe of discourse and specification of an adequate membership function.

$$A = \{(x, \mu_A(x)) \mid x \epsilon X\} \tag{2-13}$$

where:
A = fuzzy set;
X = universe of discourse;
x = each element;
μ_A = membership function.

The most basic operations in fuzzy sets are: union, intersection and complement; but a notion of containment makes place for better understanding. These operations will be described below:

- Containment: Fuzzy set A is contained in fuzzy set B if and only if $\mu_A(x) \leq \mu_B(x)$ for all x.
- Union: The union of two fuzzy sets A and B is a fuzzy set C, whose membership function is related to those of A and B by

$$\mu_C(x) = max(\mu_A(x), \mu_B(x)) = \mu_A(x) \vee \mu_B(x) \tag{2-14}$$

– Intersection: The intersection of two fuzzy sets A and B is a fuzzy set C, whose membership function is related to those of A and B by

$$\mu_C(x) = min(\mu_A(x), \mu_B(x)) = \mu_A(x) \wedge \mu_B(x) \tag{2-15}$$

– Complement: The complement of fuzzy set A, denoted by $\bar{A}$ (NOT A), is defined as

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x) \tag{2-16}$$

2.2.1.2 Membership Function

Membership functions are subjective, they map each element X of the domain to a correspondent membership grade, between 0 and 1, in relation to a fuzzy set. This grade could be seen as a probabilistic quantity associated to the degree of pertinence of X to the fuzzy set. Membership functions could be symmetrical, asymmetrical and n-dimensional. Some definition terms are important to remark [4]:

– Core: Region of the universe that is characterized by complete and full membership;

– Support: Region of the universe that is characterized by nonzero membership;

– Boundary: Region of the universe containing elements that have a nonzero membership but not complete membership;

– Normal Membership function: it has at least one element x with unitary membership value;

– Convex Membership function: it has membership values that are strictly monotonically increasing or decreasing, or whose membership values are monotonically increasing then monotonically decreasing with increase in the values of the elements of the fuzzy set.

These definitions are highlighted in Figures 2.4 and 2.5.

Along the years, many different types of fuzzy membership functions have been proposed. The most common are: triangular, trapezoidal,Gaussian, bell, sigmoidal, polynomial.

Fuzzyfication is the process of making a crisp quantity fuzzy. It is possible considering that many crisp quantities have an amount of uncertainty then may be represented by a membership function; however, despite of this representation being useful, it is not mandatory.

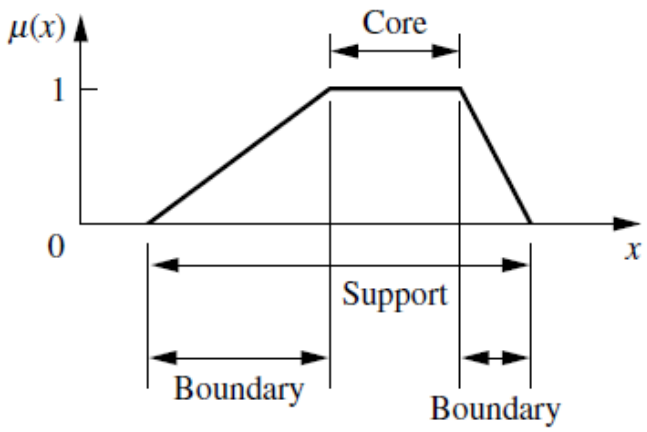

Figure 2.4: Fuzzy set core, support and boundaries [21].

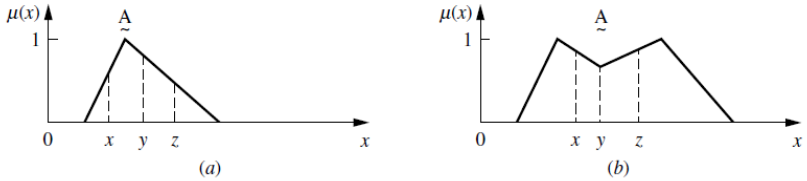

Figure 2.5: Normal convex fuzzy set (a) and normal nonconvex fuzzy set (b) [21].

2.2.1.3 Fuzzy Relations and Rules

Fuzzy Relations are fuzzy sets with n-dimensional membership functions that maps each n-dimensional element to a membership grade between 0 and 1. Applications include areas as fuzzy control and decision making. A binary fuzzy relation can be described as [21]:

$$R = \{((x, y), \mu_R(x, y)) \mid (x, y) \epsilon X \times Y\} \tag{2-17}$$

Many fuzzy relations could be combined through composition operations. The most common are presented below:

Max-min composition are also called max-min product. It is a fuzzy set defined by (2-18),where R_1 and R_2 are fuzzy relations defined on X × Y and Y × Z.

$$\mu_{R_1 \circ R_2}(x, z) = max_y \, min[\mu_{R_1}(x, y), \mu_{R_2}(y, z)] \tag{2-18}$$

Max-product composition is an alternative to the max-min composition. It as defined by equation (2-19).

$$\mu_{R_1 \circ R_2}(x, z) = max_y[\mu_{R_1}(x, y)\mu_{R_2}(y, z)] \tag{2-19}$$

Fuzzy Rules assume the form described at (2-20), where A and B are linguistic values defined the universe of discourse. A is called antecedent and B is called consequent. Sometimes the relation expressed by (2-20) is abbreviated as A $\rightarrow$ B. In general, fuzzy rules can be interpreted in two ways: A coupled with B and A entails B. These contribute to formulate a bunch of methods to calculate the relation R = A $\rightarrow$ B.

$$if\ x\ \ is\ A\ \ then\ y\ is\ B \tag{2-20}$$

Fuzzy reasoning is an inference procedure that obtains conclusions from a set of fuzzy if-then rules and known facts. The basic rule of inference is *modus ponens*, that stablishes that someone could infer the truth of A from the truth of B. From fuzzy reasoning, a system's behavior could be described by some situations, such as:

- Single rule with single antecedent;
- Single rule with multiple antecedents;
- Multiple rules with multiple antecedents.

2.2.1.4 Defuzzification

It refers to the approach used to extract a crisp value from a fuzzy set as a representative value. In general, there are five methods for defuzzifying a fuzzy set C contained in a universe of discourse Z. Three of these methods are explained below [4, 21]:

- **Centroid of area**: the most adopted defuzzification strategy. It is based on calculation of expected value.

$$Z_{COA} = \frac{\int \mu_C(z) \cdot z\, dz}{\int \mu_C(z)\, dz} \tag{2-21}$$

- **Weighted average method**: one of the most computationally efficient defuzzification methods. It is implemented by weighting each membership function in the output by its respective maximum membership value.

$$Z^* = \frac{\sum \mu_C(\bar{z}) \cdot \bar{z}}{\sum \mu_C(\bar{z})} \tag{2-22}$$

where:
$\bar{z}$ = centroid of each symmetric membership function.

– **Mean of maximum**: is the average of the maximizing z at which the MF reaches a maximum μ^*.

$$Z_{MOM} = \frac{\int_{Z'} z\, dz}{\int_{Z'} dz} \tag{2-23}$$

where:
$Z' = z \mid \mu_C(z) = \mu^*)$

2.2.2 Fuzzy Inference Systems

It is a computing framework based on fuzzy set theory, fuzzy rules and fuzzy reasoning. The basic structure consists of a set of rules, database and a reasoning mechanism to perform the inference procedure. There are three different types of inference systems, their differences relying on consequent, aggregation and defuzzification procedures [4, 21].

Mamdani fuzzy model is the most common fuzzy inference system due to its simple structure of min-max operations. It operates in four steps: evaluation of the antecedent of each rule, obtention of each rule's conclusion, results aggregation and defuzzification. Normally, this method is useful when the problem deals with a small amount of variables, otherwise some difficulties may appear, as an exponential increase of the number of rules.

Takagi-Sugeno-Kang (TSK) fuzzy model was proposed in an effort to develop a systematic approach to generating fuzzy rules from a given input–output data set. A typical rule in TSK model, with two-inputs x and y, and an output z, has the form: IF x is A and y is B, THEN z is z = f (x, y); where z = f (x, y) is a crisp function in the consequent. The overall output is obtained via a weighted average defuzzification.

Tsukamoto fuzzy model was proposed by Tsukamoto at 1979. In this method, the consequent of each fuzzy rule is represented by a fuzzy set with a monotonic membership function. The inferred output of each rule is defined as a crisp value induced by the membership value coming from the antecedent cause of the rule. The overall output is calculated by the weighted average of each rule's output. Because of the special nature of the output membership functions required by this method, it is not as useful as a general approach, and must be employed in specific situations only.

2.3 Neuro-Fuzzy

In many cases, to solve complex real-world problems several computing techniques have to work together, constructing complementary hybrid intelligent systems. Neuro-fuzzy systems brings together the ability of neural networks to recognize patterns and adapt themselves to different environments, and the ability of fuzzy inference systems to incorporate human knowledge and perform inference and decision making [21].

A neuro-fuzzy algorithm uses neural networks to tune membership functions of fuzzy systems that are employed to perform decision-making process. Fuzzy logic can encode expert knowledge using rules with linguistic labels, however it takes a long time to properly design and tune the membership functions. Neural networks can automate this process and substantially reduce development time and cost, while improving performance at the same time. The learning process of neural networks is relatively slow and the data is difficult to analyse, besides knowledge acquisition for design fuzzy rules and universe of discourse being a hard task. Then, the use of neural networks was extended to automatically extract fuzzy rules from numerical data. Two types of neuro-fuzzy algorithms are explained in the next subsections [22].

2.3.1 ANFIS

Adaptive Neuro-fuzzy Inference System (ANFIS) is a kind of feedforward neural network composed by nodes and directional links through which the nodes are connected. A part or all of the nodes are adaptive, meaning that their outputs depend on the parameters pertaining to these nodes, and the learning rule specifies how these parameters should be changed to minimize a predefined error measure [21]. The formulas for the node functions may vary from node to node, and the selection of each node function depends on the overall input-output function, which the adaptive network is required to carry out. Figure 2.6 shows an example of ANFIS architecture with five layers, that are herein described:

- **Layer 1**: every node in this layer is adaptive, with a node function that receives x and y as inputs. A_i and B_i are linguistic labels associated with the nodes. The outputs are the membership grades of a fuzzy set. The parameters of this layer are referred as premise parameters.
- **Layer 2**: all nodes are fixed and their outputs are the product of all the incoming signals. Each node output represents the firing strength of a

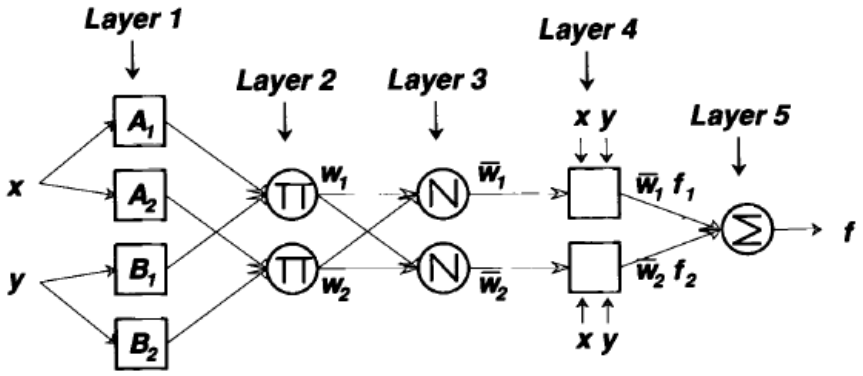

Figure 2.6: ANFIS architecture [21].

rule.

$$w_i = \mu_{Ai}(x)\mu_{Bi}(y), i = 1, 2. \tag{2-24}$$

– **Layer 3**: all nodes are fixed and the i-th node calculates the ratio of the i-th rule's firing strength to the sum of all rules' firing strengths. Outputs of this layer are also called normalized firing strengths.

– **Layer 4**: all nodes are adaptive with a node function given by (2-25). The parameters in this layer are referred as consequent parameters. Here, $\bar{w}_i$ comes from layer 3 and p_i, q_i and r_i are the parameter set of this node.

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i) \tag{2-25}$$

– Layer 5: this single layer computes the overall output as the summation of all incoming signals.

The learning process of this algorithm could be performed in a hybrid way, aiming at converging faster. In the forward pass, consequent parameters are identified by the least-mean-square algorithm. In the backward pass, premise parameters are updated by the back-propagation algorithm.

In theory, when the number of rules is not restricted, a zero-order Sugeno model has unlimited approximation power for matching any non-linear function arbitrarily well on a compact set [21].

2.3.2 CANFIS

Coactive Neuro-fuzzy Inference System is an extension of ANFIS for multiple output problems with non-linear fuzzy rules, being also known as generalized ANFIS. This extension emphasizes many characteristics of neural networks and linguistic interpretability of a FIS. In this kind of system, fuzzy rules are constructed with shared membership values to express correlations between outputs that share same antecedents. Their powerful capability stem

from pattern-dependent weights between the consequent layer and the fuzzy association layer. Figure 2.7 shows a example of CANFIS architecture.

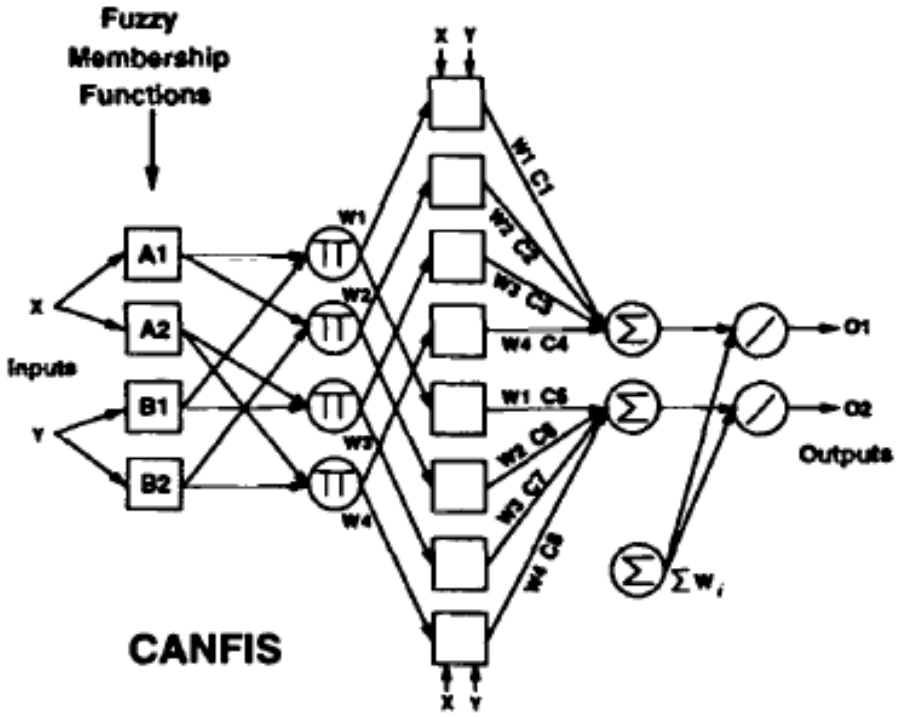

Figure 2.7: CANFIS architecture [21].

Sometimes rules may be hard to understand and membership functions should be carefully determined. Although some difficulties, automatic rules extraction methods are usually capable to outperform manually designed systems. Many combinations could be done to obtain higher precision and performance enhancement [21].

2.4 Detection and Estimation

This section briefly describes some characteristics of distance sensors that may be used in UAV systems in order to implement obstacle detection and avoidance. The topics described are vast and have numerous books and papers about them [23–26].

2.4.1 RADAR

RADAR is an acronym for "radio detection and ranging". Its physical principles of operation where stablished in 1886, but their practical uses started just a little before the second world war, driven by military applications. Nowadays, it is used in an increased number of applications as meteorology,

air traffic control, collision avoidance and earth mapping [26, 27]. One of the main advantages of RADARs in relation to optical instruments is that electromagnetic waves suffer low attenuation from fog, clouds and rain.

In radar applications, to decide if the returned signal came from an object, the amplitude of receiver output is compared with a threshold. If the received signal is higher than the threshold this indicates an echo from a reflecting object, otherwise noise. If an echo is received, the target range could be calculated based on light speed and the time required for a pulse to propagate and return. The range can be estimated by:

$$R = \frac{ct_0}{2} \tag{2-26}$$

where:
R = range;
c = speed of light;
t_0 = time delay after a pulse is transmitted.

It is of great interest to know the location and velocity of a target. In a monostatic radar, as shown in Figure 2.8, transmitter and receiver are located at the same place. The position measurements are performed in spherical coordinates with the origin at radar antenna's phase center. The antenna points in x axis direction, the angle θ is called azimuth and ϕ elevation, both are determined from antenna orientation. Velocity is estimated based on Doppler effect using the shift of target echoes. Despite the fact that Doppler shift gives only radial velocity component, a series of measurements of position and velocity may be used to infer target dynamics in three dimensions.

In a scenario with multiple targets, some RADAR parameters should be considered as range resolution and side lobes; both determined by radar waveform and antenna pattern. Range resolution is a metric to describe radar ability to detect different objects near each other, normally notated as ΔR. It could be expressed as:

$$\Delta R = R_2 - R_1 = c\frac{(t_2 - t_1)}{2} = c\frac{\delta t}{2} \tag{2-27}$$

where:
ΔR = range resolution;
R_1 = range of object 1;
R_2 = range of object 2;
c = speed of light;
t_1 = time delay of object 1;
t_2 = time delay of object 2;

δt = difference between t_2 and t_1.

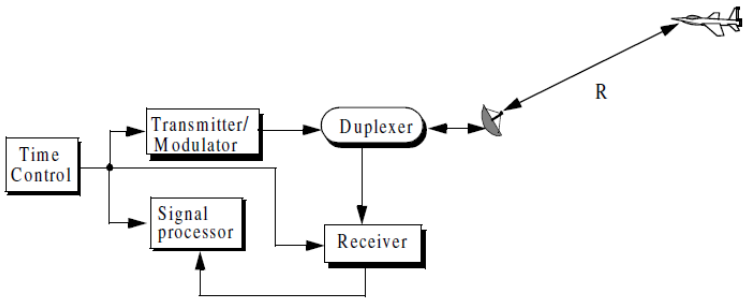

Figure 2.8: Simplified pulsed radar block diagram [27].

In directional antennas, the main lobe has the larger field strength and goes in one direction. However, some undesired radiation in other directions appears, being called "side lobes". Side lobes waste energy in antennas and increase noise level, on receivers. The power density of a radar, considering lossless propagation and a directional antenna, is given by equation (2-28), named Radar Equation.

$$P_D = \frac{P_t G}{4\Pi R^2} \tag{2-28}$$

where:
P_t = peak transmitted power;
R_2 = distance;
G = antenna gain.

2.4.2 SONAR

SONAR is an acronym for "sound navigation and ranging". It uses sound for detection, classification and location of targets [25]. Sonars could be divided in two groups: passive and active. The basic sonar equation is given by (2-29) in dB. It express how much the signal power surpasses the combination of the noise power with a given threshold. A signal excess of zero corresponds to a probability of detection (pd) of 50% and a positive signal excess indicates pd $>$ 50%.

$$SE = S - N - DT \tag{2-29}$$

where:
SE = signal excess;
S = signal;
N = noise;

DT = detection threshold.

2.4.2.1 Passive Sonar

This type of sonar listens to the sound radiated by a target using sophisticated microphones to detect signals against the ambient noise. The nature of the sound (frequency spectrum) helps to classify the target; however, it gives no information about the range. For measuring the range of the target other methods must be applied as triangulation, HDPR (Horizontal direct passive ranging) or VDPR (Vertical Direct Passive Ranging)[25].

2.4.2.2 Active Sonar

This type of sonar, also known as echo ranging systems, uses a projector to generate sound pulses that travel in a medium to a target and return as an echo to a transducer. The echo has to be detected against the ambient noise and reverberation. Knowing the speed of sound in the medium and the time between the transmission of the pulse and the returned echo, the distance range towards the target can be calculated as shown in equation (2-30).

$$R = \frac{ct}{2} \tag{2-30}$$

where:
R = range of a target;
c = velocity of the sound in the medium;
t = time between the transmission pulse and received echo.

The acoustic power radiated by a projector is less than the electrical power supplied to it, the ratio relying on the projector efficiency. Besides, sonars can also estimate the velocity of the target by Doppler Shift, similar to radars.

2.4.3 LIDAR

LIDAR is an acronym for “light detection and ranging”. They are laser-based systems that work in an similar way as radars and sonars. A extremely short light pulse is sent by a short-pulse laser into the atmosphere, being scattered in all directions with a certain probability distribution. Just a small portion of this light returns to the lidar and is collected by a telescope that

focus the light on a photodetector, which in turn converts the light to an electrical analog signal. This analog signal is converted to a digital signal and post-processed [24]. The equation for a monostatic single-scattering elastic lidar is given by (2-31). Monostatic means that laser and telescope are located in the same place and elastic means that the returned light has the same wavelength as the emitted light. Figure 2.9 illustrates a lidar system.

$$F(r) = C_1 F \frac{c\eta_0}{2} \frac{\beta_{\pi,p}(r) + \beta_{\pi,m}(r)}{r^2} exp\left[-2\int_0^r \kappa_t(x)dx\right] \qquad (2\text{-}31)$$

where:

C_1 = system constant, depends on the transmitter and receiver optics collection aperture;

F = radiant flux emitted by the laser, considered constant if laser emits short light pulses of rectangular form;

c = velocity of light in the medium;

η_0 = emitted pulse duration;

$\beta_{\pi,p}$ = particulate angular scattering coefficient;

$\beta_{\pi,m}$ = molecular angular scattering coefficient;

κ_t = particulate and molecular extinction coefficient.

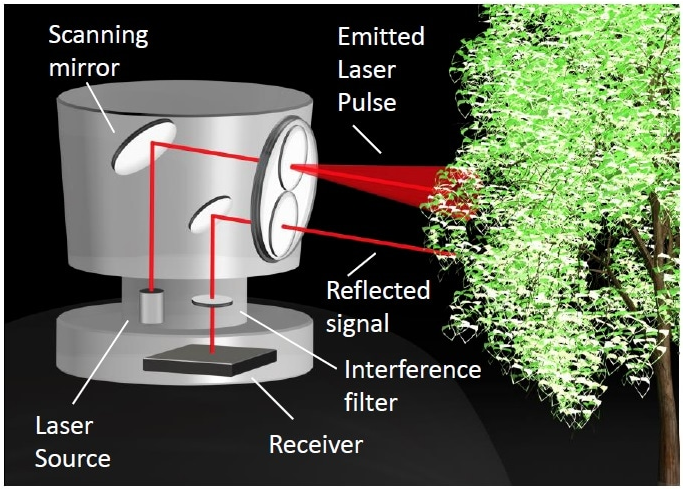

Figure 2.9: Diagram showing lidar system [28].

The laser light is almost monochromatic, so narrow-band optical filters are used to eliminate interference or unwanted light from other sources. These filters improve in the signal-to-noise ratio and increase the lidar measurement range. The range could be measured by the time-of-flight principle, according to equation (2-32), that considers the overall time between the emission of the light pulse and its return [29].

$$R = \frac{cT}{2n} \qquad (2\text{-}32)$$

where:
R = range of a target;
c = velocity of light in vacuum;
t = round-trip time of the light;
n = refractive index of the medium in which the light pulse propagates.

3
Materials and Methods

3.1
Prototype characteristics

3.1.1
Mechanical Specifications

Many things have to be considered during the design of an aircraft system. At this point, someone has to ask: What is the purpose of this system? What capability it should have? How long it should be in the air? What kind of on-board "intelligence"? In order to answer these questions and many others, the conceptual phase takes place[5].

The concept of this project aims at developing a detection and avoidance system that could be used in commercial drones. With that in mind, the use of a quadcopter fits well the purpose because it is very popular, easy to find, has many options in the market and may be acquired by a low cost. The project was based on the TBS Discovery frame, shown in Figure 3.1, that is a variation of the F450 from DJI . Its dimensions are 47 x 32 x 3.5 cm (W x L x H).

TBS Discovery has more internal space then the F450 and the version used is built with carbon fiber, providing more resistance to collisions. Once the frame is selected, it imposes some restrictions to the motor type and to the size of propellers. The Emax motor model MT2216 was chosen in combination with 1045 propellers (10" diameter and 4.5" pitch) [30]. According to the manufacturer's guide, this combination maximizes the motor thrust. It is expected to have a maximum thrust of 950 g per motor, using 4S LiPo (lithium-polymer) batteries (14.8 V).

Two important items to consider in an UAV project are the required payload and total weight of the system, as both reduce the endurance time. The maximum tolerable dimensions for the carried payload are dependent of the frame dimensions. Besides, the maximum payload weight is a function of the UAV empty weight and of the maximum thrust provided by the motors. The payload mass and dimensions may vary, so both have to be analyzed [5]. To maximize the carried mass, 4S batteries where selected to maximize the motor's thrust. The selected motor-propeller configuration, powered by 4S

Figure 3.1: TBS Discovery frame.

LiPo batteries, can reach a maximum system thrust of 3800 g. On the other hand, to increase the tolerable dimensions for payloads inserted beneath the frame, the landing gear shown in Figure 3.2 was used. Its dimensions are 32 x 33 x 15 cm (W x L x H).

The batteries are the heavier part of the system, so that they should be selected carefully. Lithium-polymer batteries were selected, because they have high energy densities and are consolidated in the UAV market. The system is powered by a 4S battery of 5000 mAh / 30C (discharge rate of the battery), with total weight of 481 g. Table 3.1 presents the weight of the system's components, including the embedded electronics. Considering a 3800 g maximum thrust capacity and that the UAV system is flyable up to 70% of the maximum thrust, the payload weight still available after assembling all components was estimated around 500 g.

3.1.2 Sensors and Flight Controller

This subsection highlights the sensors used in the Detection and Avoidance System (DAS) and the flight controller. In order to compose the "eyes" of the DAS, two different types of sensors are used, sonar and lidar. Trying to keep project costs within a reasonable limit, without compromising performance, a lidar is used in the front of the UAV, as main source of ambient detection, and five sonars were attached in the other sides; i.e. rear, right, left, top and bottom.

Figure 3.2: Landing Gear for TBS Discovery frame.

Table 3.1: Table of system's components weight.

OTY	Component	Weight (g)	Total (g)	Price ($)
1	Frame	414	414	32
1	Landing gear	313	313	24
1	Battery	481	481	100
4	ESC	28	112	25
4	Motors	62	248	30
1	Flight Controller	38	38	53
1	GPS	33	33	32
1	Radio Receptor	11	11	86
1	Telemetry TX	4	4	18
1	Camera FPV	50	50	55
1	FPV TX	25	25	30
1	Arduino Nano	7	7	4
5	Sonar	8.5	34	4
1	Lidar	265	265	750
	Miscellaneous	65	65	15
	Total Weight / Price		2100	1258

3.1.2.1 Sensors

LIDAR

The selected lidar was the Leddar® M16 Sensor Module of LeddarTech® that uses LEDDAR technology [29]. This sensor is based on a LED with a 940 nm wavelength, whose work principle is the same of conventional laser lidars. This sensor has a 16-channel photodetector array, providing simultaneous multiple detection, ranging segments and distance measurement of multiple

objects. Figure 3.3 shows the sensor and Figure 3.4 illustrates illumination area, detection segments and beam [31]. The data communication could be performed using CAN port or RS-485 serial.

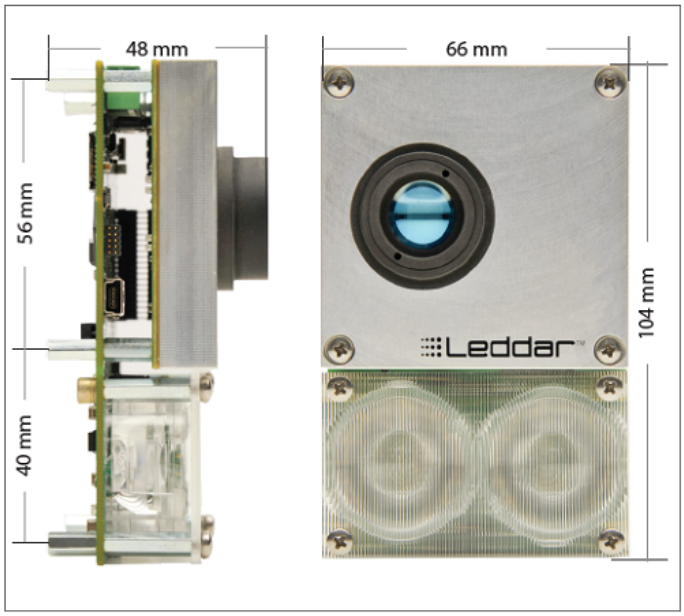

Figure 3.3: Leddar M16 [31].

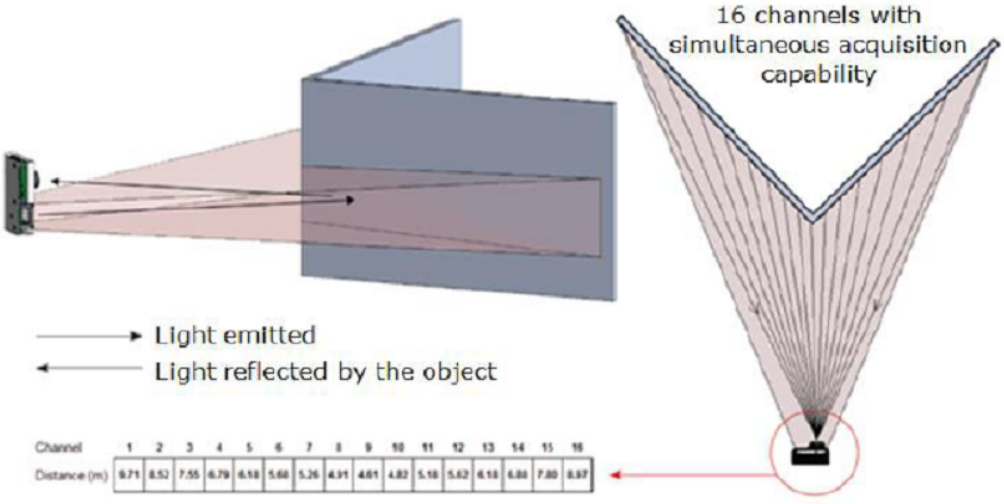

Figure 3.4: Illumination area, beam and detection zone [32].

Some characteristics of the Leddar® M16 Sensor are described below [32]:

- Rugged Technology
- Rapid acquisition rate (up to 50 Hz)
- Large illumination area (95° in this case)

- 8° vertical field of view
- No moving parts
- Short, diffused pulses from infrared light
- 100 meters detection range
- Low cost
- Lateral discrimination, for multiple object detection
- Real-time object tracking capabilities
- Long detection range with low-power light source
- Detection in adverse weather conditions
- Reliability
- Ocular safety
- USB, RS-485, CAN, UART interfaces
- 10 mm distance resolution

Leddar® M16 has limitations regarding to measurement rate, CPU load, LED intensity and others. Thus, it is necessary to find a balance between the setup of the sensor and the project needs, Figure 3.5 shows the established setup. Table 3.2 describes all setup items and shows their respective ranges. Figure 3.5 indicates a CPU load of 72% for the selected configuration, which ensures that it does not overload the system, with a reasonable safety margin.

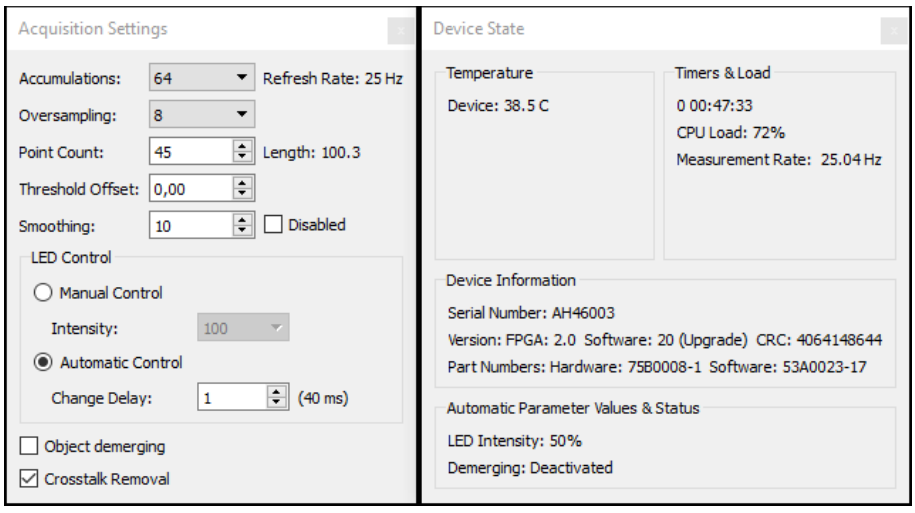

Figure 3.5: Leddar M16 configuration.

It is important to notice that this sensor handles internally all the algorithms to make the point cloud, detection and estimation of the targets. Thus, it is not necessary to do this work externally, saving time and computational cost. The sensor sends, by the CAN port or RS-485 serial, a message containing

Table 3.2: Table of configurations in Leddar M16.

Parameter	Range	Brief Description
Accumulations	1 to 1024	Act in range, measurement rate and noise.
Oversampling	1 to 8	Enhance accuracy, precision and resolution
Point Count	2 to 64	Determine maximum detection range.
Threshold Offset	1 to 100	Modification of amplitude threshold.
Smoothing	-16 to 16	Higher values enhance precision but reduce reactivity.
LED Control	Manual/Auto	Adjust LED power accordingly incoming detection amplitudes.
Change Delay	1 to 8192	Minimum frame delay between power changes.
Object Demerging	N/A	Near-objects discrimination.
Crosstalk Removal	N/A	Remove inter-channel interference noise.

information about the distance until the detected objects, in each one of the 16 channels. This information is used as input in the neuro-fuzzy controller.

In order to reduce to a feasible number of variables that DAS has to deal, the 16 channels were clustered in 3 channels: lidar left, lidar center, lidar right. To achieve that, a minimum function was applied to a group of channels and the result saved in a new variable, Figure 3.6 illustrate the process. The result from channels 1 to 5 was saved at variable lidar left, channels 6 to 11 was saved at variable lidar center and channels 12 to 16 was saved at variable lidar right.

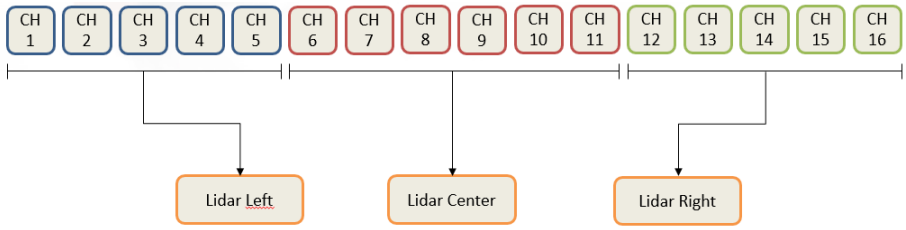

Figure 3.6: Lidar clustering process.

SONAR

Many rangefinders based on sonar technology are available in the market. However, most of them are quite expensive and difficult to find in the Brazilian market. Taking this into consideration, the Ultrasonic Ranging Module HC-SR04 was selected [33]. This sensor is very familiar to Arduino developers, being cheap and easy to find.

The sensor transmits ultrasonic waves into the air and detects reflected waves from a target. It can measure distances from 2 cm to 400 cm, with resolution of 3 mm. The transmitter, receiver and control circuit are all built-in in a single module. It has four pins: 5 V supply, trigger pulse input, echo pulse output and ground. The sensor is shown in Figure 3.7 and its electrical parameters are presented in Table 3.3.

Table 3.3: Electric Parameters of HC-SR04.

Working Voltage	DC 5 V
Working Current	15 mA
Working Frequency	40 kHz
Max Range	4 m
Min Range	2 cm
Measuring Angle	15°
Trigger Input Signal	10 μs TTL pulse
Echo Output Signal	Range in proportion
Dimension	45x20x15 mm

Figure 3.7: Sonar HC-SR04.

Its functionality is quite simple, when a trigger pulse is received on the transducer, the module sends eight pulses of 40 kHz and detects. If it receives a returned pulse in response to the transmitted pulses, the module puts the echo pulse output in high level. The time it remains in high level is proportional to the distance range. The distance until the detect object is calculated with the sonar range equation (2-30).

A careful installation is critical for correct operation of the system, since any misalignment could result in false/wrong information in the output of the sensor. Some characteristics of the environment as roughness and unevenness may interfere in the quality of the results, demanding a post processing treatment of the acquired information. The HC-SR04 does not have embedded computational capability to process data or generate the trigger pulses, so an Arduino Nano was used to perform these tasks [34].

Arduino Nano is a small board based on ATmega328 microcontroller with a 16 MHz clock speed, that is fast enough for this application. It has 22 digital I/O ports and 8 analog input ports. The digital ports are used to generate the trigger signal and to receive the output signals from the five sonar sensors. Arduino sends a trigger pulse and reads the echo time of each sensor. As sound velocity varies considerably in the air according with the temperature,

it is extremely important to measure the temperature to properly calculate the distance towards the target [35]. Consequently, a NTC 10kΩ temperature sensor of GBK Robotics was also connected to the Arduino platform [36], this sensor is a thermistor with a nominal resistance of 10 kΩ at 25° Celsius and the resistance of this kind of sensor decreases with temperature. Thermistor circuit could be seen at Figure3.8, Arduino sends +5 Vdc to sensors boards and reads Vout. The value of RT1 are calculated using Equation 3-1.

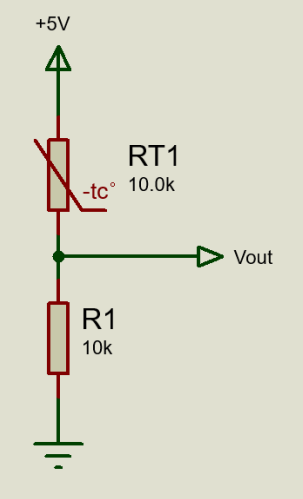

Figure 3.8: Thermistor board electrical diagram.

$$RT1 = \frac{5 * R1}{V_{out}} - R1 \tag{3-1}$$

where:
R1 = fixed resistor;
RT1 = sensor measured resistance.

The calibration of the sensor was done using the Steinhart-Hart Thermistor Equation (3-2). The coefficients of this equation were empirically adjusted, aiming at minimizing the error between the temperature values measured by the sensor compared and the temperatures measured by the Instrutherm TH-1300 thermometer.

$$T = \frac{1}{0.001129148 + 0.0002278707836[ln(R)] + 0.0000000876741[ln(R)^3]} \tag{3-2}$$

where:
T = temperature in kelvin;
R = sensor measured resistance.

The temperature measurements were applied to Equation (3-3) in order to estimate sound speed. The sound velocity calculated by (3-3) is then applied

in (2-30) to estimate the distance towards the target.

$$e = 331.3 + 0.606 * T \tag{3-3}$$

where: e = velocity of sound;
T = temperature in degrees Celsius.

The estimated distances are saved in a matrix that stores the last three distance values measured by each sensor, to eliminate some wrong measurements that could happen, this information is applied to the input of a median filter, whose output is sent to the neuro-fuzzy controller. From Signal Processing Theory, if the distribution is symmetric, the mean is equal to the median. Median filter is robust to outliers and to maintain the speed and softness of the system, only three values are analysed by each turn of the filter, this amount of samples was adjusted empirically.

3.1.2.2 Flight Controller

Nowadays, many flight controllers are commercially available in the market, being easy to find from simple controllers (with gyroscopes and accelerometers only) to very sophisticated ones (with several sensors, ports and computational power). Some of them are proprietary [37], while others are open source [38], with a variety of operational systems.

The Pixhawk Flight Controller was selected to be used in this project [8]. It is an independent open-hardware project developed by ETH Zurich (Swiss Federal Institute of Technology), Autonomous Systems Lab, 3D Robotics, ArduPilot Group and other individuals. It was developed aiming at providing hardware to academic, hobby and industrial communities with low cost, high availability and high-end autopilot design. Many series and manufactures may be seen in the market. The system developed in this work is based on a generic Pixhawk 1 flight controller, updated with FMU version 2, shown in Figure 3.9. This update corrected a problem in version 1 that restricted addressable RAM to 1 MB.

Pixhawk autopilot module runs a NuttX RTOS and supports two flight stacks: PX4 and APM. This operational system is standards compliant, scalable from 8 bit to 32 bit microcontroller environments, highly configurable, has many embedded device drivers , C/C++ compatible, supports graphics and other resources [39]. APM flight stack was chosen due to its flexibility in vehicle types, easy implementation of obstacle avoidance, control override, path planning and GPS based navigation [40]. Below is a list of the hardware

Figure 3.9: Pixhawk Flight Controller.

resources [8]:

- Microprocessor:
  - 32 bit STM32F427 Cortex M4 core with FPU [41]
  - 168 MHz/256 kB RAM / 2 MB Flash Memory
  - 32 bit STM32F103 failsafe co-processor
- Sensors:
  - Micro L3GD20 3-axis, 16 bit gyroscope
  - Micro LSM303D 3-axis, 14 bit accelerometer / magnetometer
  - Invensense MPU6000 3-axis accelerometer / gyroscope
  - MEAS MS5611 barometer
- Interfaces:
  - 5x UART ports, one high-power capable, 2x with hardware flow control
  - 2x CAN
  - Spektrum DSM / DSM2 / DSM-X® Satellite compatible input up to DX8
  - Futaba S.BUS® compatible input and output
  - PPM sum signal
  - RSSI input
  - I2C

  - SPI
  - 3.3 and 6.6V Analog-to-Digital Converter inputs
  - External micro USB port

- Power System:

  - Ideal diode controller with automatic fail-over
  - Servo rail high-power (7 V) and high-current ready
  - All peripheral outputs over-current protected, all inputs ESD protected

3.2 Neuro-Fuzzy Controller

The neuro-fuzzy controller is the decision-maker of DAS, infering the necessary obstacle avoidance decision based on the information received from sonars, lidar and UAV velocity. The proposed controller was developed with the help of a black-box Simulink library, available at MathWorks® Community [42]. This library already has three different implementations of ANFIS and CANFIS system: Scatter, Grid and ART (Adaptative Resonance Theory). These implementation methods modify how the information of input space is partitioned, and, consequently, affect the architecture, operation and approximation capacity of each ANFIS/CANFIS system. Scatter-type uses a quantity of fuzzy rules equal to the number of the fuzzy subsets of each input; Grid-type has a exponential relation between number of inputs and number of rules; on the other hand, ART-type resembles the scatter-type but it is a much smarter unsupervised learning technique that clusterizes the input data exclusively on the areas of the input space where data appear [43, 44]. The selection of input space partitioning method was empirical. Scatter-type gave unsatisfactory results, and Grid-type crashed the computer all times, probably because of the increased number of rules, more precisely 48828125 in this case. However, as expected, ART-type presented good results and computational feasibility.

CANFIS-ART networks consist of six layers, they employ 2 algorithms for parameter learning (i.e. RLS and error-backpropagation) and 1 algorithm for automatic structure learning (i.e fuzzy-ART). Following a brief explanation about each layer:

- Layer 1 (Inputs Normalization): CANFIS-ART uses the technique of complement coding to normalize the input training data. Complement coding is a normalization process that replaces an n-dimensional input vector with its 2n-dimensional complemented coded, this helps avoiding the problem of category proliferation.

- Layer 2 (Input Fuzzification): The nodes belonging to this layer are called input-term nodes and each represents a term of an input-linguistic variable and functions as an 1-D membership function. Here it was used a trapezoidal membership function.
- Layer 3 (Fuzzy-AND Operation): Each node in this layer performs a fuzzy-AND operation and the T-norm operator of the algebraic product was selected. The output of each node in this layer represents the firing strength of the corresponding fuzzy rule.
- Layer 4 (Normalization of each rule firing strength): The output of the k-th node in this layer, is the firing strength of each rule divided by the total sum of the activation values of all the fuzzy rules. This results in the normalization of the activation values of all fuzzy rules.
- Layer 5: The output of all nodes are adjusted by the consequent parameters that represents the contribution of the k-th rule to the m-th output, it is regulated by RLSE algorithm.
- Layer 6: The m-th output of the network is computed as the algebric sum of the m-th node's inputs.

CANFIS parameters were adjusted empirically by try-and-error. Many combinations were evaluated of learning rate, number of membership fuctions, gamma factor, among others. For the sake of brevity, only the best found configuration is presented at Figure 3.10. The detailed explanation of each parameter could be seen at 2 and [42–44]. Following a brief explanation about some parameters:

- Ita: This is the "learning rate" constant used in the back-propagation algorithm to adjust Layer 1 parameters;
- alpha: This is the "momentum term" constant that relates to the error back-propagation algorithm used to adjust the parameters of Layer 1;
- Lambda: This is the "forgetting factor" associated with the Recursive Least Squares (RLS) algorithm that is used to adjust the linear parameters of Layer 4;
- rho_a: This is the "vigilance parameter" of the fuzz-ART algorithm that serves the task of input space partitioning.
- Alpha: The "choice parameter" of the fuzzy-ART algorithm;
- Beta: The fuzzy-ART "learning rate" parameter.
- MF Inclination Factor: This parameter sets the fuzziness of the trapezoidal membership functions that constitute the function of the input term nodes.

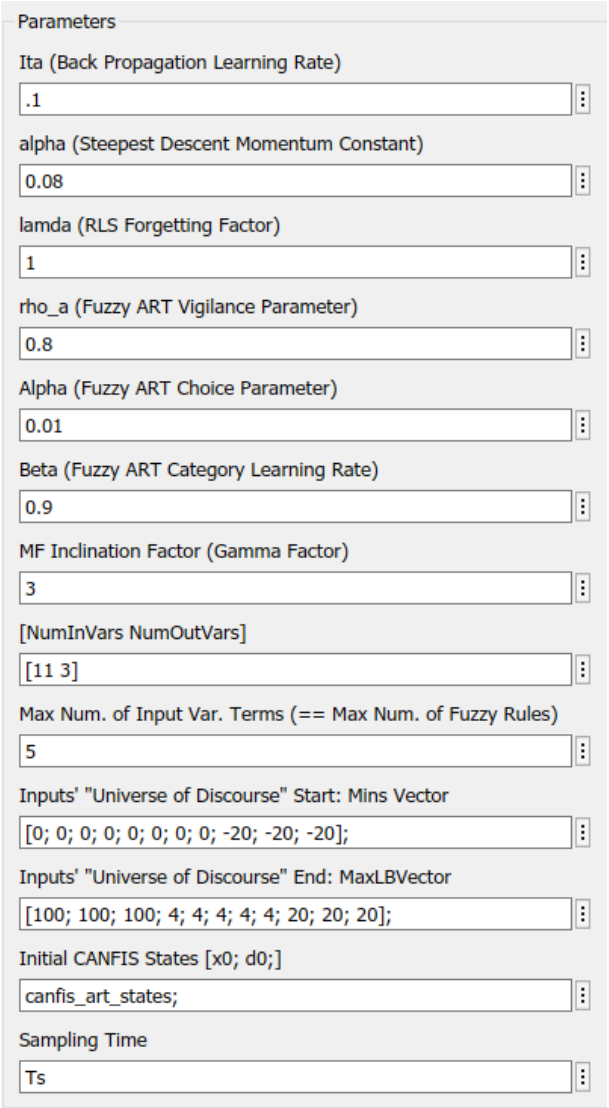

Figure 3.10: CANFIS setup.

The database for training, validation and test is an important issue to consider. Then, a dataset was created, consisting in 1200 different kind of simulated situations. Each situation is a combination between the input variables (lidar left, lidar center, lidar right, sonar left, sonar right, sonar back, sonar up, sonar down, velocity X, velocity Y, velocity Z) and the respective avoidance attitude (pitch, roll and throttle). For correct avoidance attitude setup, each input combination was presented to a specialist, that established the most adequate avoidance attitude response to it. The database was divided in 50% training, 25% validation and 25% test. The number of epochs was used as training stopping criteria and in this case was 600 epochs. The CANFIS structure utilized for training and validation process is highlighted in Figure 3.11. When the signal LE is high (one), the structure performs training; and when LE is low (zero) the structure performs validation.

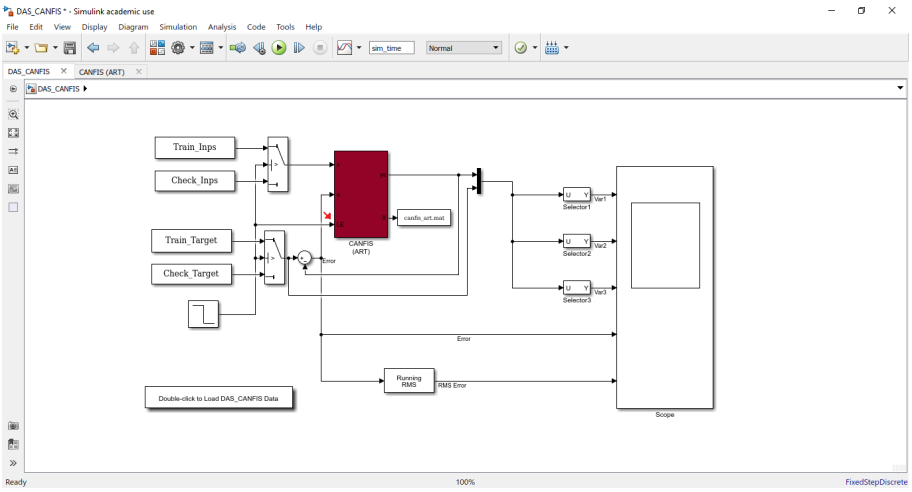

Figure 3.11: CANFIS training setup.

The system output consists of three variables: roll (output variable 1), pitch (output variable 2) and throttle (output variable 3, acts as car accelerator controlling altitude).). As the roll and pitch movements can assume two directions: right / left for roll and forward / backward for pitch, their range can assume negative or positive values, defined between -100 to 100. On the other hand, throttle is unidirectional, so it was defined from 0 to 100. All outputs were normalized in order to provide compatibility with PX4 and APM. Inside the flight controller, the flight stack converts the variables for motor useful information. Figure 3.12 shows the RMSE after the training stage and before outputs normalization. The training algorithm converged after 250 epochs, achieving the RMSE values presented in Table 3.4, that also shows the pitch, roll and throttle RMSEs obtained for the test dataset.

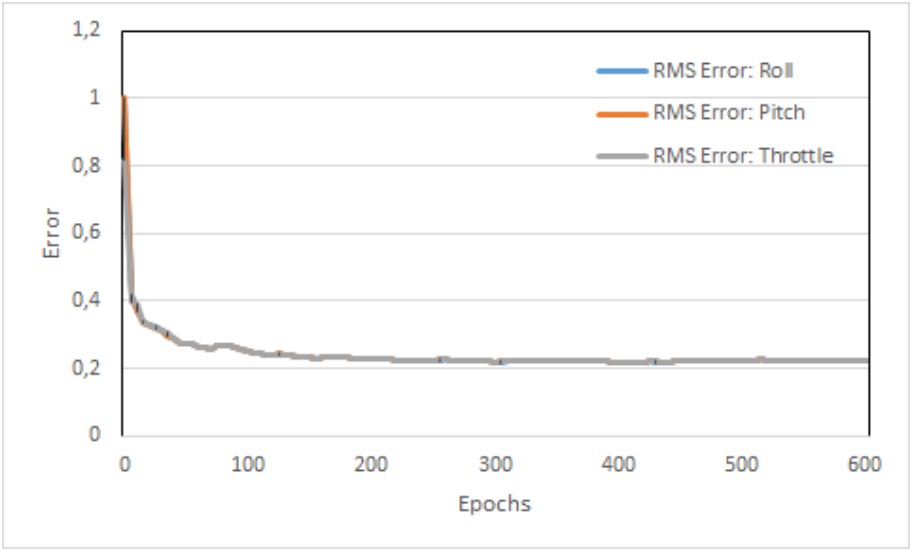

Figure 3.12: CANFIS RMS output error.

Table 3.4: RMSE values.

Output	Training
Pitch	0.2485
Roll	0.2445
Throttle	0.2430

3.3 System Integration

Due to safety issues, the DAS behavior was analysed before integration with the flight stack. To achieve that, a Raspberry Pi 3 model B (RPi) was used to emulate the flight controller and communicate with the sensors and Pixhawk. Raspberry Pi 3 is a small development computer with on-boarded Linux OS and several hardware resources. As the velocity parameters comes from Pixwawk, communication between both of them had to be implemented. This systems integration takes shape in two spheres: hardware and software.

Hardware - RPi has four USB ports and a standard UART [45]. These ports were used in the systems integration, because they are easily integrated. Communication between Arduino, use to read the sonars data, and RPi was done directly by their USB ports. On the other hand, the Lidar sensor has CAN and RS-485 ports. The RS-485 was preferred, even considering the necessity of using a RS-485 to USB adapter. Finally, since USB communication with Pixhawk during flight should not be performed [40], this communication was implemented by UART. Figure 3.13 presents a block diagram representing the hardware integration.

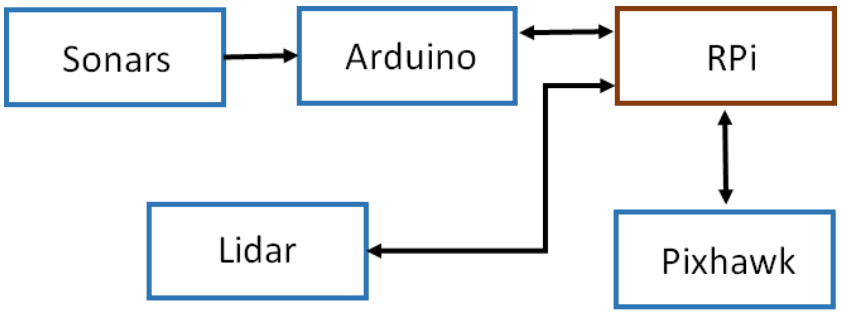

Figure 3.13: Hardware integration block diagram.

Software - A communication protocol was implemented to read information sent from Arduino. Every time RPi requests data, an interruption is triggered in Arduino, that responds with the target distances measured by all sonars.

Communication with lidar was done using MODBUS protocol ([46]), that was already implemented in the sensor by its manufacturer. Besides, the communication between RPi and Pixhawk was established by MAVlink protocol [47].

For this analysis, all the codes used in the neuro-fuzzy controller at Simulink were transcripted to Python. A program was written to receive all sensors measurements and the UAV velocity, and then send this information to the controller and save DAS results to be further analysed. Figure 3.14 presents a block diagram representing the software integration. Besides, Figure 3.15 and 3.16 show the platform with all systems integrated.

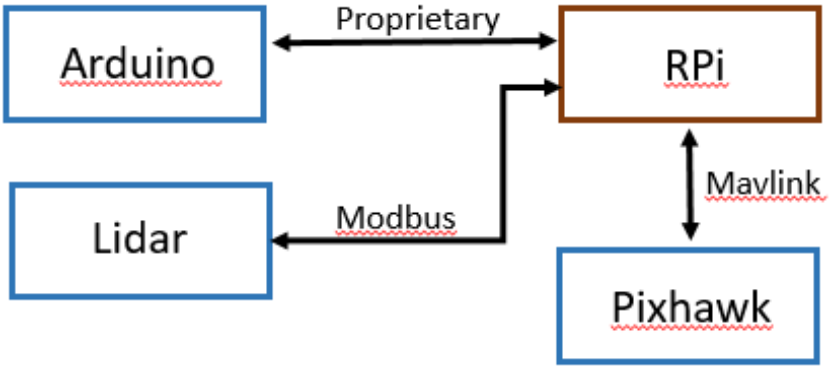

Figure 3.14: Software integration block diagram.

Figure 3.15: Frontal view of platform with DAS on-board.

Figure 3.16: Superior view of platform with DAS on-board.

4
Results

With the UAV assembled, some tests should be performed to verify the behaviour of DAS. The tests were conducted on two different environments: inside a controlled site and open air. Following are the explanations.

4.1
Controlled Site

To conduct this test, a closed ambient with some obstacles was chosen and eight tests were done. Due to the fact that it was a closed ambient, a real flight could infer some risk; thus, some constrains had to be applied to guarantee the integrity of the platform and operator. Following are the description of each test.

Test 1 - here, the UAV was stopped, on a table and with bottom sonar disabled. In this test the intend was to check if false positives were happening. Figure 4.1, Tables 4.1 and 4.2 show the environment conditions and system outputs. It was expected that once the UAV was stopped, everything detected was not a possible collision, thus a zero condition in all of the outputs would be a good value. Considering the 5% radio controller dead-zone, with these outputs, no command of avoidance was send; then a valid output.

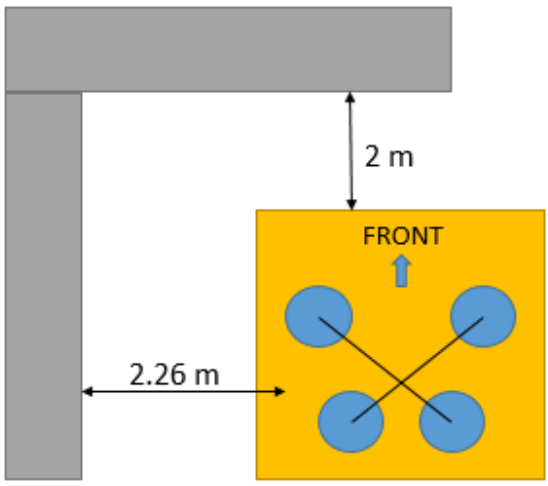

Figure 4.1: First test scenario.

Table 4.1: First test inputs.

Parameter	Entrance Value
Left Lidar	2 m
Center Lidar	2 m
Right Lidar	0 m
Left Sonar	2.26 m
Right Sonar	0 m
Back Sonar	0 m
Up Sonar	0 m
Down Sonar	0 m
Axe X velocity	0.01 (m/s)
Axe Y velocity	0.0 (m/s)
Axe Z velocity	0.0 (m/s)

Table 4.2: First test output.

Parameter	Normalized Value
Roll Output	-0.03
Pitch Output	0.00
Throttle Output	0.05

Test 2 - at this moment, velocity in one axis was considered and all sonars were available. Figure 4.2, Tables 4.3 and 4.4 show the environmental conditions and system outputs. As can be seen, the scenario was crowded of objects, then few actions were available. A object was detected in the back, exactly in the direction of flight (Vx = 1 m/s), the option of the system was compensate the pitch with 12% in the opposite direction, a valid decision. With obstacles in both sides and velocity to the left (Vy = 1 m/s), any movement to any side could be risk; thus the option of the DAS was increase throttle and pass up the obstacle, a valid option. Considering the 5% radio controller dead-zone (area at the center of the joystick that count as zero), the roll command could be ignored.

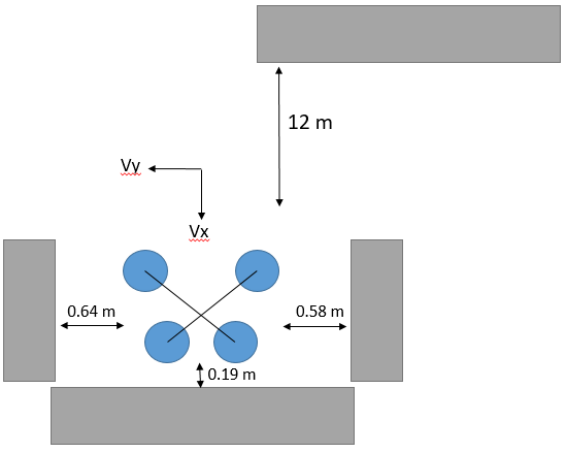

Figure 4.2: Second test scenario.

Table 4.3: Second test inputs.

Parameter	Entrance Value
Left Lidar	0 m
Center Lidar	0 m
Right Lidar	12 m
Left Sonar	0.64 m
Right Sonar	0.58 m
Back Sonar	0.19 m
Up Sonar	2.70 m
Down Sonar	0.19 m
Axe X velocity	1.0 (m/s)
Axe Y velocity	1.0 (m/s)
Axe Z velocity	0.0 (m/s)

Table 4.4: Second test output.

Parameter	Normalized Value
Roll Output	0.04
Pitch Output	0.12
Throttle Output	0.21

Test 3 - here, velocity in one axe was considered and all sonars were available. Figure 4.3, Tables 4.5 and 4.6 show the environment conditions and system outputs. Like test 2, the scenario was crowded of objects, then few actions were available. However, now the system has velocity to go front (Vx = -3 m/s), the near object at this was 12 m of distance; then, at this moment, no avoidance was necessary. It is important to highlight the totally different behavior of the system compared with test 2; with the same obstacles and opposite direction of movement, the DAS was capable of changing its output taking into consideration only the parameters of velocity and direction of movement. This behaviour is very important because obstacles could represent a threat or not depending on the movement of the UAV.

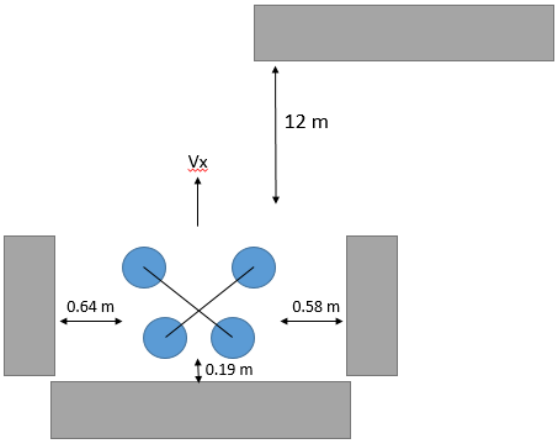

Figure 4.3: Third test scenario.

Table 4.5: Third test inputs.

Parameter	Entrance Value
Left Lidar	0 m
Center Lidar	0 m
Right Lidar	12 m
Left Sonar	0.64 m
Right Sonar	0.58 m
Back Sonar	0.19 m
Up Sonar	2.70 m
Down Sonar	0.19 m
Axe X velocity	-3 (m/s)
Axe Y velocity	0.0 (m/s)
Axe Z velocity	0.0 (m/s)

Table 4.6: Third test output.

Parameter	Normalized Value
Roll Output	0.0
Pitch Output	0.0
Throttle Output	0.0

Test 4 - once again, velocity in one axe was considered and all sonars were available. Figure 4.4, Tables 4.7 and 4.8 show the environment conditions and system outputs. The difference between test 3 and test 4 is the velocity to go front (Vx = -17 m/s). With these conditions, the system judge that no avoidance was necessary, probably because there is still enough space between the UAV and the object for a avoidance action, e.g. to go left or go up. Thus, could be considered a valid decision.

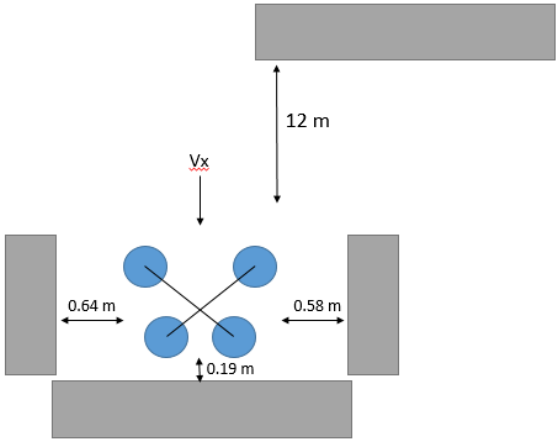

Figure 4.4: Fourth test scenario.

Table 4.7: Fourth test inputs.

Parameter	Entrance Value
Left Lidar	0 m
Center Lidar	0 m
Right Lidar	12 m
Left Sonar	0.64 m
Right Sonar	0.58 m
Back Sonar	0.19 m
Up Sonar	2.70 m
Down Sonar	0.19 m
Axe X velocity	-17 (m/s)
Axe Y velocity	0.0 (m/s)
Axe Z velocity	0.0 (m/s)

Table 4.8: Fourth test output.

Parameter	Normalized Value
Roll Output	0.0
Pitch Output	0.0
Throttle Output	0.0

Test 5 - going ahead the test 4, velocity in one axe was considered and all sonars were available. Figure 4.5, Tables 4.9 and 4.10 show the environment conditions and system outputs. The difference between test 4 and test 5 is the distance of the object reported by the right lidar (2 m in this case). With this velocity (Vx = -17 m/s) and distance of the object, an avoidance action becomes urgent. The answer of the system was increase the throttle in 26% trying to surpass the obstacle. This could be a valid decision if the obstacle is not a huge wall or something like that. The roll and pitch answer was considered a wrong decision because put the system in collision route.

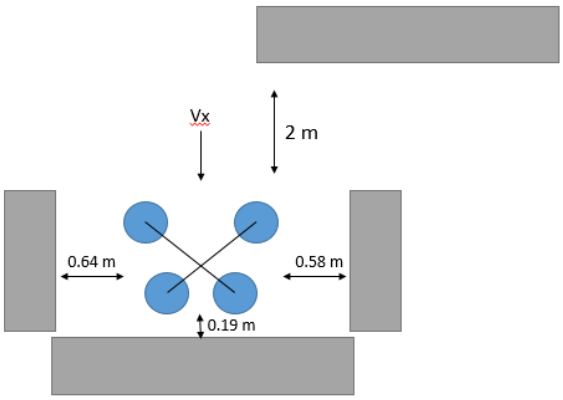

Figure 4.5: Fifth test scenario.

Table 4.9: Fifth test inputs.

Parameter	Entrance Value
Left Lidar	0 m
Center Lidar	0 m
Right Lidar	2 m
Left Sonar	0.64 m
Right Sonar	0.58 m
Back Sonar	0.19 m
Up Sonar	2.70 m
Down Sonar	0.19 m
Axe X velocity	-17 (m/s)
Axe Y velocity	0.0 (m/s)
Axe Z velocity	0.0 (m/s)

Table 4.10: Fifth test output.

Parameter	Normalized Value
Roll Output	0.13
Pitch Output	0.08
Throttle Output	0.26

Test 6 - here, the scenario is completely different. Velocity in two axes were considered and all sonars were available. Figure 4.6, Tables 4.11 and 4.12 show the environment conditions and system outputs. A smooth movement to front and left in a diagonal movement was done, all obstacles very far, the system did not find any collision route and none avoidance was necessary.

Table 4.11: Sixth test inputs.

Parameter	Entrance Value
Left Lidar	73 m
Center Lidar	36 m
Right Lidar	45 m
Left Sonar	0.0 m
Right Sonar	0.0 m
Back Sonar	2.1 m
Up Sonar	0.0 m
Down Sonar	0.19 m
Axe X velocity	-0.37 (m/s)
Axe Y velocity	-0.46 (m/s)
Axe Z velocity	0.0 (m/s)

Table 4.12: Sixth test output.

Parameter	Normalized Value
Roll Output	0.0
Pitch Output	0.0
Throttle Output	0.0

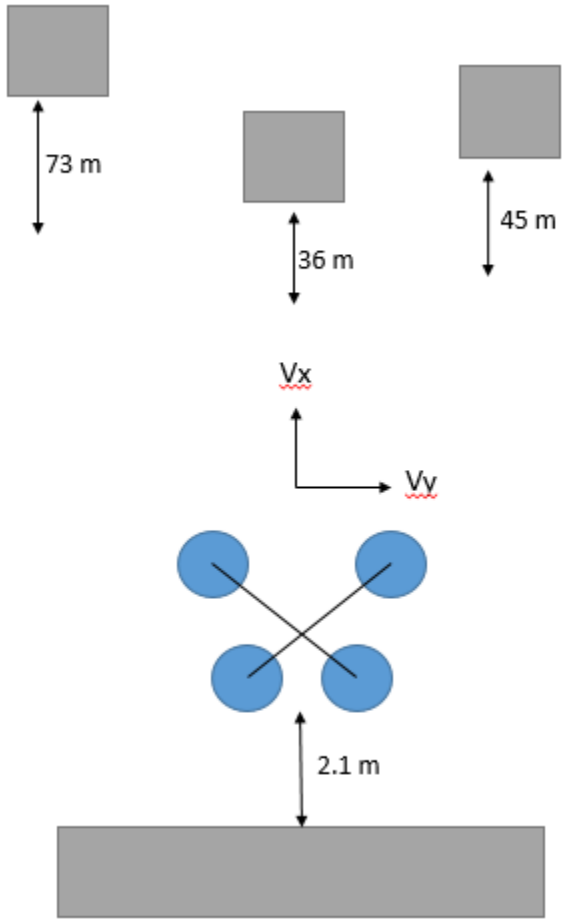

Figure 4.6: Sixth test scenario.

Test 7 - the idea was to check the DAS behaviour in a movement against solo. Figure 4.7, Tables 4.13 and 4.14 show the environment conditions and system outputs. As can be seen, once the movement was slowly, the system reacted smoothly against the ground approximation. This fact is very interesting since it avoids unnecessary sudden movements.

Table 4.13: Seventh test inputs.

Parameter	Entrance Value
Left Lidar	0.0 m
Center Lidar	0.0 m
Right Lidar	0.0 m
Left Sonar	0.0 m
Right Sonar	0.0 m
Back Sonar	0.0 m
Up Sonar	0.0 m
Down Sonar	0.0 m
Axe X velocity	0.0 (m/s)
Axe Y velocity	0.0 (m/s)
Axe Z velocity	0.3 (m/s)

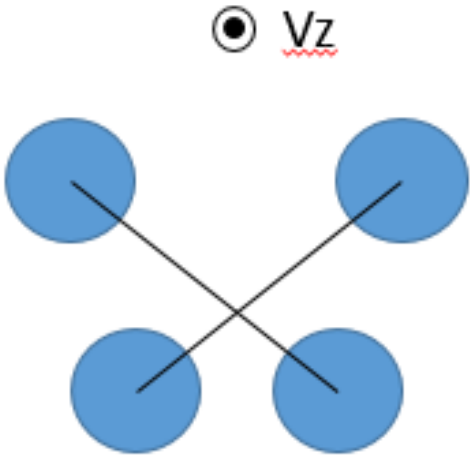

Figure 4.7: Seventh test scenario.

Table 4.14: Seventh test output.

Parameter	Normalized Value
Roll Output	0.07
Pitch Output	0.06
Throttle Output	0.06

Test 8 - in this test, a object was quite near the UAV and velocity in one axe was considered. Figures 4.8, 4.9 and Table 4.15 show the environment conditions and system outputs. The idea was to check the behaviour of the system in the same scenario with different velocity pattern, i.e. velocity changing from its maximum to minimum and from front to back. As can be seen, the DAS output got its maximum when the UAV was in collision route with maximum velocity, as the speed reduces, the output of the DAS decreases too. Considering the 5% radio controller dead-zone, all the outputs below 0,05 can be disconsidered.

Table 4.15: Eighth test inputs.

Parameter	Entrance Value
Left Lidar	1 m
Center Lidar	1 m
Right Lidar	1 m
Left Sonar	0.0 m
Right Sonar	0.0 m
Back Sonar	0.0 m
Up Sonar	0.0 m
Down Sonar	0.0 m
Axe X velocity	-20..20 (m/s)
Axe Y velocity	0.0 (m/s)
Axe Z velocity	0.0 (m/s)

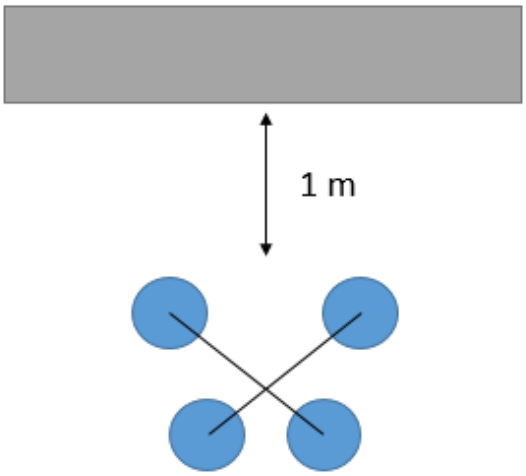

Figure 4.8: Eighth test scenario.

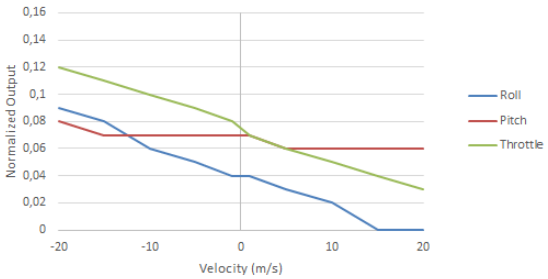

Figure 4.9: Eighth test outputs.

4.2 Open Air Test

This test was conducted at Popular Theater, in the city of Niterói, RJ. It is a large open-air space with good flight conditions and little obstacles, Figure 4.10 shows the place. The test had the objective of finding false-positives (system avoiding obstacles that do not exist) and founds its importance in analysing aleatory commands of DAS. It consisted in hovering and doing some movements like circles, square, front-back, and altitudes varying from 2.5 m to 20 m. Part of the DAS log is shown in Figure 4.11 and the moments highlighted were when the UAV was near the ground, its back facing the operator and system indicating a very small movement to up-front-right, this output condition is shown in Table 4.16 in a simplified way. This output suggest that DAS started to react to the proximity of the floor and was trying to compensate the downward movement.

Figure 4.10: Popular Theater open-air space (Google Earth photo).

Table 4.16: Seventh test output.

Parameter	Normalized Value
Roll Output	0.07
Pitch Output	0.05
Throttle Output	0.06

```
0.06,0.05,0.06,02 - 09 - 2018, 18:27:25
0.06,0.05,0.06,02 - 09 - 2018, 18:27:45
0.06,0.05,0.06,02 - 09 - 2018, 18:28:05
0.06,0.05,0.06,02 - 09 - 2018, 18:28:25
0.07,0.06,0.06,02 - 09 - 2018, 18:29:23
0.07,0.06,0.07,02 - 09 - 2018, 18:29:15
0.07,0.06,0.06,02 - 09 - 2018, 18:29:36
0.06,0.06,0.04,02 - 09 - 2018, 18:29:22
0.07,0.06,0.07,02 - 09 - 2018, 18:29:23
```

Figure 4.11: Real flight DAS log.

5
Conclusions and Future Works

5.1
Conclusions

The main goal of this dissertation was to develop a system capable of detecting and analysing the risk situations during flights and of taking action that mimic a human being. To achieve that, a detection sensor network and an artificial intelligent system based on neuro-fuzzy algorithm was developed. Therefore, a prototype was projected and built to support real experiments. All the work, research done and test results led to some conclusions:

- Lidar technology presents outstanding performance for detection. However, its high costs still only makes it affordable for high value-added operations;
- Sonars have a good value for money, but for in air application show a limited range;
- For small UAVs, radar is impractical due to its weight, price and space necessary;
- Attention has to be given in the UAV design, its purpose and application;
- Databases for detection and avoidance system development is still hard to find and a lot of effort has to be given to its development;
- Neuro-fuzzy algorithm could be a good artificial intelligence solution for this kind of application. However, attention has to be given to networking tuning and training error;
- More tests with the DAS have to be done in different environments to reach a final conclusion about its behavior.

5.2 Future Works

By following this dissertation research, some improvements can be made:

- Database consistency check to find any discrepancies
- Implement the developed algorithm in C++ and run it inside the Pixhawk;
- Substitute the sonars for other model with longer distance range;
- Deep analyse of the training error in order to minimize it;
- Perform more tests to ensure performance and behaviour of the system.

Bibliography

[1] E. Atkins, **Unmanned Aircraft Systems**. Wiley, 2017.

[2] P. Sorokowski, M. Skoog, S. Burrows, and S. Thomas, *Small UAV automatic ground collision avoidance system design considerations and flight test results NASA/TM—2015–218732*, 2015.

[3] S. Haykin, **Neural networks: a comprehensive foundation**. Prentice Hall PTR, 1994.

[4] T. J. Ross, **Fuzzy logic with engineering applications**. John Wiley & Sons, 2009.

[5] R. Austin, **Unmanned aircraft systems: UAVS design, development and deployment**, vol. 54. John Wiley & Sons, 2011.

[6] C. Stöcker, R. Bennett, F. Nex, M. Gerke, and J. Zevenbergen, *Review of the current state of UAV regulations Remote sensing*, vol. 9, no. 5, p. 459, 2017.

[7] Praxis Aeronautics, **Solar Powered Drone**. Acessed on 07/05/2018, disponible at https://www.facebook.com/Praxis-Aeronautics-1314379125241005/.

[8] **PX4 User Guide**. https://docs.px4.io/en/. Acessed on 06/06/2017.

[9] 3DR, **"Supported Drones"**. Acessed on 07/05/2018, disponible at https://3dr.com/products/supported-drones/.

[10] Latitude Engineering, **"Latitude Airframes"**. Acessed on 07/05/2018, disponible at https://latitudeengineering.com/products/hq/.

[11] ANATEL, **"Carta de Serviços"**. Acessed on 07/06/2018, disponible at http://http://www.anatel.gov.br/institucional/carta-de-servicos.

[12] **Regulamento para Certificação e Homologação de Produtos para Telecomunicações**. http://www.anatel.gov.br/legislacao/resolucoes/2000/129-resolucao-242#. Acessed on 10/17/2017.

[13] **ANAC-O que fazemos**. http://www.anac.gov.br/A_Anac/o-que-fazemos. Acessed on 07/16/2018.

[14] **Sistemas de Aeronaves Remotamente Pilotadas e o Acesso ao Espaço Aéreo Brasileiro**. https://publicacoes.decea.gov.br/?i=publicacao&id=4510. Acessed on 10/17/2017.

[15] R. L. C. Araújo, V. Lacerda, A. Hernandes, A. Mendonca, and M. Becker, *Classificação de pedestres usando câmera e sensor LIDAR Anais do Simpósio Brasileiro de Automação Inteligente*, pp. 416–420, 2011.

[16] K. Zemalache and H. Maaref, *Intelligent control for a drone by self-tunable fuzzy inference system* in *2009 6th International Multi-Conference on Systems, Signals and Devices*, pp. 1–6, IEEE, 2009.

[17] V. S. FNU, **Autonomous control of a quadrotor UAV using fuzzy logic**. PhD thesis, University of Cincinnati, 2015.

[18] S. Kundu, **Navigational Strategies for Control of Underwater Robot using AI based Algorithms**. PhD thesis, 2015.

[19] V. Artale, M. Collotta, G. Pau, and A. Ricciardello, *Hexacopter trajectory control using a neural network* in *AIP Conference Proceedings*, vol. 1558, pp. 1216–1219, AIP, 2013.

[20] S. S. Haykin and S. S. Haykin, **Neural networks and learning machines**, vol. 3. Pearson Upper Saddle River, NJ, USA:, 2009.

[21] J.-S. R. Jang, C.-T. Sun, and E. Mizutani, **Neuro-fuzzy and soft computing; a computational approach to learning and machine intelligence**. Prentice Hall, Inc, 1997.

[22] R. Fullér, **Introduction to neuro-fuzzy systems**, vol. 2. Springer Science & Business Media, 2013.

[23] H. L. Van Trees, **Detection, estimation, and modulation theory, part I: detection, estimation, and linear modulation theory**. John Wiley & Sons, 2004.

[24] V. A. Kovalev, W. E. Eichinger, and W. Eichinger, **Elastic lidar: theory, practice, and analysis methods**. John Wiley & Sons, 2004.

[25] A. D. Waite and A. Waite, **Sonar for practising engineers**, vol. 3. Wiley London, 2002.

[26] M. A. Richards, **Fundamentals of radar signal processing**. Tata McGraw-Hill Education, 2005.

[27] B. R. Mahafza, **Radar Systems Analysis and Design Using MATLAB Second Edition**. Chapman and Hall/CRC, 2005.

[28] AZO Optics, **"Thin-Film Interference Filters for LIDAR"**. Acessed on 08/08/2018, disponible at https://www.azooptics.com/Article.aspx?ArticleID=1211.

[29] LeddarTech, **"Overview of a Novel LED-Based Detection and Ranging Technology"**. Acessed on 10/17/2017, disponible at http://www.robotshop.com/media/files/images2/overview-of-a-novel-led-based-detection-and-ranging-technology.pdf.

[30] Yinyan Model Tech. Ltd., **"MT2216"**. Acessed on 12/12/2017, disponible at http://www.yinyanmodel.com/En/ProductView.asp?ID=250.

[31] LeddarTech Inc., 2740 Einstein Street, Quebec, Canada, G1P 4S4, **Leddar M16 and Configurator User Guide**. Acessed on 12/16/2016, disponible at www.leddartech.com.

[32] LeddarTech, **"Specifications Sheets Leddar M16"**. Acessed on 10/17/2017, disponible at http://leddartech.com/.

[33] STMicroelectronics, **"HC-SR04 datasheet"**. Acessed on 10/30/2017, disponible at http://www.micropik.com/PDF/HCSR04.pdf.

[34] Arduino Foundation, **"Arduino Nano"**. Acessed on 10/31/2017, disponible at https://store.arduino.cc/usa/arduino-nano.

[35] U. Papa and G. Del Core, *Design of sonar sensor model for safe landing of an UAV* in *Metrology for Aerospace (MetroAeroSpace), 2015 IEEE*, pp. 346–350, IEEE, 2015.

[36] GBK Robotics, **"NTC 10K Temperature Sensor"**. Acessed on 08/08/2018, disponible at https://gist.github.com/gbkrobotics/97839ccd170b52e92fe5ffffe6d3abc9.

[37] DJI, **Commercial Flight Controllers**. Acessed on 10/17/2017, disponible at http://www.dji.com/products/components?site=brandsite&from=nav.

[38] Robotics Tommorow Magazine, **"Open Flight Controllers"**. Acessed on 10/17/2017, disponible at https://www.roboticstomorrow.com/article/2015/12/five-open-source-autopilot-uav-projects/7436/.

[39] Gregory Nutt, **"NuttX Real-Time Operating System"**. Acessed on 10/18/2017, disponible at http://nuttx.org/.

[40] PX4 Dev Team, **"PX4 Flight Stack"**. Acessed on 10/18/2017, disponible at http://px4.io/.

[41] STMicroelectronics, **"STM32F417xx datasheet"**. Acessed on 10/18/2017, disponible at http://www.st.com/content/ccc/ resource/technical/document/datasheet/98/9f/89/73/01/b1/48/98/ DM00035129.pdf/files/DM00035129.pdf/jcr:content/translations/ en.DM00035129.pdf.

[42] Ilias Konsoulas, **"Adaptive Neuro-Fuzzy Inference Systems (ANFIS) Library for Simulink"**. Acessed on 12/18/2017, disponible at https://www.mathworks.com/matlabcentral/fileexchange/36098-adaptive-neuro-fuzzy-inference-systems-anfis-library-for-simulink.

[43] C.-J. Lin and C.-T. Lin, *An ART-based fuzzy adaptive learning control network IEEE Transactions on Fuzzy Systems*, vol. 5, no. 4, pp. 477–496, 1997.

[44] G. A. Carpenter, S. Grossberg, and D. B. Rosen, *Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system Neural networks*, vol. 4, no. 6, pp. 759–771, 1991.

[45] Raspberry Pi Foundation, **"Raspberry Pi 3 model B Specs"**. Acessed on 08/27/2018, disponible at https://www.raspberrypi.org/products/raspberry-pi-3-model-b/.

[46] MODICON Inc., **Modicon Modbus Protocol Reference Guide**. Acessed on 12/15/2017, disponible at http://modbus.org/docs/PIM_BUS_300.pdf.

[47] Lorenz Meier, **MAVlink Developer Guide**. Acessed on 06/06/2017, diponible at https://mavlink.io/en/.