



Universidade do Minho
Escola de Engenharia

Artur Manuel Pereira Faria

Automotive Gesture Interfaces based on Infrared Technology

Dissertação de Mestrado

Mestrado Integrado em Engenharia Eletrónica Industrial e
Computadores

Trabalho efetuado sob a orientação do

Professor Doutor Jorge Cabral

Janeiro de 2017

Declaração do Autor

Nome: Artur Manuel Pereira Faria

Correio Eletrónico: a63358@alunos.uminho.pt

Telemóvel: 934039842

Cartão de Cidadão: 14214578

Título da dissertação:

Automotive Gesture Interfaces based on Infrared Technology

Orientador:

Professor Doutor Jorge Miguel Nunes Santos Cabral

Ano de conclusão: 2017

Designação do Mestrado ou Ramo de Conhecimento do Doutoramento:

Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores

DE ACORDO COM A LEGISLAÇÃO EM VIGOR, NÃO É PERMITIDA A REPRODUÇÃO DE QUALQUER PARTE DESTA TESE/TRABALHO.

Universidade do Minho, _____ / _____ / _____

Assinatura:

Acknowledgements

The elaboration of this thesis would not be possible without the support, cooperation and motivation of different persons and entities.

I would first like to thank to my thesis coordinator, Prof. Doc. Jorge Cabral, of the Electronic Engineering Department at University of Minho. Prof. Jorge Cabral was available to answer all the needs and problems, giving the best guidance and giving at the same time its sincere opinion whenever was needed.

I also like to thank an unofficial co-coordinator, Eng. Marco Martins, who is a part of the INNOVCAR project from University of Minho. Eng. Marco Martins was always available for all the problems, doubts and questions about my research and writing, giving me always the best guidance and support.

Another special thank goes to my project partners and colleagues, especially all the laboratory colleagues, for their support, comprehension and friendship. I will not forget the lab environment created by this fantastic group of individuals, namely António Ribeiro, Áurea Salgado, Bruno Mendes, Diogo Cruz, Kevin Costa, Pedro Barbosa, Pedro Leite and Rui Paixão .

Another important appreciation goes to all the official project partners, involved in the INNOVCAR project at University of Minho, for the important support given during the dissertation development. It is also worth noting that this work has been financially supported by the Portugal Incentive System for Research and Technological Development in scope of the projects in co-promotion n^o 036265/2013 (HMIExcel 2013-2015), n^o 002814/2015 (iFACTORY 2015-2018) and n^o 002797/2015 (INNOVCAR 2015-2018).

Finally a very special thanks to my parents, João Faria e Maria Faria, and my sister Sara Faria for all the support, comprehension and motivation given not only in this thesis but in all my academic life. Without their crucial support this thesis would not be possible.

Thank you all for everything!

Other Acknowledgments:

European Structural and Investment Funds in the FEDER component, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) Project n° 002797; Funding Reference: POCI-01-0247-FEDER-002797



Universidade do Minho



BOSCH
Tecnologia para a vida



Cofinanciado por:



UNIÃO EUROPEIA
Fundo Europeu
de Desenvolvimento Regional

Resumo

A tecnologia está a evoluir rapidamente e a área automóvel está a evoluir a um ritmo similar. Os veículos, em especial os carros, estão a ficar mais complexos e inteligentes, mas as interfaces entre condutor e veículo (*HMI*s automóveis), tendem a ser menos intuitivas e simples de utilizar com o aumento, em quantidade e variedade, de soluções tecnológicas.

Um problema relevante diretamente ligado às *HMI*s automóveis, é a distração dos condutores. A distração dos condutores é uma das causas mais comuns de acidentes de viação. Em termos globais, a cada ano, cerca de 1,25 milhões de pessoas morrem em acidentes de viação, sendo a distração dos condutores a causa de aproximadamente 20% de todos os acidentes de viação.

Muitas tecnologias e conceitos estão agora a emergir, para minimizar o tempo e a distração do condutor na utilização dos *HMI*s tradicionais, sendo que o reconhecimento de gestos é uma dessas tecnologias e conceitos.

O reconhecimento de gestos, aplicado a interfaces automóveis, é um conceito muito interessante, principalmente porque junta as funcionalidades de um *HMI* "tradicional" com uma interface fácil de utilizar, consistindo principalmente num simples conjunto de gestos, em vez de um variado conjunto de ações, envolvendo os mais "tradicional" botões, botões rotativos, interruptores e ecrãs.

Esta dissertação está integrada no projeto "INNOVCAR: The cockpit of the future" e tem como propósito o estudo e implementação de conceitos alternativos de *HMI* automóvel, que não necessitem de toque ou foco visual, usando reconhecimento de gestos.

Para alcançar estes objetivos, o sistema de reconhecimento de gestos desenvolvido e descrito neste documento, será baseado essencialmente na tecnologia de infravermelhos, principalmente pela flexibilidade e relação custo-benefício desta tecnologia.

Esta dissertação tem como principais contribuições, o levantamento do estado da arte atual e a criação de conhecimento crítico, a ser essencialmente partilhado com os parceiros do projeto. Foi ainda objetivo desta dissertação, o desenvolvimento e a implementação de protótipos funcionais de sistemas de reconhecimento de gestos e o estudo de conceitos *HMI*, usando a tecnologia de infravermelhos. Esta base de conhecimento é importante, essencialmente porque pode ser usada para selecionar novos conceitos e tecnologias para investigação posterior e para o desenvolvimento de novos produtos.

Como conclusão, a dissertação espera oferecer uma visão clara dos possíveis usos de reconhecimento gestual aplicado a interfaces automóveis, bem como os benefícios em termos de performance do condutor, conforto e experiência de condução. A questão da segurança é muito relevante e é esperado que com a redução da perda do foco de atenção na estrada, o número de acidentes baixe, bem como o possível número de mortos e feridos.

Palavras-Chave: Reconhecimento de Gestos, Sensorização Infravermelha, *HMI* Automóvel

Abstract

Technology is evolving fast and the automotive area is evolving at a similar rhythm. The vehicles, especially the cars, are getting more complex and smarter but the interfaces between the driver and vehicle (automotive HMIs), tend to be less intuitive and simple to use with the increasing quantity and variety of the technological solutions.

One major problem, directly connected with automotive HMIs, is the driver's distraction. The driver's distraction is one of the most common causes of traffic accidents. Globally, each year, around 1.25 million people die in traffic accidents, being driver's distraction the cause of almost 20% of all traffic accidents.

Many technologies and concepts are now emerging, to minimize the time and the distraction of the driver in the utilization of the traditional HMIs, being that gesture recognition is one of those technologies and concepts.

Gesture recognition, applied to automotive interfaces, is a very interesting concept, mainly because joins the functionalities of a "traditional" HMI with an easy to use interface, consisting mainly in a simple set of gestures instead of a varied amount of actions involving the more "traditional" buttons, dials, switches and screens.

This dissertation is integrated in the project "INNOVCAR: The cockpit of the future" and has as purpose the study and implementation of alternative automotive HMI concepts, that do not require touch or visual focus, using gesture recognition.

To achieve these objectives, the gesture recognition system developed and described in this document, will be essentially based on infrared technology, mainly because of the flexibility and the relation cost-benefit of this technology.

This dissertation has as main contributions, the lifting of the actual state of the art and the creation of critical knowledge, to be mainly shared with the partners of the project. It was also objective of this dissertation, the development and the implementation of functional gesture recognition system prototypes and the study of automotive HMI concepts using infrared technology. This base of knowledge is important, mainly because it can be used to select new concepts and technologies, for further investigation and to the development of future products.

As conclusion, the dissertation hopes to offer a clear vision about the possible uses of gesture recognition applied to automotive interfaces as well as its benefits in terms of driver's performance, comfort and driving experience. The question of safety is very relevant and is expected that by reducing the loss of the point of focus on the road, the number of crashes should fall as well as the possible number of deaths, wounded and other material costs.

Keywords: Gesture Recognition, Infrared Sensing, Automotive HMI

Content

Declaração do Autor	ii
Acknowledgements	v
Resumo	vii
Abstract	ix
Content	xi
List of Abbreviations	xv
List of Figures	xvii
List of Tables	xxi
List of Algorithms	xxiii
1 Introduction	1
1.1 Background and Motivations	1
1.2 Research Questions	2
1.3 Research Objectives	3
1.4 Research Contributions	4
1.5 Research Requirements	4
1.5.1 Functional Requirements	4
1.5.2 Non-Functional Requirements	4
1.6 Research Resources	5
1.7 Research Risks	5
1.7.1 Technical risks	6
1.8 Thesis Structure	6
2 State of the Art	7
2.1 Gestures	7
2.1.1 Definition of Gesture	7
2.1.2 Types of Gestures	8
2.1.3 Procedural Memory	11
2.1.4 Brain and Procedural Memory	12
2.1.5 Gestures and Procedural Memory	13
2.2 Human Machine Interfaces	13
2.2.1 Gesture-Based Automotive HMIs	14
2.3 Usability	15
2.3.1 Problem of Usability in Automotive HMIs	17
2.4 Gesture Recognition Technologies and Devices	18
2.4.1 Capacitive Sensors	18

2.4.2	Ultrasonic Sensors	20
2.4.3	Radar Sensors	20
2.4.4	Infrared Sensors	21
2.4.5	Laser Sensors	23
2.4.6	Cameras Systems	24
2.4.7	Wearable Devices	26
2.5	Fundamental Principles of Infrared Technology	27
2.5.1	Infrared Radiation	27
2.5.2	Infrared Spectrum	27
2.5.3	Scientific Laws	29
2.5.4	Time-of-flight Concepts	30
2.6	Market Forecasts	32
2.6.1	Market Forecasts for Automotive HMIs Growth	32
2.6.2	Market Forecasts for the Automotive Gesture-Based HMIs Growth	33
2.7	Case Studies and Concepts	33
2.8	Mathematical Models for Gesture Recognition	35
2.8.1	Threshold-Based Model	35
2.8.2	DTW - Dynamic Time Warping Model	36
2.8.3	HMM - Hidden Markov Model	38
2.9	Related Work	41
3	System Specification	45
3.1	Methodology Specification	45
3.2	Development Requirements	46
3.3	System Overview	47
3.3.1	Detailed Overview	47
3.3.2	Gesture Recognition System Overview	47
3.4	Technology Specification	48
3.4.1	Technology Selection Criteria	48
3.4.2	Definition of the technology.	49
3.5	Hardware Specification	49
3.5.1	IR Sensing Devices	49
3.5.2	Selection of Sensing Devices	50
3.5.3	Criteria to select the Development Board	51
3.5.4	Selection of the Development Board	52
3.6	Software Specification	52
3.6.1	Programming Languages	53
3.6.2	Software Development Platforms	53
3.6.3	Algorithm Development	54
3.7	System Architecture	55
3.7.1	Local System	55
3.7.2	Avago APDS-9960 and ams TMG4903	56
3.7.3	PixArt PAJ7620U2	61
3.7.4	ST VL6180X	63
3.7.5	Remote Server	66
3.7.6	GUI Application	68
3.8	Conceptual Design	71
3.9	Data Acquisition and Processing	73
3.9.1	Raw Data Processing	74
3.9.2	Processed Data Treatment	74

3.9.3	HMM Learning Procedure	75
3.9.4	Data Validation	75
3.9.5	Validation Criteria	75
3.10	Validation Processes Specification	77
3.10.1	Experimental Setup	77
3.10.2	Test Methods	78
3.11	Prototype Testing Platform	79
4	System Implementation	81
4.1	System Architecture Implementation	81
4.1.1	Local System	82
4.1.2	Remote Server Interface	90
4.1.3	GUI Application Application Interface	91
4.2	Prototype Testing Platform	95
5	Validation and Results	97
5.1	Calibration Tests	97
5.1.1	Avago APDS-9960	97
5.1.2	ams TMG4903	98
5.1.3	PixArt PAJ7620U2	99
5.1.4	ST VL6180X	100
5.2	Initial Tests	101
5.2.1	Local System	101
5.2.2	Remote Server	104
5.2.3	GUI Application	105
5.3	Integration Tests	106
5.3.1	Avago APDS-9960	106
5.3.2	ams TMG4903	107
5.3.3	PixArt PAJ7620U2	108
5.3.4	ST VL6180X	108
5.4	Final Tests	109
5.4.1	Test Settings	109
5.4.2	Test Results	109
5.4.3	Tests Analysis	112
5.4.4	HMI Concepts Analysis	113
6	Conclusions and Future work	115
6.1	Conclusions	115
6.2	Future Work	116

List of Abbreviations

AC	Air Conditioning
AG	Aktiengesellschaft
ADI	Around Device Interaction
API	Application Programming Interface
BMW	Bayerische Motoren Werke
CAN	Controller Area Network
CCW	Counter Clockwise
CES	Consumer Electronic Show
CW	Clockwise
DSM	Driver Simulator Mockup
FOV/FoV	Field Of View
GM	General Motors
GUI	Graphical User Interface
HCI	Human-Computer Interface/Interaction
ICE	In-Car Entertainment
HMI	Human-Machine Interface
HMM	Hidden Markov Models
HVAC	Heating Ventilation and Air-Conditioning
IoT/IOT	Internet Of Things
I²C	Inter-Integrated Circuit
IP	Internet Protocol
IR	InfraRed
Laser	Light amplification by stimulated emission of radiation
LCD	Liquid Crystal Display
LED	Light Emitting Diode
MCU	Microcontroller Unit
OEM	Original Equipment Manufacturer
PIR	Passive InfraRed
PSA	Peugeot Societé Anonyme
SMS	Short Message Service
SPI	Serial Peripheral Interface
TCP	Transmission Control Protocol
ToF/TOF	Time-Of-Flight
UART	Universal Asynchronous Receiver/Transmitter
USART	Universal Synchronous Asynchronous Receiver/Transmitter
VW	Volkswagen
WHO	World Health Organization
Radar	Radio detection and ranging

List of Figures

1.1	Main research resources.	5
1.2	Main research risks.	5
2.1	Example of gestures being used in our daily life [6]	7
2.2	Example of static gestures [23]	8
2.3	Example of dynamic gestures [24]	9
2.4	Memory classification scheme [27]	11
2.5	Brain structure scheme [33]	12
2.6	Example of gesture-based HMIs.	14
2.7	Visteon’s Horizon cockpit concept (A) [40] and 2016 BMW 7 Series Gesture Control(B) [41]	15
2.8	Google patent proposal for gesture based multimedia control (A)[42] and Ford patent proposal for HVAC control using gestures (B)[43]	15
2.9	Model of the attributes for system acceptability [45].	16
2.10	Model of attributes for system design using ISO standards [46].	17
2.11	Shunt Mode Capacitive Sensing [50]	18
2.12	Load Mode Capacitive Sensing [51]	19
2.13	Transmit Mode Capacitive Sensing [52]	19
2.14	Ultrasound sensor working principle [53].	20
2.15	Radar sensor working principle [54].	21
2.16	Active IR sensor working principle [55].	22
2.17	Passive IR sensor working principle [56].	22
2.18	ToF IR sensor working principle [57].	23
2.19	Laser based sensing - triangulation technique [58].	23
2.20	Structured Light System Schematic(A) [59] and 3D object reconstruction based on Structured Light system(B) [60]	25
2.21	ToF(Time of Flight) System Schematic (A) [61] and 3D object reconstruction based on a ToF system(B) [62]	25
2.22	Stereo camera schematic representation (A) [63] and computer rendered stereo image/object(B) [64]	26
2.23	MYO armband (A) [65], Sony smartwatch 3 (Sony SW3) (B) [66], and Smarty Ring (C) [67]	27
2.24	Electromagnetic spectrum [71]	28
2.25	ToF - Pulsed Modulation Method [72]	31
2.26	ToF - Continuous Wave Method [73]	31
2.27	System response with threshold level.	36
2.28	Analisis of data without(A) [80] or with DTW algorithm(B) [80].	36
2.29	Representation of the two sequences of data [81]	37
2.30	HMM example model, with 6 states (N = 6) [82].	39
2.31	Linear HMM model [83].	39
2.32	Left-to-Right HMM model [84].	40
2.33	Ergodic HMM model [85].	40

2.34	The Gesture Watch (A) and an example of gesture detected (Clockwise Circle) (B) [86]	41
2.35	GBECI gesture detection technique [87]	42
2.36	Hoverflow Concept (A) and Hoverflow Prototype(B)	42
2.37	Contactless Gesture Recognition System prototype [88]	43
2.38	Contactless Hand Gesture Recognition System [90]	43
2.39	Motion gesture sensor scheme (A) and prototype(B)	43
3.1	Waterfall Model Development Steps.	45
3.2	Global System perspective.	47
3.3	Simple diagram of the system.	47
3.4	Complete diagram of the system.	48
3.5	A - Avago APDS-9960, B - ams TMG4903, C - PixArt PAJ7620U2, D - ST VL6180X	50
3.6	A - MikroElektronika IR Gesture click board, B - ams TMG4903 Eval Kit, C - Seeedstudio Groove - Gesture v1.0, D - ST VL6180X SATEL board	51
3.7	EA LPC4088 QuickStart Board [95]	52
3.8	NXP LPC1768 Board [96]	52
3.9	Local System diagram.	55
3.10	Device interfaces scheme.	56
3.11	Main device interface flowchart.	57
3.12	Gesture detection block.	57
3.13	Gesture data read block.	58
3.14	Process Gesture Data block.	59
3.15	Dataset collection example.	59
3.16	Decode Gesture Data block.	60
3.17	Main device interface flowchart.	61
3.18	Gesture detection routine flowchart.	62
3.19	Main device interface flowchart.	63
3.20	Gesture detection routine flowchart - Detection Component.	64
3.21	Gesture detection routine flowchart - Classification Component.	65
3.22	Remote Server Sequence Diagram.	66
3.23	Remote Server Architecture.	67
3.24	Main Thread Architecture.	67
3.25	Client Thread Architecture.	68
3.26	Remote Server Architecture.	69
3.27	GUI Application Architecture.	70
3.28	Main Thread (A) and Support Thread(B).	70
3.29	Timeout Routine1 (A), Timeout Routine2 (B) and Timeout Routine3 (C).	71
3.30	Basic selection of gestures	72
3.31	Basic diagram of concepts outside the car.	72
3.32	Basic diagram of in-car concepts.	73
3.33	Data Acquisition and Processing	73
3.34	HMM Approach for gesture recognition	74
3.35	Confusion Matrix example [97]	76
3.36	Confusion Matrix example [98]	76
3.37	Prototype Testing Platform Mockup	79
4.1	Local System simple diagram.	81

4.2	Prototype Testing Platform Mockup	95
5.1	Functional test using as fixed parameter the number of pulses per measurement and as variable the pulse wavelengths	98
5.2	Functional tests using ams GUI application for TMG4903 sensing device.	99
5.3	Battery of 3 tests for 1x or 2x proximitygains	100
5.4	Sensor response with proximity gain 1x (A), 5x (B), 10x (C) and 40x (D)	101
5.5	Avago APDS-9960 initial interface test.	102
5.6	ams TMG4903 initial interface test.	102
5.7	PixArt PAJ7620U2 initial interface test.	103
5.8	ST VL6180X initial interface test.	103
5.9	Server(A) - Client(B) communication protocol test.	104
5.10	Server(A) - Clients(B) communication test.	105
5.11	GUI application communication test.	105
5.12	GUI application plotting test.	106
5.13	AVAGO APDS-9960 integration test.	107
5.14	ams TMG4903 integration test.	107
5.15	PixArt PAJ7620U2 integration test.	108
5.16	ST VL6180X integration test.	108
5.17	Demo application screenshots.	112
5.18	HMI Concept and possible Sensing devices.	113

List of Tables

2.1	Comparison between 3D camera technologies	26
2.2	Commonly used scheme of regions in the IR Spectrum	28
2.3	ISO 20473 regions of the IR Spectrum	28
2.4	CIE regions in the IR Spectrum	28
2.5	Table of forecasts of the HMI Market Growth	32
2.6	Table of forecasts for the gesture-based HMI Market Growth	33
3.1	Comparison of IR, Capacitive and ToF Camera Technologies	49
5.1	Avago APDS-9960 battery of tests	109
5.2	ams TMG4903 battery of tests	110
5.3	PixArt PAJ7620U2 Battery of tests results	110
5.4	ST VL6180X Battery of tests results	111

List of Algorithms

4.1	Main Routine Highlights	82
4.2	Gesture Detection Routine	82
4.3	Gesture Processing Routine	83
4.4	Gesture Data Processing Routine	83
4.5	Gesture Decode Routine	85
4.6	Main Routine Highlights	86
4.7	Gesture Decode Routine	87
4.8	Main Routine Highlights	88
4.9	Gesture Decode Routine	88
4.10	Main Thread Highlights	90
4.11	Client Threads Highlights	90
4.12	Main Thread Highlights	91
4.13	setupRealtmeData Routine - Timer and Configuration	92
4.14	Support Thread Highlights	92
4.15	Support Routine Highlights	93
4.16	Support Routine Highlights	93
4.17	Support Routine Highlights	94

Chapter 1

Introduction

This chapter is structured to provide a brief, simple and intuitive contextualization to this thesis. In the chapter, the thesis project will be presented together with a brief explanation about the background, motivations, research questions, research objectives and research contributions. Additionally, the research risks, requirements and resources will be also presented.

Finally, the structure of this thesis will be briefly explained for a better perception of its organization and constitution.

1.1 Background and Motivations

Nowadays, mobility is a major necessity and concern for everyone. Worldwide, millions of vehicles are used daily. Cars are one of the preferred means of transportation, mainly because of their flexibility, freedom of use, privacy, security and efficiency.

Besides the possible advantages, car driving is not a “perfect” activity, because some disadvantages and problems exist. One of the main disadvantages or problems of driving a car is the question of safety, both for the car driver and other people, such as pedestrians, other drivers, animals and even infrastructures.

The question of safety is very important because each year, around the world, about 1.25 million people die and 20 to 50 million suffer non-fatal injuries on traffic accidents [1,2].

Another aspect besides the human life’s cost is that traffic accidents cause huge economic losses to the victims, their families, and also to the nations as a whole. The losses mainly come from material losses (vehicles, infrastructures, etc.), medical and psychological treatments, accident investigations, insurance costs, as well as productivity loss for the people with injuries (for example: wages) and for family members who need to take time off work/school to assist the injured.

There are just a few global estimates of the costs on car accidents for the countries, but a research from WHO (World Health Organization) suggests that traffic accidents cost countries about 3% of their gross national product, costs even higher, around 5%, on low-income to middle-income countries [2].

Traffic accidents have a large number of causes, but some of them are more relevant in terms of cost for the car drivers, either in human lives or material losses. Causes such as the driving fatigue, over speeding, drunk driving and distraction are very relevant [1,3]. It is estimated that around 20% of the traffic accidents are caused by driver’s distraction [3].

Traditional solutions of automotive HMIs are usually a source of distraction because they require a considerable set of actions, attention and visual awareness. Furthermore the increasing amount of dynamic information or content in vehicles, like notifications,

alerts and phone interactions and the increasing amount of technological resources make HMIs complex and non-intuitive to use [4, 5].

Being traditional solutions of automotive HMI a source of effort, complexity and distraction, many efforts are being made to try to minimize these problems. New technologies, methods and concepts using gesture recognition, voice recognition, touch sensitive surfaces, haptic feedback, augmented reality, adaptive interfaces, IOT support and many others, are gaining more and more popularity because of the expansion of new technological solutions but also because the market needs more natural, complete, intuitive and safe HMIs.

Gesture recognition is an interesting way to create intuitive and innovative automotive HMIs, because it allows a more natural interaction between the driver and the vehicle, using just a set of gestures to ensure a set of functionalities with minimal effort, actions and attention.

Gesture recognition may be also a good marketing feature for car manufacturers to gain the consumers attention, and to beat the offers/solutions from other major competitors.

Gesture recognition systems will be a confirmed reality in the automotive area, like many studies, forecasts and prototypes predict (see in more detail sections 2.6 and 2.7), because of the cost vs benefits of this systems. Furthermore, there are benefits very notorious in terms of the design of the automotive interfaces, helping to create solutions with more simplicity, flexibility, functionality and less need for attention.

The previous problems and scenarios are especially relevant to the “INNOVCAR: The Cockpit of the future” project and all its partners. Since this thesis is a part of the project it is important the study of this problems and the design of possible solutions.

The purpose of this thesis is the research and development of gesture-based interfaces using primarily infrared(IR) technology to implement alternative concepts of automotive HMI that are contactless and eyes-free.

IR and capacitive technologies are generally one of the preferred technologies for gesture recognition, and are one of the choices in the project in which this thesis is inserted, mainly because of their flexibility and relation cost vs. benefit.

IR technology is a key subject of this thesis and is a very interesting technology to work as a solution to implement a set of automotive HMIs concepts that are contactless and eyes-free because it allows the implementation of a gesture recognition system with 1 to 3 dimensions of sensing, capable to recognize a varied set of gestures such as swipe gestures, circle gestures or even wave gestures.

The set of gestures capable to be recognized can be joined in a complete gesture library and can be used, in this project, to implement a set of use cases like AC control, audio control, GPS control, and many others such as the control of mirrors, windows, roof, lights and others.

Since this thesis does not have as an objective the integration of the gesture based interfaces in a ”real” automotive platform/vehicle or a driving simulator, the development, test and validation is made in-lab, and under controlled conditions.

1.2 Research Questions

For the development of this thesis and considering the subject in question, the study of gesture-based automotive HMIs, some research questions were taken in account to create a research methodology and research plan. These questions were selected after a period of study and consideration and are respectively:

- What devices, based on IR technology, should be used for the development of automotive HMI concepts?
- What gestures, based on the selection of IR sensing devices, can be recognized and can be used for the development of automotive HMI concepts?
- What automotive HMI concepts should be validated with the previous selection of devices and gestures?

Note:

IR technology was selected not as result of a personal study like regularly it should be, but because it was a strategical choice done, after a period of consideration and market analysis, in the planning phase of the project INNOVCAR which this thesis is a part of. Besides this fact, a brief study was conducted in this thesis, involving the major alternative technologies as a complement to the study involving IR technology.

1.3 Research Objectives

The main research objectives for this thesis focus on the study and creation of gesture-based interfaces, using infrared sensing devices and hand gesture recognition. These interfaces, achieved through the creation of gesture recognition libraries, are an important step for the development of alternative and innovative automotive HMI concepts capable of being implemented and tested in a simulated or real world environment.

The list of research objectives considered is composed by six main objectives, that are respectively:

- study and selection of devices, based on infrared sensing, to be used for gesture recognition purposes;
- implementation of gesture libraries to interface with the selection of gesture recognition devices and to recognize the gestures, in real time;
- study and selection of a set of automotive HMI concepts, using gesture recognition, to be implemented and validated(in-lab);
- implementation of demo automotive HMI concepts, as part of a research for a prototype system that must be functional, robust and simple to use;
- in-lab integration and validation of the demo HMI concepts; here the goal is not to simulate the "final" integration of the concepts in a "real world" environment, but only test the functionality of the concepts of interaction testing mainly 3 factors: the reliability, efficiency, and responsiveness;
- collecting and synthesizing of all the critical knowledge collected on the study, development and testing phases.

1.4 Research Contributions

This thesis is inserted on the project “INNOVCAR: The Cockpit of the future” and has at least two major contributions.

- The development of a new base of critical knowledge, to be mainly shared with all the project partners in terms of gesture recognition technologies, devices, algorithms and especially concepts of interaction using infrared technology applied to automotive HMI’s.

This base of knowledge is very important because it can be used to create new concepts of HMI to be used in further studies and also possibly to insert in future commercial solutions, or in this case, in future automotive HMI’s.

- Since the gesture detection algorithms will be developed and tested, their use could be extended in the future to other types of interfaces, not only to automotive HMI’s. The applications of this type of systems and concepts can be extended to a huge variety of applications, like outdoor advertising placards, virtual showcases in shops, contactless controllers, home appliances, and many other applications or use cases.

1.5 Research Requirements

1.5.1 Functional Requirements

For the development of this thesis, a set of functional requirements were selected. The functional requirements selected were:

- collect gesture data from the gesture recognition device(s);
- process the collected data using gesture recognition algorithms;
- communicate a response for each gesture detected to a central stack/infotainment system using an Ethernet connection (TCP/IP protocol).

1.5.2 Non-Functional Requirements

For the development of this thesis, a set of non-functional requirements were selected. The non-functional requirements selected were:

- be able to capture valid gesture data in gestures with different hand positions, speeds, distances and execution times;
- ensure low levels of calibration for the gesture recognition algorithms;
- locally process the gesture data, and communicate the system responses to a central stack/infotainment system in a period of time of less than 1 second.
- ensure that the rate of gesture detection can be equal or above 70%;
- ensure that, for each type of gesture, the rate of recognition can be equal or above 70%;
- create a prototype capable of being integrated and tested in a “real world” driving simulator;

- ensure that a basic set of gestures can be recognized by all the different gesture recognition devices, creating that way a gesture recognition platform independent of the gesture detection device.

1.6 Research Resources

This research project was considerably dependent on hardware and software resources. The list of resources was divided in four blocks, namely gesture recognition devices, development platforms/boards, other hardware and software tools (such as software applications, software frameworks and auxiliary hardware elements) and a central stack/infotainment system.

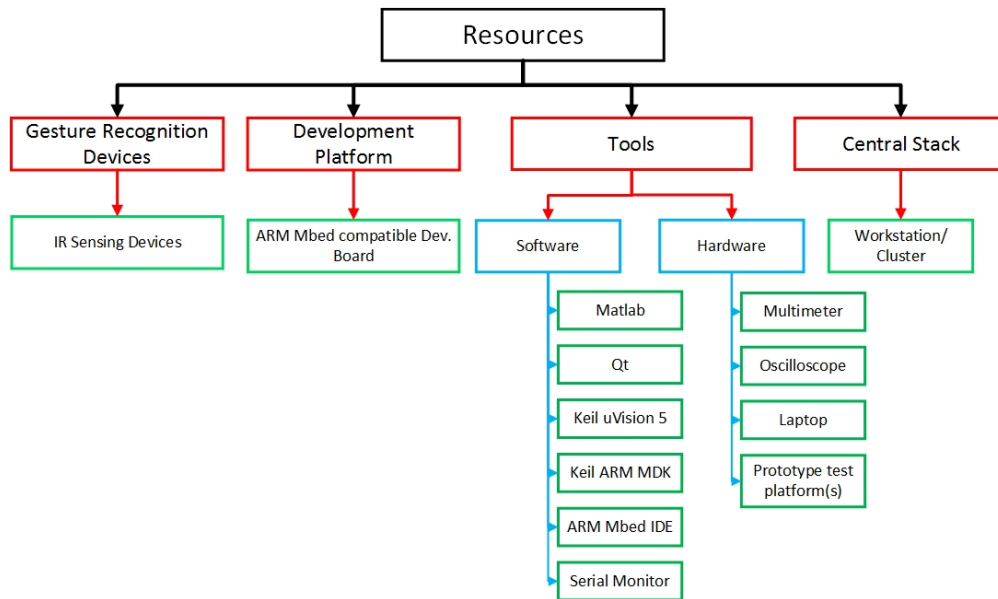


FIGURE 1.1: Main research resources.

1.7 Research Risks

This research project had also important risks to be taken in consideration. The nature of the risks was divided in four types, the technical risks, external risks, organization and management risks.

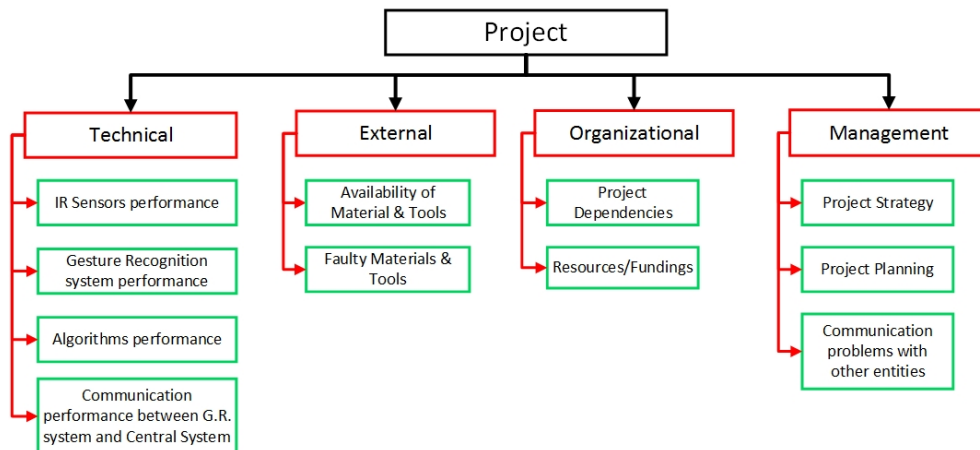


FIGURE 1.2: Main research risks.

1.7.1 Technical risks

- IR sensors can suffer interferences related to varying working and environmental conditions. The sensors performance is a huge variable to consider. The varying performance with different temperatures, object colors and surface shapes allied to the false detection of gestures and the varying quality of collected data, can be a source of problems.
- The performance of the gesture detection algorithms is also an important variable. The risk of low processing speed, low quality of data acquisition and treatment as well as low rate of effective detection, are important factors that make algorithm development a source of risk.
- The general system performance is another variable to consider, and the risk of a low performance system must be considered. The system efficiency, reliability, rate of detection and throughput of responses, for example, are important variables to consider.
- The communication performance and reliability between the gesture recognition system and the central system (central stack/cluster) is also an important risk. The faulty communication between both systems can lead to an unresponsive and inefficient system causing a bad working order.

1.8 Thesis Structure

This thesis is organized and described in six main chapters. The first chapter is an introductory chapter where it is presented a simple overview about the scope of the project and research motivations, questions, objectives, contributions, requirements, resources and risks. Here the goal is to briefly justify the why of this thesis, present the field of work and also other important information related to the management of the thesis.

The second chapter covers all the state of the art. In this chapter, all the theoretical concepts related to the use of gestures in HMI's are presented. Besides that, a small analysis of the available technologies is conducted. Also in this chapter, it is presented a small state of the market, with the presentation of market forecasts, case studies and concepts. Finally a review of the fundamental principles in IR technology is conducted, and a brief study of some mathematical models used for gesture recognition is developed.

The third chapter will cover the methodology used for this thesis. In this chapter, the methodology is defined and all the systems specifications are presented, including the system architecture, hardware and software components, conceptual design, and validation procedures.

The fourth chapter will cover important implementation details in the development of this thesis. In this chapter, a brief explanation about important choices and procedures is made covering both hardware and software implementations.

In the fifth chapter, all the experimental findings are presented and a more detailed explanation of the procedures and tests is also presented. A final analysis is also done to comment the test results and objectives.

The sixth chapter presents all the conclusions related to the developed work during this thesis, mainly in terms of achievements and questions to solve. Also in this chapter, a brief analysis of possible future work and research is conducted.

Chapter 2

State of the Art

This chapter is structured to present a set of important knowledge's related to the area of work developed in the thesis, gesture-based HMIs using IR sensing.

This chapter will start by describing important concepts related to gestures, gesture interfaces, gesture technologies, and IR sensors theory of operation. Then, an analysis of the related work and important market forecasts and concepts of study will be presented. Finally, a small introduction to some mathematical models will be also presented.

2.1 Gestures

Communication can be defined as a process of sending and receiving information, ideas, thoughts and emotions, among different individuals, through speech, gestures, writing, or behavior.

From the early ages of man to the present, many forms of communication were developed and used, being all of them important for, at least two main purposes: the transmission of knowledge (ideas, concepts, facts, etc) and the development of relations between different persons or individuals. It is known, as a fact, that the capability of the human individuals to communicate with each other in a efficient and organized way, was one of the reasons of the human species success.

2.1.1 Definition of Gesture

In a simple definition, a "gesture" is a form of non-verbal or non-vocal communication in which visible body actions are used to communicate specific ideas and emotions.

Considering the Porto Editora dictionary, a gesture is a "movement of the body, especially the head and arms, that expresses ideas or feelings". Another definition, from the Merriam-Webster dictionary, defines a gesture as "a movement of the body (especially of hands and arms) that shows or emphasizes an idea or a feeling".



FIGURE 2.1: Example of gestures being used in our daily life [6]

In a more general perspective, a gesture is defined as the movement of the body, limbs, arms, hands or other body parts that expresses or emphasizes an idea, feeling, or attitude.

Gestures are a form of communication usually used as a complement of the regular speech, transmitting that way a more or less subtle set of messages, but can also be a substitute to the speech itself.

2.1.2 Types of Gestures

The classification or typification of the gestures is covered, in a extensive way, in a huge quantity and variety of literature, from computer vision, psychology to the linguistic areas [7–21]. Besides this fact, the classifications from each one of the areas are not completely equivalent or even convergent. The contrast between the areas of computer vision with the psychology or even linguistic areas is especially notorious.

During the research several types of gestures were identified and in the following topics, this types of gestures will be presented.

A first classification of the gesture types can be established by analyzing their evolution over time, dividing the gestures in two major types: the static and the dynamic gestures.

Another classification can be done by analyzing the level of conscious or awareness of the individual who performs the gestures, diving the gestures in two major types: the "Conscious Gestures" and the "Unconscious Gestures" [11, 22].

2.1.2.1 Static vs. Dynamic Gestures

A possible classification of gestures is based on the nature of the gesture during a certain period of time, where the evolution of the gesture differentiates its nature. Applying this classification, two types of gestures are identified: the static gestures and dynamic gestures.

A static gesture is typically a gesture where in a certain period of time, the hand, fingers, arm or other body parts that are performing a gesture are stopped without movement. Basically if the body part is the hand, a typical static gesture could be the "V" gesture, the "stop" gesture (open hand gesture), the "Ok" gesture or many others. These gestures are usually used in gesture interfaces, especially in computerized vision implementations.



FIGURE 2.2: Example of static gestures [23]

A dynamic gesture is typically a gesture where in a certain period of time, the hand, fingers, arm or other body parts that are performing a gesture are moving transmitting a certain message with the movement. These gestures are used in gesture interfaces, both in touch based interfaces or touchless interfaces.

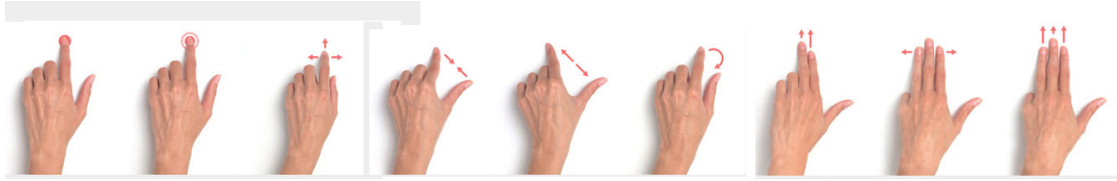


FIGURE 2.3: Example of dynamic gestures [24]

2.1.2.2 Conscious vs. Unconscious Gestures

The conscious gestures are a result of the persons intentions, ideas and emotions, possessing a meaning (subtle or very clear) by their own.

These gestures represent a good percentage of gestures that all the persons use on the daily routines, despite being less used than unconscious gestures.

These gestures can be divided in two great categories: the "Emblematic Gestures" and the "Propositional Gestures".

- **Emblematic Gestures**

Emblematic gestures are a type of gestures that are culturally specific, because one single gesture, can be interpreted with different meanings, dependent of the culture where it is used. These gestures can be so specific, that even inside the same culture, the meanings can be very divergent, with multiple meanings dependent of the associated contexts.

For example, the traditional "V" gesture, done with the index and middle finger, can be interpreted as a political gesture, as an official salute, as an insulting gesture, as the number 2 or "V" in sign languages, as the "V" of victory, as symbol of peace, as symbol of friendship and many others.

Many more emblematic gestures exist, mainly because usually these gestures are easy to understand (with the associated context), easy to remember and easy to do, being commonly used in human-machine interfaces, for example.

- **Propositional Gestures**

Propositional gestures have a "proposal" associated, so the idea is to transmit an idea or proposal through the execution of a gesture. These gestures tend to be intuitive and very natural to use in the daily life.

A good example of these gestures is the use of pointing gestures to mark certain movements or actions, like the "move that card to this place", or "move that book over there".

These gestures are important in task-oriented speech, mainly because of the associated action or proposal, and for this reason the use is somehow limited in the daily routines.

Besides these gestures being intuitive and easy to do, in HMIs the interactions using "proposal" or subtle "actions" are not the most common. Besides this fact, some HMIs tend to use some proposition gestures, for example to navigate in menus, open hidden menus or other actions.

Unconscious gestures are the type of gestures that are unintentional, instinctive and non premeditated. These gestures are very important because they correspond to

the vast majority of gestures that every person uses daily.

This type of gestures tend to be a gestural vehicle of the humans communicative intent. However, these gestures tend to be ignored when it comes to the development of gesture interfaces, either in computer vision interfaces or even other types of Human-Machine interfaces. The main reason for this fact is that unconscious gestures do not come immediately to mind when a person reflect about them.

These gestures can be divided in four great categories, the Iconic gestures, Metaphoric gestures, Deictic gestures and the Beat gestures, all of them explored ahead.

- **Iconic Gestures**

Iconic gestures are a type of gestures closely related to the speech itself, illustrating in a more visual way what is being said by someone. These gestures are important to add detail to the speech, for example in the description of the action of holding or throwing an object.

These gestures represent physically and visually the message of the speech, and can even pass more information than the speech itself. For example, the way that a speech describes an action like "kicking a ball", can be very divergent from the gesture itself. For this reason, iconic gestures are a representation or a viewpoint from the action narrated by someone, because if a person does not fire a gun and someone explain how to fire it, the action of the listener will be based on the example of the narrator.

- **Metaphoric Gestures**

Metaphoric gestures are a type of gestures where the representational aspect is very important. The difference or particularity is that the concepts or ideas that they represent have no physical form or direct connotation to something, thus being considered a metaphor of something.

For example, in the speech "the wedding was going and going ..." accompanied by a hand spinning movement is a form of speech plus metaphoric gesture, where the hand spinning passes the message of the time passing during the wedding.

These type of gesture or metaphor is sometimes conventionalized in the language of the speech, so the speech itself is naturally complemented by the metaphoric gesture. However a recognizable link must exist between the form of the gesture and the meaning of the speech being said.

- **Deictic Gestures**

Deictic gestures are a type of gestures where the idea is to locate or mark the location, of certain aspects or elements in the speech of the narrator. The aspects or elements used in the speech can be physical or non-physical entities.

An example of physical entity referenced by an deictic gesture is a person talking about the characteristics of a paint and pointing an hand to it. Here, the audience will have a speech reference, as well as a "reference" to the location of the paint.

An example of non-physical entity referenced by an deictic gesture is a person talking about another person looking for something in a table and at the same time, moving his hand from the right to the left of the table.

Typically, deictic gestures identify or map speech entities in the space between the speaker and the listener(s), as they are referenced by the speaker. In this gestures usually the speaker can make use of fingers or even hands to map or locate the

entities referenced in the speech.

- **Beat Gestures**

Beat gestures are a type of gestures where typically there are rhythmic beating movements of fingers, hands or arms. These gestures can be as short as a single beat or as long as it is needed for a certain context.

Beating, with or without repetition, is a good way to emphasize and complement a speech that illustrates, for example, some kind of action. Beat gestures may signal that a information is valid or invalid, may signal the feelings of the speaker, may signal a object movement, and others.

Beatings and repetitive actions "call" to primitive feelings or patterns, and can vary in sense according to the context. A beat gesture usually creates emphasis and grabs attention to the source of the gesture.

Note:

For the development of this thesis, the gestures that were considered for study and exploration were the dynamic gestures, in particular the propositional gestures (conscious gestures). This type of gestures is relevant to the work developed, because with a simple set of gestures a set of connections to a set of actions or functionalities can be made, implementing that way a meaning to the execution of the gesture.

2.1.3 Procedural Memory

Procedural memory is a type of long-term memory, in particular implicit/automatic memory, that is acquired during life, used unconsciously, and can be especially affected by thoughts and behaviors [25,26]. See Figure 2.4, where the classification of the long term memories is presented.

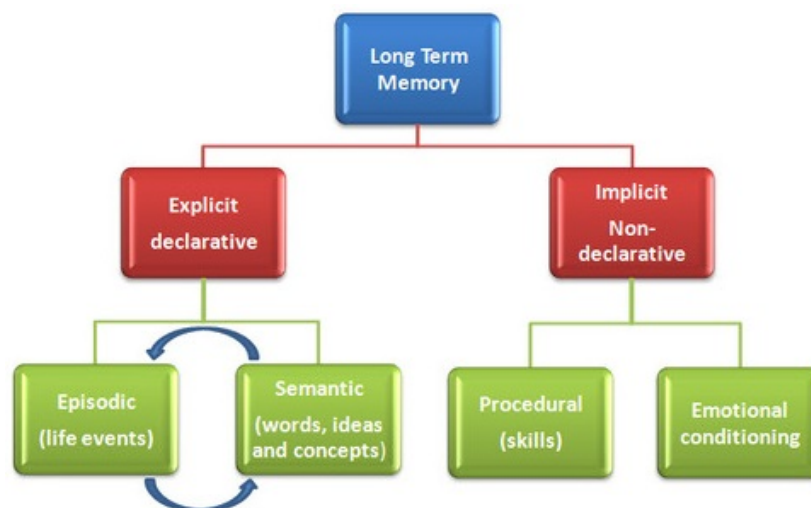


FIGURE 2.4: Memory classification scheme [27]

This type of memory drives the way that people perform a huge set of processes in an unplanned way, below the level of conscious awareness, being usually known as the "Knowing How" memory because it is connected to the personal skills and the how to

do things.

Procedural memories, are called when needed and are automatically retrieved and utilized for the execution of procedures either involved in cognitive or motor skills [25,28].

Procedures like walking, talking, riding a bike, tying the shoes, calculating directions, waving to someone, reading a book, and many others are based on procedural memories [29,30].

These memories are also particularly interesting because the process of acquisition is through procedural learning, consisting mainly in the repetition/practice of a complex activity over and over again until all the relevant neural systems work together to produce the activity or procedure almost automatically.

Another important concept is that procedural memory can be divided into 3 types being them the motor, perceptual, and cognitive memories [29].

2.1.4 Brain and Procedural Memory

The human brain is one of the main organs of the human body, in fact is the main organ of the human central nervous system.

In the human brain, several components are engaged in learning motor and cognitive skills, like the prefrontal cortex, parietal cortex and cerebellum [31,32].

The cerebellum is especially important, because it is involved in balance, motor control and also in some cognitive functions such as attention, language, and emotions as well as procedural memories processing.

The prefrontal cortex, part of the frontal lobe, is known for playing an important role in the processing of short-term memories and in the retaining process of long-term memories which are non task-based.

The parietal cortex is involved in the integration of sensory information from the various senses and in the manipulation of objects(determining spatial sense and navigation).

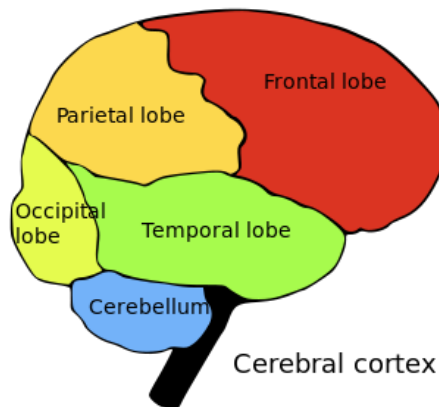


FIGURE 2.5: Brain structure scheme [33]

While every person has almost all of the neurons that need for life at birth, they have to be taught or programmed through lifetime with different life experiences and practices to perform tasks. With this experiences, tasks such as seeing, hearing, walking, talking, writing and many others are possible [34].

In the case of the procedural memories, they can be simple or complex, but are always formed when repeated signals reinforce synapses. A procedural memory can

be as basic as forming a connection between two nerve cells in the fingertip, or more complex, taking more time to be acquired.

2.1.5 Gestures and Procedural Memory

Gestures, learned using a training procedure, can be relevant in automotive HMIs, because they are typically based on procedural memories, so the level of attention and conscious needed to perform these gestures is lower than other actions or activities used in regular HMIs that need a considerable more visual effort, conscious level and attention [35, 36].

With the use of gestures, the driver can potentially be more focused on the road and on the car driving itself. Besides this, the impacts of a gesture recognition system must be studied with a real system, because different implementations and designs typically lead to different impacts in terms of intuitiveness, learnability, satisfaction and distraction.

The study of this impacts will not be covered in this thesis but a good variety of studies are already available for different systems [37–39] and can be used as a mark in future work.

It is important to specify that not all of the studies agree that the use of gestures reduce distraction, increase the intuitiveness or the functionality of an automotive HMI, but a fairly good percentage of the studies and market analysis support this assumption. The interest that exists in many different areas, such as the automotive area, in gesture interfaces is a good measure of the current support for this assumption.

2.2 Human Machine Interfaces

The technological evolution allowed great advances for the humanity. Advances in product design, development and manufacturing contributed to this evolution.

Now, more and more devices and products come to the market with a higher level of complexity, functionality, performance, and at the same time with competitive costs.

In the more developed societies, people use a huge variety and quantity of products or devices that need a variable level of interaction. Devices like TV's, laptops, tablets, smartphones, washing machines, refrigerators, and many others, have some kind of Human-Machine interface.

Besides this evolution and invasion of technologies in our lives, the evolution in the Human-Machine interfaces was more limited and phased, but in the last years, advances in technology and product development, allowed a proliferation of new interfaces especially in the area of consumer electronics, that are now migrating to other areas, thus expanding, the ways that people interact with important devices in their life's.

The use of gestures as form of interaction was until a few years ago almost a Sci-fi concept or idea, but the technological evolution turned the use of gestures possible and somehow desirable in modern HMIs.

Today the use of gestures is very diverse, starting from medical equipment to dedicated gaming equipment.

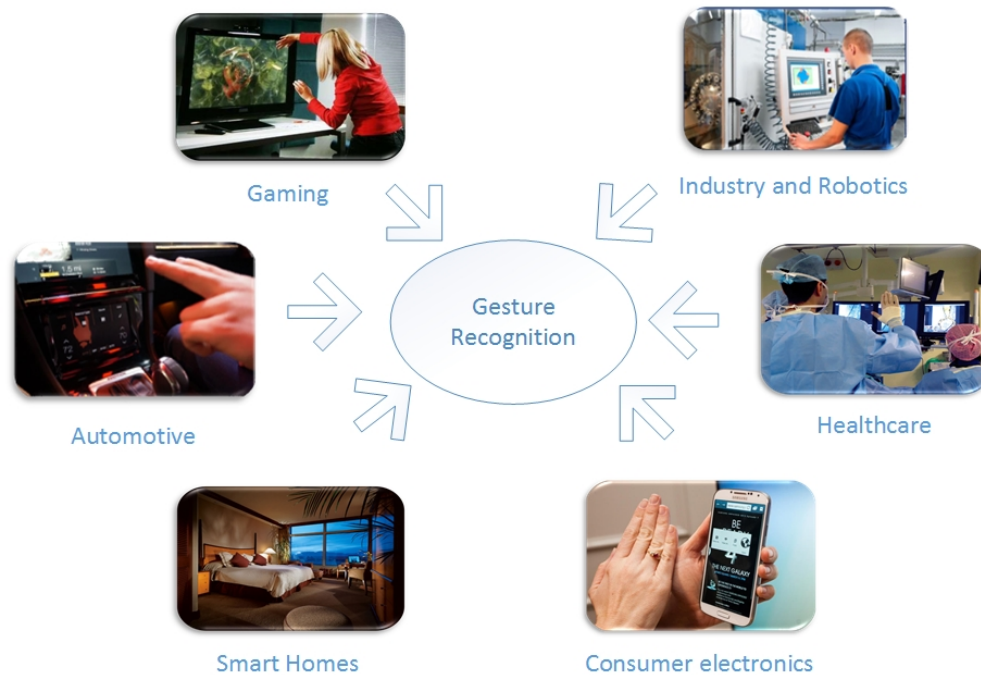


FIGURE 2.6: Example of gesture-based HMIs.

Like it is possible to see in figure 2.6, the use of gestures is very diverse in HMIs, and the previous image shows only a few of the existing examples.

2.2.1 Gesture-Based Automotive HMIs

The migration of technology and forms of interaction is remarkable in a diverse variety of areas. Nowadays, the automotive area is one of the most evolving areas mainly because of the "outdated" image of the in-car technology and forms of interaction.

Changes in various areas, notably in consumer electronics technologies, are molding the way that an automotive HMI is seen and also how it must be idealized in the future.

People, are now demanding a facelift in the technologies on the in-car environment and in the forms of interaction. The automotive industry is trying, at a fast rhythm, answer these demands by exploring new technologies, ideas and concepts.

Nowadays, the automotive area is facing many challenges, and certain trends are now arriving to the scenario. Trends like seamless smartphone integration, haptic feedback, gesture controls, autonomous driving, IoT support and many others are now being developed, implemented and validated.

Gestures are gaining more popularity as a new way of interaction. Unfortunately, the difficulty in building gesture interfaces and gesture controlled systems has prevented such systems from being totally explored.

The case of the automotive HMIs is distinct from others, because it is a special area in terms of challenges, requirements, functionalities and especially users. This challenge of building automotive gesture controlled HMIs is now being accepted by different companies and entities.



FIGURE 2.7: Visteon's Horizon cockpit concept (A) [40] and 2016 BMW 7 Series Gesture Control(B) [41]

Nowadays, companies like Google, Microsoft, Ford, VW AG, BMW, GM, Hyundai-Kia AG, Toyota, Nissan, Mercedes-Benz, PSA, Visteon, Gesturetek and many others are investing in the area. A simple proof of this is the quantity of concepts and patents being registered by different entities like for example Google, Gesturetek or Ford.

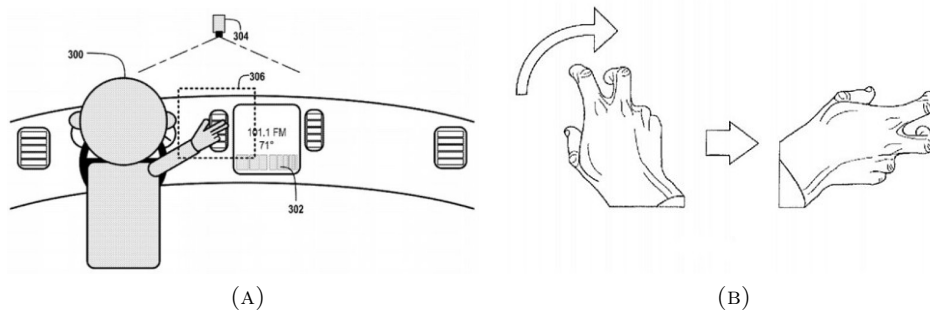


FIGURE 2.8: Google patent proposal for gesture based multimedia control (A)[42] and Ford patent proposal for HVAC control using gestures (B)[43]

These patents, especially Google and Ford patents, propose a set of concepts and relations between gestures vs. functionalities in the use of gestures as well as certain gesture detection techniques. These patents try to shape the way that the control of the multimedia system, HVAC system, navigation system, and others is done.

For a effective and natural gesture based interface between a person and a machine, it is very important for the gesture recognition system to distinguish "true" gestures from non-gestures such as the "usual" hand movements used for grabbing objects, control the radio, control the AC, and others. This capacity will avoid restricting how people place or move their hands when they are not doing any purposive gestures. This concern is especially relevant to all the players which are investigating the use of gestures in automotive HMIs.

2.3 Usability

The Nielsen Norman Group, from which Jakob Nielsen [44] is a part of, describes usability as "a quality attribute that assesses how easy user interfaces are to use". The word usability also refers to methods for improving the ease-to-use factor during the design process. It is important to realize that usability is not a single property of metric, instead it is an association of multiple components and it is traditionally associated

with five attributes or metrics being them: learnability, efficiency, memorability, errors and satisfaction.

Learnability signifies how quickly a new user can begin efficient and error-free interaction with a system. This metric is important for a good "first image" of a system and also to the long-term use of the system.

Efficiency is a useful metric, because it measures how quickly can the users perform a certain set of tasks. A efficient system is a system, that after a period of learning, allows a high level of productivity to the users.

Memorability is a metric, useful to "measure" how the users remember the "how to do the things" after a period of non use of a certain system. This metric is useful to measure the knowledge that needs to be acquired for each time a system is used.

The quantity and variety of errors is a important metric because it is important to know how many errors a user makes, how severe they are and how easily can the user correct them. Ideally a system should have a low error rate and should not have severe errors.

Satisfaction is important, but relative and subjective. The definition of satisfaction is not consensual or concrete. Many authors and entities consider different definitions. A system should be pleasant to use, allowing the users to be satisfied when using it. Users must use a system and at the same time like to use it.

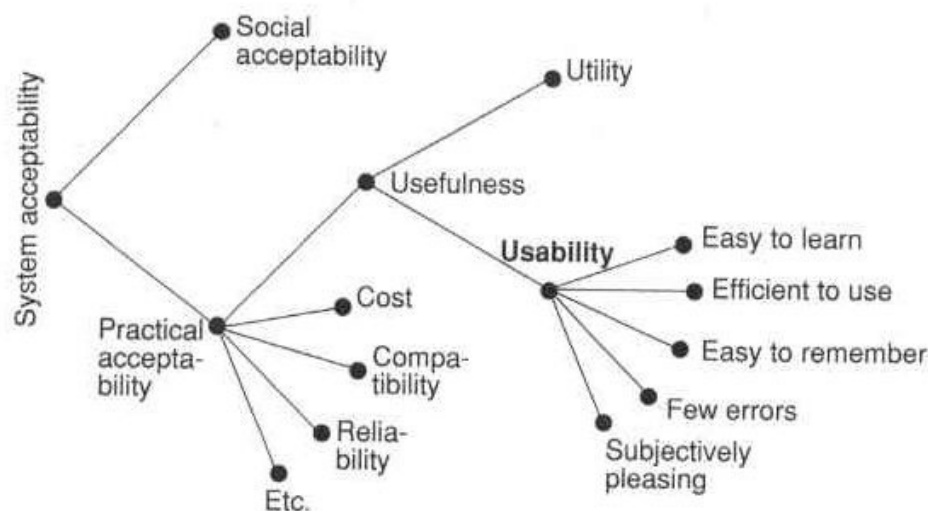


FIGURE 2.9: Model of the attributes for system acceptability [45].

The International Standard (ISO), particularly the ISO 13407:1999 and 9241-11, cover the area of ergonomics of human-hand interaction. The ISO 9241 is concerned with the usability as a wide quality objective in system design, and defines usability as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use".

Following this standard, Usability should be based on the following objectives:

- Effectiveness
- Efficiency
- Satisfaction

This standard also emphasizes that usability is independent on the context of use and that the level of usability will be dependent on the specific circumstances in which a

product is used. The context of use is established by the users, tasks, equipment, and the physical and organizational environments.

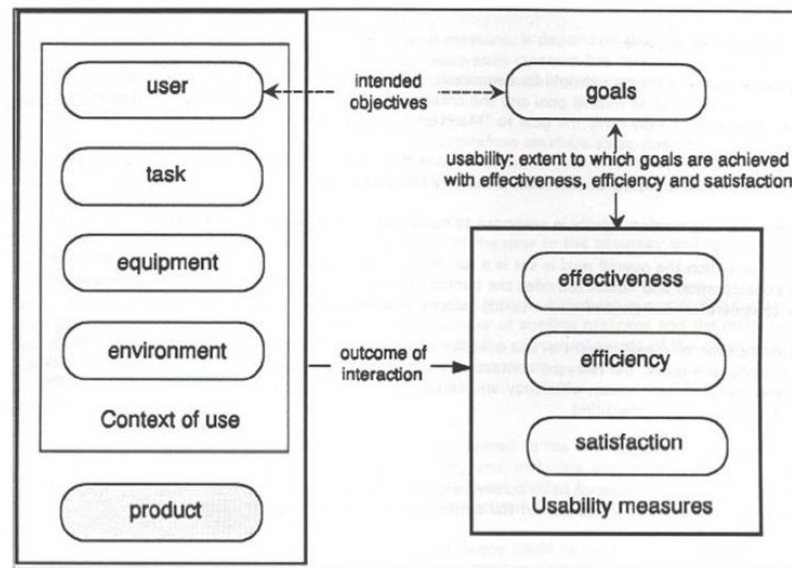


FIGURE 2.10: Model of attributes for system design using ISO standards [46].

2.3.1 Problem of Usability in Automotive HMIs

Vehicles are getting safer, more efficient, smarter and more complex. A car, nowadays, can be a very complex system, either in terms of components, functionalities, and in particular, in terms of interfaces or forms of interactions.

Today, cars are becoming a showroom for different technologies and concepts, because the users demanded an evolution in the car "environment", especially in terms of multimedia content and technological solutions. These demands are a fruit of the expansion and massification of some trends related to many areas, but mainly in the consumers electronics area.

Nowadays, the "common" functions of a driver are not only to drive and be aware of the exterior environment. A driver can be driving, interacting with personal devices, using a certain software application, taking a phone call, searching for a location in the navigation system or expecting an sms or email. Basically the car has become a not only a mean of transportation but also a mean of communication and entertainment.

This reality is helpful to the driver, but at the same time dangerous, because all the "loss of attention" and "focus on the road" of the driver increases the risk of accidents.

Besides this, other problems like the quality of the design and implementation in the automotive interfaces are in many cases questionable, leading to non-intuitive and non-efficient interfaces.

Today, the need and desire to have a car with a safer, friendlier and at the same time more powerful HMI is a very important subject to researchers and automotive companies, who are now investing much in this area. For this reason, the need for a multimodal approach in the design of automotive HMIs is flagrant. Another impulse factor to researchers and automotive companies is the surprise factor to the clients and

users. This is a great promotional/marketing tool if the user experience(UX) is well designed, making the investment in technology acceptable or desirable.

2.4 Gesture Recognition Technologies and Devices

Nowadays, gesture recognition is almost everywhere from smartphones, laptops, tablets, smart TV's and gaming consoles to industrial equipment, medical equipment, robotics and others. With the expansion of gesture recognition applications and concepts, lots of technologies are being explored and others are still to be explored, both for commercial or academic purposes.

The automotive area is a special field of application that until now is not fully explored, so there is an interest to test different technologies and concepts using gesture recognition.

The list of technologies that can be used to track or detect gestures has many options such as touchless capacitive technology, ultrasound technology, radar technology, infrared technology, laser technology, 2D/3D cameras systems and even wearable devices. The next topics will present a brief perspective about the sensors and market acceptance for each one of the technologies.

2.4.1 Capacitive Sensors

Since any living organism produces an electric field that is directly caused by the cell activity and ionic currents in the nervous system [47], in the case of the humans, it is possible to sense the influence of the human body in two ways, either using external electric fields and measuring the influence of the human body movement, or coupling the body to an electric transmitter and measure the resulting field.

Capacitive touchless sensors are a type of contactless sensors capable to detect metallic and non-metallic objects, more precisely conductive or dielectric materials. These sensors use the variation of capacitance between the sensor and the object being detected (ex: human hand) to track movements or gestures. There are three main types of touchless capacitive sensing modes [48, 49]:

2.4.1.1 Shunt mode

In this mode, if an object approaches the two conductive materials(receiving and transmitting electrodes), the capacitance between them will change (will decrease), because both the electrodes are generating a static field, that in the presence of an object is grounded, reducing that way the energy stored. The advantage of this mode is that one single emitting electrode can be used with more than one receiver, increasing the sensing capability with limited hardware.

A good example of the previous explanation is the figure below, Figure 2.11.

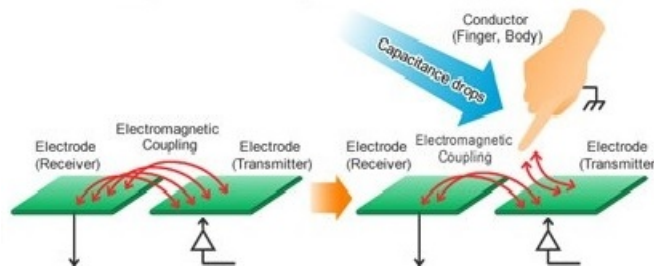


FIGURE 2.11: Shunt Mode Capacitive Sensing [50]

2.4.1.2 Load mode

In this case the capacitance is measured from a single electrode with respect to ground. Here if the object approaches to the electrode, for example a human hand, the capacitance of the electrode will change according to the distance. In this sensors, higher distances between the sensors and the objects means lower capacitance and lower distances to the objects mean higher capacitance. This happens because the electrode is inducing an oscillating field that is affected by any approaching grounded objects, where if the object distance decreases, the capacitance of the system increases.

A good example of the previous explanation is the figure below, Figure 2.12.

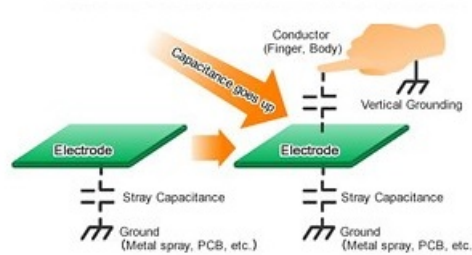


FIGURE 2.12: Load Mode Capacitive Sensing [51]

2.4.1.3 Transmit mode

In this case, two electrodes are used (a transmitter and a receiver). The transmitter is coupled to a conductive object, such as a human hand, and with the variation of the distance to the object, the electric field generated with respect to the receiver electrode will also vary, increasing or decreasing the effective range of the transmitter.

In this sensors, higher distances between the sensors and the objects means lower capacitance and lower distances to the objects mean higher capacitance.

A good example of the previous explanation is the figure below, Figure 2.13.



FIGURE 2.13: Transmit Mode Capacitive Sensing [52]

In terms of market acceptance, the capacitive technology is well accepted mainly because it can be used to create a system with 1 to 3 dimensions of sensing, which is important for a gesture detection system.

Other factors are also important like a good detection range, cost, resolution, reliability and power consumption. This technology have also some problems, mainly in terms of sensitivity, that is strongly dependent of the distance to the object and depends of factors like temperature and humidity. Besides that, possible problems of false triggering (object interference) can occur.

2.4.2 Ultrasonic Sensors

Ultrasonic sensors are another type of contactless proximity and gesture detection sensors, and their application goes from simple distance detection to 3D gesture detection.

Ultrasonic sensors work by sending ultrasound signals from speakers to the air, then the signals are recorded by microphones and interpreted. To proceed with the calculation of valid distances/positions, the sound waves should be reflected, as an echo signal or reflected wave, by the objects/surfaces.

To calculate the distance, the recorded signals are analyzed and the time of measurement is considered, basically it's a time-of-flight analysis. This technology can, for example, be used to recognize hand gestures/movements in a very similar way to how bats use echolocation to navigate.

A good example of the previous explanation is the figure below, Figure 2.14, which shows the working principle of an ultrasonic sensor.

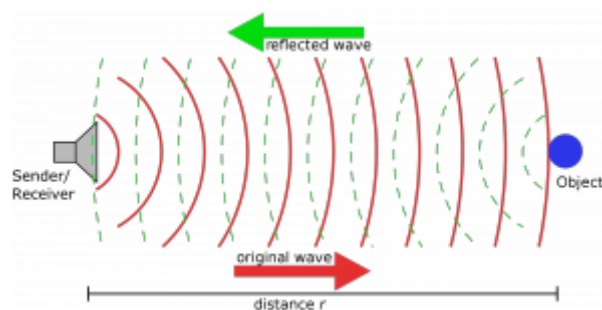


FIGURE 2.14: Ultrasound sensor working principle [53].

In terms of market acceptance, this technology is well accepted mainly for distance detection because of factors like a very good detection range, invariable performance with object color, good cost, good mechanic reliability and good power consumption. This technology is still under exploration in terms of 2D and 3D gesture detection systems, but there are some concepts already developed.

This technology has also problems because, in order to have a good performance, the surfaces/objects must have an acceptable size and be as flat as possible. Another problem is that changes in the environment, such as temperature, pressure, humidity, air turbulence, and airborne particles could affect ultrasonic sensing response. The final problems are that surfaces with low density materials (like cloth or foam) tend to absorb sound, and that ultrasonic sensors require time to stop ringing after each transmission burst before be ready to receive the returned echoes, slowing the data acquisition.

2.4.3 Radar Sensors

Radar is a technology similar to ultrasound technology, in terms of description because both rely on the propagation and reflection of wave signals. Radar by contrast, works not with sound waves, but with electromagnetic waves (with a band of frequencies from few MHz to GHz).

Like with ultrasonic signals, the electromagnetic signals bounce off the objects and travel at a known speed, approximately the speed of light, much faster than ultrasonic waves, which travel at the speed of sound. The electromagnetic waves previously emitted by a transmitter are reflected if they meet an electrically leading object/surface. If these reflected waves are received again at the place of their origin, then that means

that an obstacle is in the direction of propagation.

The figure below, Figure 2.15, shows the working principle of a radar sensor.

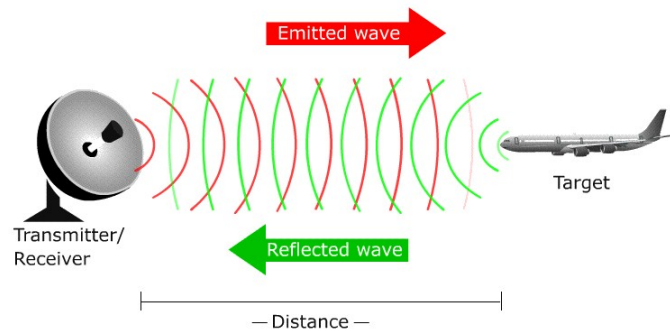


FIGURE 2.15: Radar sensor working principle [54].

In terms of market acceptance, this technology is well accepted mainly to determine range, angle and velocity of diverse means of transportation, guided missiles, weather formations and terrain, because of factors like a great detection range, good reliability, good resolution and a good capacity to operate in environments with varied temperature and pressure (being well suited for operation in vacuum or high pressure environments). An important factor is that the target/object materials dielectric constant should not be low (like in non-conductive materials) because in that case the electromagnetic waves will not be sufficiently reflected, and will pass through the objects. This technology is still under study and exploration in terms of gesture detection systems, but there are some prototypes and concepts already in development.

The three main problems for application in gesture recognition systems are the cost, the state of the technology (under development for small applications) and the availability of the majority of the market offers.

2.4.4 Infrared Sensors

Infrared sensors are another type of contactless sensors that rely on the propagation and reflection of wave signals, more precisely electromagnetic waves in the infrared spectrum. There are two main types of IR sensors, the active IR sensors and the passive IR sensors.

2.4.4.1 Active IR Sensors

Active IR sensors work by using specific light sensing elements, sensitive to electromagnetic waves in the Infrared spectrum, and infrared light sources. Basically infrared light sources produce a beam of light, with a specific FOV, that can be reflected in the presence of objects/surfaces. The reflected light is detected by the light sensing element typically with an intensity dependent of the proximity to the objects/surfaces. Basically higher distance means lower intensity of light reflected and lower distance means higher intensity.

There is a type of active IR sensors, that do not rely on the intensity of the reflected light but on the times it takes to reflect the light itself in the presence of objects/surfaces. These sensors are called ToF (Time-of-Flight) sensors.

The figure below, Figure 2.16, shows the working principle of an active IR sensor with or without the presence of an obstacle/object.

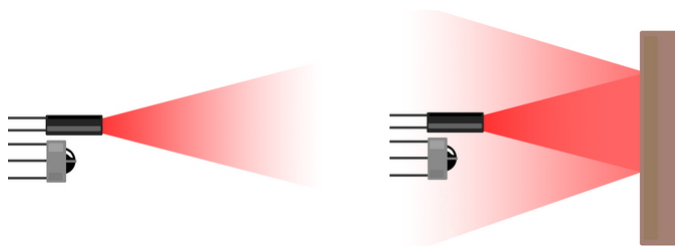


FIGURE 2.16: Active IR sensor working principle [55].

2.4.4.2 Passive IR Sensors (PIR)

Passive IR sensors work by using specific light sensing elements, sensitive to electromagnetic waves in the Infrared spectrum, but in opposition to the active IR sensors do not use IR light sources. These sensors detect and measure the IR radiation emitted from objects or surfaces in their FOV. A passive infrared sensor therefore reacts to objects or surfaces that radiate a particular temperature, such as humans or animals for example.

In practice, a passive infrared sensor is mainly intended for use in a conditioned environment, where the interferences/noises are lower, such as inside buildings, where is used for motion tracking or surveillance purposes.

The figure below, Figure 2.17, shows the working principle of a passive IR sensor.

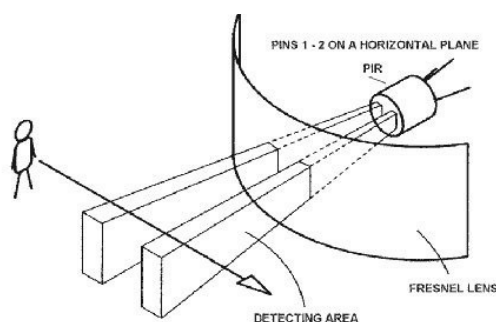


FIGURE 2.17: Passive IR sensor working principle [56].

Like the previous image shows, the sensing device has "detection slots" (in this case 2), that can sense the variation of IR radiation. Then if a warm body passes by a slot, the quantity of IR radiation changes, and this is detected by the sensing device. The same happens in the other slot(s), if the object continues its movement.

2.4.4.3 ToF (Time-of-Flight) IR Sensors

A particular type of active IR sensors is the ToF sensors, which unlike most IR proximity and gesture detection sensors, do not rely on the reflected light intensity or reflected light angles to determine range, but instead use a precise clock to measure the time it takes to the light bounce back from a surface/object after being emitted. Typically for this reason these sensors tend to be more accurate and more immune to noise.

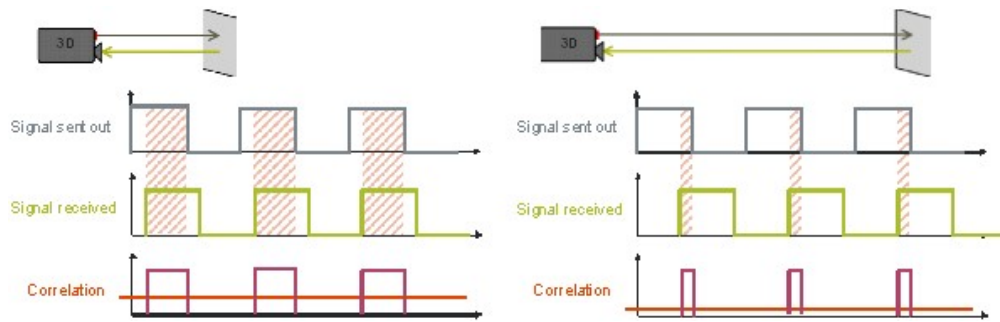


FIGURE 2.18: ToF IR sensor working principle [57].

Like the figure 2.18 shows, the difference in the emitted and reflected signals (phase shift) is higher for greater distances and lower for small distances (like the correlation signal shows).

In terms of market acceptance, IR sensing technology is well accepted in industrial applications, robotics and surveillance systems. For proximity and gesture recognition, this technology can be used from 1 to 3 dimensions of sensing. Other factors especially relevant are a good detection range, good cost, good reliability and good power consumption.

IR sensing technology have also some problems, mainly in terms of accuracy, linearity, color sensitivity and external light source interferences. Some of the problems of IR technology are less problematic in the case of ToF sensors, like the accuracy, linearity and color sensitivity.

2.4.5 Laser Sensors

Photoelectric sensors, using Laser light emitters, are another type of contactless proximity/distance sensors that rely on the propagation and reflection of light signals. A laser is a device that emits light through a process of optical amplification based on the stimulated emission of electromagnetic radiation.

Lasers can be used, using various techniques, to measure distance or displacement without physical contact, being used in a considerable amount of applications. Some of the most important techniques, used for distance/proximity measurements are:

- Triangulation - geometric process, used to determine the location of a point/object from a known perspective by forming triangles from known points. This technique uses time measurements between the emitted and received pulses to calculate the distance;

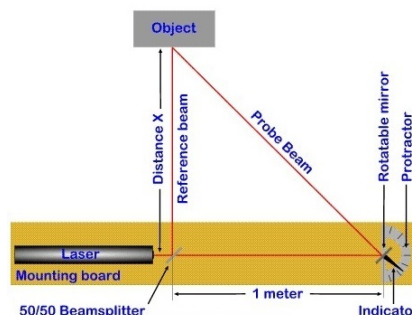


FIGURE 2.19: Laser based sensing - triangulation technique [58].

- TOF - laser pulses are sent to the target and are reflected back. During this period, the time is accurately measured and used to calculate the physical distance between the emitter and the object.

In terms of market acceptance, laser ranging technology is a well accepted technology with applications from robotics, military systems, 3D modeling to measuring tools.

For proximity/distance detection, this technology is popular because of the long detection range, average cost, size, reliability, accuracy, precision and power consumption.

This technology has some problems, mainly in terms of the lack of measurement (reflection) in transparent objects and high interference under direct light sources, like sunlight.

2.4.6 Cameras Systems

The expansion of the use of camera devices in our daily life, the technological advances and the efforts of some companies, turn the camera devices, well suited for applications with gesture recognition.

With a huge variety of camera technologies and solutions, there is a space in this technology for almost every project needs.

In terms of camera solutions applied to gesture recognition, there are two main types of systems to consider, the 2D and 3D camera systems.

2.4.6.1 2D Camera Systems

In terms of 2D camera systems, they vary from regular cameras to specific or professional cameras, but their use for gesture recognition is somehow limited. The quantity of information that can be collected is limited (no depth sensing) and the capacity to detect gestures is somehow limited (typically simple hand gestures, finger gestures and motion tracking).

These cameras are typically very sensitive to light variations, slower than other cameras (TOF) and have a middle-high power consumption. The main advantages of this systems are the simplicity, size and cost.

2.4.6.2 3D Camera Systems

In terms of 3D cameras (depth-sensing cameras), three types were studied: the structured-light cameras, ToF cameras and stereo cameras.

• Structured-Light Cameras

Structured-light cameras work by projecting a narrow band of light, typically a set of stripes, onto a 3D surface or object.

With the influence of the object shape, the projected patterns will suffer distortion. Due to this distortion, the surface shape can be reconstructed onto an image, where each pixel contains a value proportional to the distance of the object or surface. Usually to speed up the process of scanning, many patterns are projected at once, typically using IR or laser projectors.

The figure below, Figure 2.20, show respectively a schematic representation of a structured-light camera system and a 3D object reconstruction process.

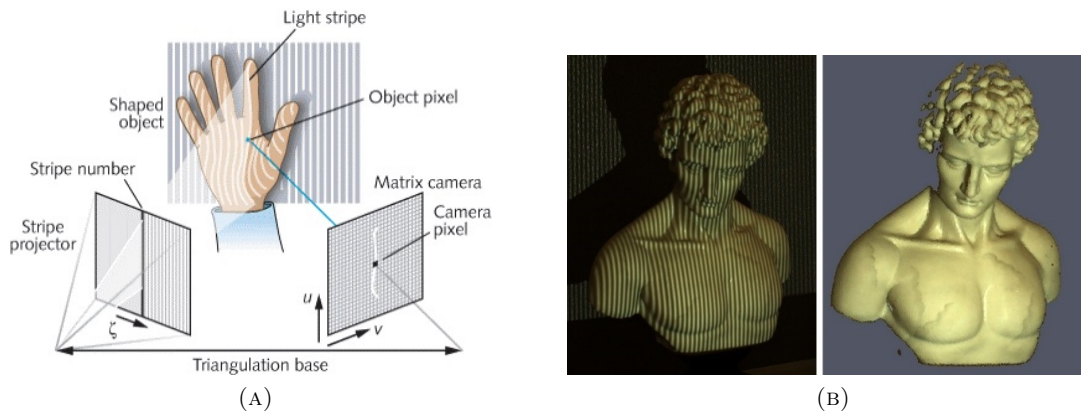


FIGURE 2.20: Structured Light System Schematic(A) [59] and 3D object reconstruction based on Structured Light system(B) [60]

• ToF (Time-of-Flight) Cameras

ToF (Time-of-Flight) cameras typically work by projecting infrared/laser light pulses, onto a 3D surface or object, but with a different purpose than in structured light. Here the aim is to project light pulses and through time measurement and light reflection calculate the depth of the 3D surface or object. The resulting image (range image) has pixel values proportional to the object physical distance (calculated using the time-of-flight time).

The figure below, Figure 2.21, shows respectively the working principle of a ToF camera system and a 3D scenario reconstruction/scanning.



FIGURE 2.21: ToF(Time of Flight) System Schematic (A) [61] and 3D object reconstruction based on a ToF system(B) [62]

• Stereo Cameras

Stereo cameras are a type of depth-aware cameras with typically two lenses and image sensors. The objective of this configuration is to allow the camera to simulate human binocular vision, giving the ability to capture 3D images. The usual distance between lenses is about the same distance of the human eyes and its around 6.35 cm, but can vary in different systems.

Stereo cameras are used to capture 3D images, using a process of image building that uses two 2D images from different angles and by performing a full comparison of the images, the relative depth information can be obtained from the disparities between the images.

The figures below, Figure 2.22, shows respectively a schematic representation of a stereo camera system and a 3D object reconstruction.

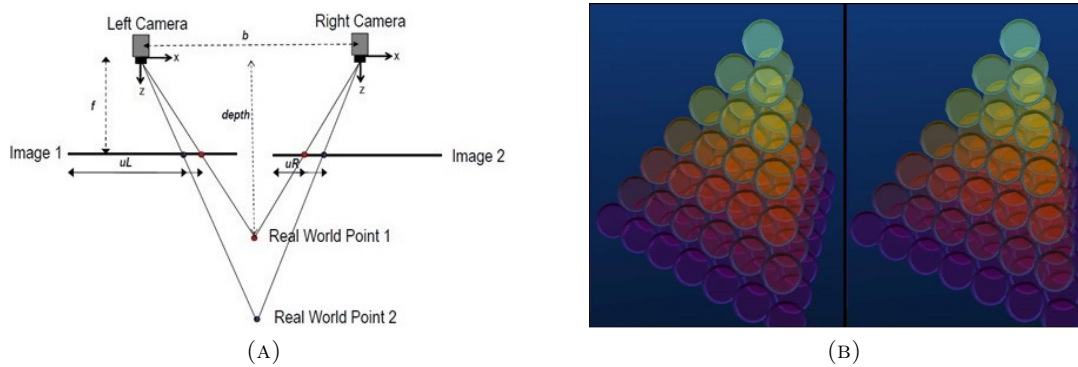


FIGURE 2.22: Stereo camera schematic representation (A) [63] and computer rendered stereo image/object(B) [64]

• 3D Camera Technology Comparison

These three main 3D camera technologies have different characteristics, advantages and disadvantages. The following table, Table 2.1, shows a comparison between the three camera technologies.

Considerations	Stereo Vision	Structured-Light	Time-Of-Flight
Software Complexity	High	Medium	Low
Material Cost	Low	High	Medium
Compactness	Low	High	Low
Response Time	Medium	Slow	Fast
Depth Accuracy	Low	High	High
Low-Light Performance	Low	High	High
Bright-Light Performance	High	Low	High
Power Consumption	Low	Medium	Scalable
Range	Limited	Scalable	Scalable

TABLE 2.1: Comparison between 3D camera technologies

2.4.7 Wearable Devices

Wearable devices are nowadays a valuable technological solution for millions of people. Devices like smartwatches, smart bands, smart rings, and many other smart controllers or devices are gaining more popularity. The main interest of these devices is that they act like an extension of the human body, and usually pack a diverse amount of different sensors like IR sensors(used in heart rate monitoring), accelerometers, and gyroscopes. There are smart controller's, like the MYO armband, that monitor the muscle activity of the human arm, turning the movements and gestures almost as natural as using other contactless alternatives like infrared or capacitive sensors, where the arm is free of controllers.

Wearable devices have also some problems like battery life, size (in some cases), flexibility and accuracy.

The figure below, Figure 2.23, shows respectively the MYO armband, Sony Smartwatch 3, and Smarty Ring, three examples of wearable devices.



FIGURE 2.23: MYO armband (A) [65], Sony smartwatch 3 (Sony SW3) (B) [66], and Smarty Ring (C) [67]

2.5 Fundamental Principles of Infrared Technology

In this section, it will be presented a set of theoretical concepts related to active IR sensors, since it is the type of sensor that will be used in this thesis project.

The fundamental principles, will be divided in two parts, the first is related to generic concepts and the second is related to ToF sensors.

2.5.1 Infrared Radiation

Infrared radiation (IR) is a type of electromagnetic radiation as are radio waves, ultraviolet radiation, X-rays, Gamma-rays and microwaves [68–70].

Infrared radiation was discovered in 1800, by the British astronomer Sir William Herschel, who discovered a type of invisible radiation with lower energy than red light, by detecting its effect on a thermometer [69].

Infrared radiation occupies a portion of the electromagnetic spectrum and has wavelengths longer than visible light wavelengths, but at the same time smaller than microwaves.

For the human eyes the IR radiation is not visible, but people encounter it in their daily life, although most of it goes unnoticed. Human eyes cannot detect differences in infrared energy because they are primarily sensitive to visible light (electromagnetic waves with wavelengths from 400nm to 700nm).

People cannot see it, but can feel it as heat since IR radiation is one of the three forms of heat transference between two elements, being the other convection and conduction. In the reality, every element with a temperature above $-268\text{ }^\circ\text{C}$ emits IR radiation [70].

2.5.2 Infrared Spectrum

The electromagnetic spectrum is generally divided into seven regions in order of decreasing wavelength and increasing energy and frequency.

The region from $0.75\text{ }\mu\text{m}$ to $1000\text{ }\mu\text{m}$ is usually considered the infrared radiation region.

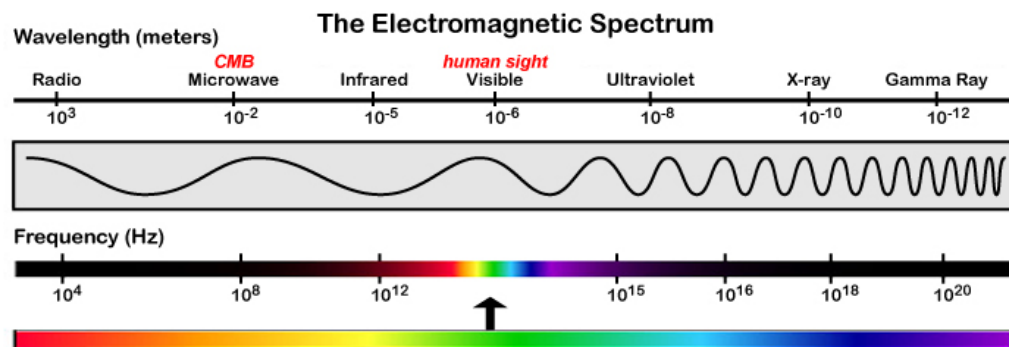


FIGURE 2.24: Electromagnetic spectrum [71]

Infrared Spectrum Regions

The infrared spectrum is an important region of the electromagnetic spectrum, and it is a region with an important interval of wavelengths, because of this the infrared spectrum is divided in several regions.

The regions of the infrared spectrum to consider aren't a closed topic, because different entities and organizations have different standards.

The main standards to consider are the following:

Commonly used scheme

Region	Abbreviation	Wavelength (μm)	Frequency (THz)	Temperature ($^{\circ}\text{C}$)
Near-infrared	NIR	0.75 to 1.4	214 to 400	1797 to 3591
Short-wavelength infrared	SWIR	1.4 to 3	100 to 214	693 to 1797
Mid-wavelength infrared	MWIR	3 to 8	37 to 100	89 to 693
Long-wavelength infrared	LWIR	8 to 15	20 to 37	-80 to 89
Far-infrared	FIR	15 to 1000	0.3 to 20	-270.15 to -80.15

TABLE 2.2: Commonly used scheme of regions in the IR Spectrum

International Organization for Standardization - ISO 20473 scheme

Region	Abbreviation	Wavelength (μm)
Near-Infrared	NIR	0.78 to 3
Mid-Infrared	MIR	3 to 50
Far-Infrared	FIR	50 to 1000

TABLE 2.3: ISO 20473 regions of the IR Spectrum

International Commission on Illumination (CIE) scheme

Region	Abbreviation	Wavelength (μm)	Frequency (THz)
IR-A	NIR	0.70 to 1.4	215 to 430
IR-B	MIR	1.4 to 3	100 to 215
IR-C	FIR	3 to 1000	0.3 to 100

TABLE 2.4: CIE regions in the IR Spectrum

2.5.3 Scientific Laws

There are different types of IR sensing devices, working in specific regions of the IR spectrum but the working properties behind them are mainly governed by three laws.

2.5.3.1 Planck's Law:

Planck's law describes the electromagnetic radiation emitted by a black body in thermal equilibrium at a definite temperature.

A black body is an idealized physical body that absorbs all incident electromagnetic radiation, regardless of frequency or angle of incidence. A black body in thermal equilibrium (at a constant temperature) emits electromagnetic radiation called black body radiation. According to Planck's law, the radiation emitted has a spectrum that is determined by the temperature of the body alone, regardless of the body shape or composition. By contrast, a white body is one that reflects all incident rays completely and uniformly in all directions.

A black body in thermal equilibrium has two important properties:

- It is an ideal emitter because at every frequency, it emits as much or more energy as any other body at the same temperature.
- It is a diffuse/isotropic emitter because the energy is radiated with the same intensity of radiation in all directions.

Considering this law, basically, every object at a temperature T not equal to 0 K emits radiation. Infrared radiant energy is determined by the temperature and surface condition of an object.

The spectral radiance of a body B_ν describes the amount of energy radiated by the body as radiation at different frequencies, and B_λ describes that same amount of energy radiated but measured for different wavelength λ .

$$\text{Spectral radiance, measured per unit of frequency}(\nu): B_\nu(\nu, T) = \frac{2h\nu^3}{c^2} \frac{1}{e^{\frac{h\nu}{k_B T}} - 1}$$

$$\text{Spectral radiance, measured per unit of wavelength}(\lambda): B_\lambda(\lambda, T) = \frac{2hc^2}{\lambda^5} \frac{1}{e^{\frac{hc}{\lambda k_B T}} - 1}$$

where:

$$k_B = k = \text{Boltzmann constant} = 1.38064852 \times 10^{-23} m^2 kg s^{-2} K^{-1}$$

$$h = \text{Planck constant} = 6,626\ 069\ 3(11) \times 10^{-34} \text{ Js} = 4,13566743(35) \times 10^{-15} eV s$$

$$c = \text{speed of light} = 299792458 \text{ m/s}$$

2.5.3.2 Stefan-Boltzmann Law:

Stefan-Boltzmann law describes the power radiated from a black body in terms of its temperature. Specifically, this law states that the total energy radiated per unit of surface area of a black body across all wavelengths per unit time (also known as radiant exitance) is directly proportional to the fourth power of a black body temperature.

Total energy radiated (j^*):

$$j^* = (\sigma)T^4$$

where:

$$\sigma \text{ (Stefan-Boltzmann constant)} = 5.670373 \times 10^{-8} \text{ Wm}^{-2}\text{K}^{-4}$$

T is the absolute temperature of the body (in Kelvin).

Briefly resuming this law, an object with higher temperature will irradiate more energy compared to other with lower temperature.

2.5.3.3 Wien's Displacement Law

Wien's displacement law states that the black body radiation curve, for different temperatures, peaks at a wavelength inversely proportional to the temperature of the body. In a formal approach, Wien's displacement law states that the spectral radiance of black body radiation, per unit wavelength, peaks at the wavelength λ_{max} .

Considering this:

$$\lambda_{max} = \frac{b}{T}$$

where:

$$b \text{ (Wien's displacement constant)} = 2.8977729 \times 10^{-3} \text{ mK} \approx 2900 \mu\text{mK}$$

T is the absolute body temperature (in Kelvin).

The world is not full of black bodies, instead of this, it is composed by various types of radiating bodies like water, rocks, plastic, wood, etc.

The relationship between the two is given by emissivity (ϵ).

$$\epsilon = \left(\frac{M_e}{M_e^o} \right)$$

where:

M_e = Radiant exitance of a body/surface;

M_e^o = Radiant exitance of a black body at the same temperature;

Emissivity is extremely dependent of the surface color, roughness, moisture content, degree of compaction, field of view, viewing angle and wavelength. For this reason, emissivity measurements for many surfaces are compiled in many documents and books.

2.5.4 Time-of-flight Concepts

As previously stated, ToF sensors work by emitting and receiving light pulses. During the phase between the emission and reception of the light pulses, reflected by the object in the sensor field of view, the time is accurately measured.

The main advantage of this sensors compared to the generic active IR sensors is the precision of measurement using the time basis as referential instead of the intensity of light reflected.

ToF typically work following two methods, the pulsed wave method and the continuous wave method. These methods are briefly explained in the next topics.

Pulsed Modulation Method

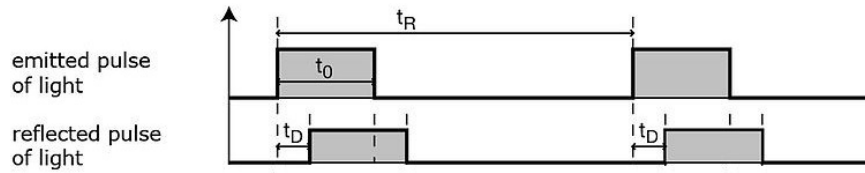


FIGURE 2.25: ToF - Pulsed Modulation Method [72]

The pulsed modulation method is a straightforward method and works by using a light source for a brief period t_0 , then the time until the reflected light pulse is received t_D , is accurately measured. Like the previous image shows, see Figure 2.25, the phase shift between the emitted and the reflected waves exist, being measured and translated to the actual real distance.

Time-of-flight calculation:

$$t_D = 2 \times \frac{D}{c}$$

Distance calculation:

$$D = \frac{t_D \times c}{2}$$

where:

c = speed of light $\approx 299792458m/s$

D = physical distance

t_D = time between emission and reception of reflected light pulses

Continuous Wave method

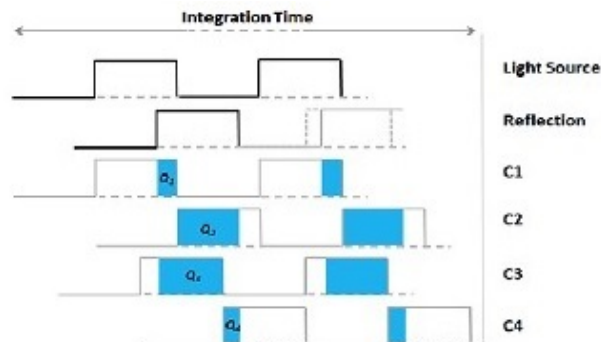


FIGURE 2.26: ToF - Continuous Wave Method [73]

The continuous wave (CW) method takes multiple samples per measurement, as seen in the previous figure (Figure 2.26). Each sample is actually phase-stepped by 90 degrees, so a total of four samples are taken each phase.

Using this technique, the phase angle between illumination and reflection, ϕ , and the distance, d , can be calculated by:

Phase angle calculation:

$$\phi = \arctan \frac{Q_3 - Q_4}{Q_1 - Q_2}$$

Distance calculation:

$$d = \frac{c \times \phi}{4 \times \pi \times f}$$

where:

f = signal frequency

$c \approx 299792458$ m/s

Comparing both methods, it is possible to see a fairly simpler approach using Pulsed modulation method, either in calculation procedures or comprehension.

2.6 Market Forecasts

The automotive HMI area is recently a technological area with great developments, mainly because of the expansion of the vehicle infotainment or In-Car Entertainment (ICE) systems that are used by drivers and passengers almost every second in a car.

Gesture recognition is another area gaining more and more popularity, and the automotive area, in an effort to introduce new technologies and concepts and at the same time solve in-car problems or challenges, is now investing considerable in this area. A proof of this fact, is the increasing number of supportive market forecasts, prototypes or case studies from different car manufacturers and OEM suppliers.

The next topics present some forecasts of the automotive HMI area growth and of the automotive gesture recognition area growth.

Notes:

CAGR - Compound Annual Growth Rate (CAGR) is the mean annual growth rate of an investment over a specified period of time longer than one year.

$$CAGR = \left(\frac{\text{EndingValue}}{\text{BeginningValue}} \right)^{\left(\frac{1}{\#ofyears} \right)} - 1$$

2.6.1 Market Forecasts for Automotive HMIs Growth

Source	Study	CAGR	Period
Technavio	"Global Automotive Human Machine Interface (HMI) Market 2014 - 2019"	11.25%	2014-2019
Sandler Research	"Global Automotive Human Machine Interface (HMI) Market Forecast to 2015-2019"	11.25%	2015-2019
RNR Market Research	"Global Automotive Human Machine Interface (HMI) Market 2015-2019"	7.73%	2015-2019
MarketsandMarkets	"Global Automotive Human Machine Interface (HMI) Market"	10.13%	2012-2017
Transparency Market Research (TMR)	"Global Automotive Human Machine Interface (HMI) Market"	10.4%	2013-2019

TABLE 2.5: Table of forecasts of the HMI Market Growth

2.6.2 Market Forecasts for the Automotive Gesture-Based HMIs Growth

Source	Study	CAGR	Period
Technavio	"Global Gesture Recognition Market in Automotive Sector 2014-2018"	12.8%	2014-2018
Technavio	"Global 2D Gesture Recognition Market 2015-2019"	32.12%	2015-2019
Sandler Research	"Gesture Recognition Market Growth in Automotive Sector 2014-2018"	12.8%	2014-2018
MarketsandMarkets	"Gesture Recognition & Touchless Sensing Market by Technology (Touch-based & Touchless), Application (Consumer Electronics, Automotive, & Others), Product (Biometric & Sanitary Equipment) & by Geography - Global Forecast to 2020"	28.2%	2014-2020
MarketsandMarkets	"Global Automotive Gesture Recognition System Market 2016-2020"	55.44%	2016-2020

TABLE 2.6: Table of forecasts for the gesture-based HMI Market Growth

2.7 Case Studies and Concepts

The automotive market is evolving in terms of technological needs, applications and solutions. Gesture recognition is a promising trend and is one of the solutions to explore.

The automotive manufacturers and suppliers are working to make possible the use of gesture recognition in cars, either for basic or complex gestures applied to a considerable variety of functionalities.

Several car makers and OEM suppliers, are now working to create their own solutions and concepts using gesture recognition. Companies such as Audi, BMW, Cadillac, Continental, Ford, GM, Harman, Kia, Hyundai, Mercedes-Benz, Nissan, SoftKinect, Toyota, Visteon, Melexis and VW, for example, show many efforts and work being developed in this area.

In this section a set of examples are explored to show a small sample of what is being done in the automotive area in terms of gesture-based interfaces.

• BMW iDrive and BMW i Vision Future Interaction Car

In the BMW's iDrive infotainment system with AirTouch, showed at CES 2015, for the first time BMW was considering the use of touchscreen and gesture recognition as viable sources of interface. The gesture recognition system was first presented on the BMW 7 series model of 2016.

The gesture recognition system consists mainly in a 3D TOF camera located at the headlining of the car (near the roof light), with the functionality of monitoring the dashboard space between driver and passenger.

Gestures like finger rotations, one and two finger pointing, and swiping are recognized and used in contexts like audio control, incoming call response and navigation control.

The BMW iVision Future Interaction Car is another concept mockup that shows

how the user interfaces of the future might look like. This system includes a 3D gesture recognition system, similar to the system used in the iDrive, to control communications, information and entertainment functionalities.

- **VW Golf R Touch Concept and VW E-Golf**

VW presented a concept car, in the CES 2015, with an HMI designed to remove the physical/mechanical dials and switches. The usual mechanical switches were replaced by dials, switches and bars with proximity-aware or touch sensing capabilities. An important add-on is the gesture recognition system, implemented with a TOF camera near the roof light, capable to detect mainly one and two finger pointing and swipe gestures, used for purposes like selection on main menu, climate settings control, volume control, mirrors control and others.

VW also presented the E-Golf concept car at CES 2016. This concept car also has gesture recognition capabilities, but this time in a simpler implementation. The system mainly allows the detection of left and right swipes.

The gestures can be used in tasks that vary from the control of the image gallery to the switch between songs. This system uses a cost-friendly infrared sensor, a solution than VW is studying due to the lower implementation costs compared to the previous TOF system.

- **VW BUDD-e Concept**

VW presented a concept car (microbus), in the CES 2016, called the BUDD-e. This concept is one of the biggest bets of VW in the electric mobility and at the same time in the HMI area. This concept car has also gesture recognition capabilities being the gesture recognition system called "Gesture Recognition 2.0". The gesture recognition system uses IR sensors and is used for distinct applications in the interior of the car, in a similar way of the VW E-Golf.

In the exterior of the car, the gesture recognition system recognizes the swiping of the hand to open the sliding door and the foot movement to open the electric tailgate.

- **Hyundai HCD-14 Genesis Concept & Genesis New York Concept**

Hyundai HCD-14 Genesis concept is a cockpit concept with 3D gesture recognition capabilities in the car HMI. The cockpit is able to detect and recognize the driver commands and use them to control the navigation system, audio system, HVAC system, and other infotainment functions including smartphone interaction.

Simple hand gesture can be used to play/pause music, advance to the next track or return to the previous track. Hand gesture recognition can be accomplished with dedicated infrared and camera sensors.

A distinctive feature of this concept is that the gesture interface is not only focused on the driver, since the passenger can also use gestures to interact with the car and the driver for actions like selecting a GPS route, and moving that route to the central console screen, helping that way the driving operation.

Genesis New York Concept is a concept car, presented at 2016 New York auto show, from Genesis (Hyundai Group), that uses a 3D gesture sensor to allow the occupants to switch data between different screen areas. The sensors are used as a "grab & throw" option for content like navigation, audio, personal data and others. The "grab & throw" meaning is that the user grab/selects (swipe up gesture) the content to work with and then throw the content(left/right swipe) to a certain screen area.

- **Visteon Horizon Concept**

Visteon Horizon concept is a cockpit concept testing different technologies and concepts like gesture recognition, dual-layered displays and virtual touch screens (touchpads for the console LCD).

The gesture recognition system is able to detect 3D gestures, using one finger or the entire hand, thus making possible for the driver to control the interior temperature, audio and navigation systems. The driver can select the console screen status, go to the previous or next track, increase and decrease the volume, control the navigation coordinates and others functionalities.

For the gesture recognition functionalities, the system uses a TOF camera located almost at the middle of the console (in the space between the driver and the passenger). This system is different from the previous ones because it is more a showroom system rather than a showroom prototype.

- **Continental Steering Wheel with Gesture-Based Control**

Continental presented a concept of gesture-based control on the vehicle steering wheel, where the driver keeps the hands on the steering wheel and the eyes on the road, trying to "ensure more safety when driving" [74, 75].

This solution is a different approach from many other automotive gesture recognition systems that need more actions from the driver and specific areas of the console to detect the gestures.

The system uses a 3D TOF camera and four simple swipe gestures with the thumb to handle with five basic contexts: on-board navigation control, apps browsing, multimedia control, incoming calls control, and on-board computer controls.

The main advantage of this implementation is that the driver keeps both hands on the wheel and the eyes closer to the road, since the only effort besides the thumb gestures is to keep looking for the changes in the dashboard panel.

- **Ford C-Max HMI by Visteon**

The Ford C-Max HMI prototype shown by Ford and Visteon, at CES 2016, is a HMI concept with gesture recognition support. The gesture recognition system uses a TOF camera to recognize gestures (in reality only one finger pointing gestures). These gestures are used in different contexts like in multimedia control, menu navigation, lights control, glove box opening control, and HVAC control.

This system is distinct from the usual since it is inserted in a HMI system where the console screen do not have touch capabilities, but using the overhead TOF sensor, an estimation of the finger position is made, creating a the virtual touch on correspondent the screen area.

2.8 Mathematical Models for Gesture Recognition

2.8.1 Threshold-Based Model

A threshold model in any type of model, that relies on a threshold value or a set of threshold values. The threshold values are reference values used to distinguish ranges of values when the behavior predicted by a system varies in some important way with the time.

Briefly, if a system response varies with the time, the system could have its response controlled by a threshold or set of threshold values [76].

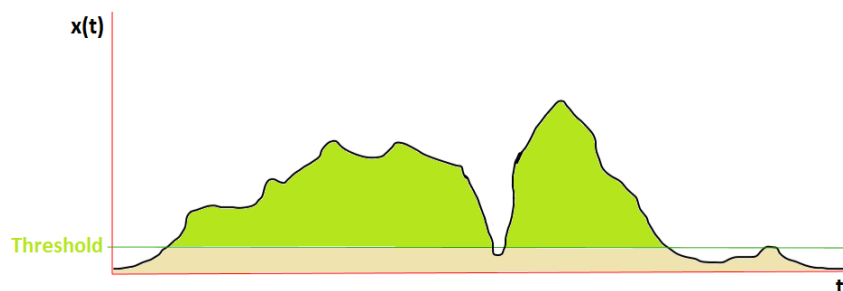


FIGURE 2.27: System response with threshold level.

This type of model is widely used in many applications and in many different areas, from software applications, computer vision, automation, animal training to even drug effect analysis. This model is typically used when the simplicity of the system is important or the response of the system is acceptable using the model.

The related work studied in this thesis identifies many cases of threshold based systems with different implementations but with good results[77–79]. As a model to use is a fairly interesting model because can give good results with low level of complexity.

2.8.2 DTW - Dynamic Time Warping Model

The Dynamic Time Warping (DTW) is a time series alignment algorithm, that measures the similarity between two temporal sequences of data which may vary or differ in time or speed. The algorithm is based on the alignment of two sequences of feature vectors and works by warping the time axis iteratively until an optimal match (according to predefined metrics or restrictions) between two sequences is found.

Thanks to the sequences of warping, it is possible to determine the similarity of the data independently of certain non-linear variations in the time dimension

DTW has been applied since its creation to many distinct applications, like in the analysis of temporal sequences of video, audio (especially speech recognition) and graphical data (image recognition).

DTW Application

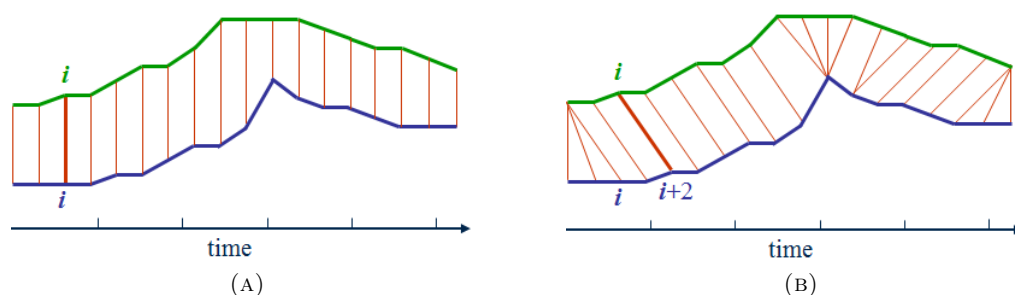


FIGURE 2.28: Analisis of data without(A) [80] or with DTW algorithm(B) [80].

Comparing the two figures, on the left in the figure 2.28a, it is possible to identify a poor similarity between both of the sequences of data, mainly because both sequences

were analyzed, point by point, on the same time basis. On the right, in the figure 2.28b, the similarity is higher, mainly because the alignment of the data was non-linear, allowing similar shapes to match without the same time basis (different phase in the time axis).

DTW Algorithm

Considering two sequences of feature vectors:

$$\mathbf{A} = \{a_1, a_2, \dots, a_n\}$$

$$\mathbf{B} = \{b_1, b_2, \dots, b_m\}$$

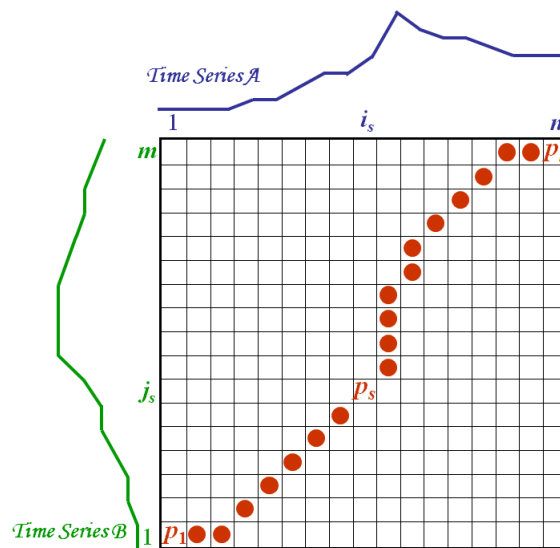


FIGURE 2.29: Representation of the two sequences of data [81]

The warping algorithm must ensure that the best alignment between A and B path is granted. In other words, the algorithm must ensure that the discrepancy between A and B is small.

$P = p_1, p_2, \dots, p_s, \dots, p_k$, where for example $p_s = (i_s, j_s)$ and P represents the warping function.

For the best alignment path between A and B,

$P_0 = \arg_{P \min} (\text{Distance}(A, B))$, where the Distance is calculated using:

$$D(A, B) = \left[\frac{\sum_{l=1}^k d(p_l) \cdot w_l}{\sum_{l=1}^k w_l} \right]$$

Because the number of possible warping paths is exponentially large, some restrictions are taken into account on the warping function, like the monotonicity, continuity, boundary conditions, warping window and slope constraint.

2.8.3 HMM - Hidden Markov Model

The Hidden Markov Model (HMM) is a stochastic model used to model sequences of data with finite number of states where it can be assumed that future states depend only on the current state and not on the previous states.

In HMM's, the next state is only dependent on the current state and on the fixed calculated probabilities to the transition between different states. In a general perspective each one of the states has a symbol, that can be observed, that is emitted with a specific probability. For this reason HMM's are a tool for representing probability distributions over sequences of observations where the states are hidden.

The general principles of the Hidden Markov Models(HMM) are very explored in the available literature, but there are a series of papers particularly relevant that were published by Leonard E. Baum and his colleges, that are one of the foundations of the work developed in HMM's. These papers were particularly important because they were one of the "start points" on the HMM topic.

Later, other papers were published by other authors like Lawrence R. Rabiner, Rakesh Dugad and U.B. Desai, providing a detailed analysis of HMM's, especially focused on a methodical review of the theoretical concepts of this type of statistical models.

With this work of gathering mathematical details and proofs, HMM's started gaining popularity, thanks to the rich mathematical support founded and the success of the applications of this model in areas like gesture recognition, speech recognition and human gene sequence structure modeling.

The rest of this section is used to describe a brief review of the theory behind HMM, mainly based on Lawrence R. Rabiner work, with some level of abstraction regarding the mathematical details of the models.

Discrete HMM - DHMM

Considering a system which may be described at any time as being in one state of N states, $S_1, S_2, S_3, \dots, S_N$, at a regularly spaced time, the system suffers a change of state, dependent on a set of probabilities for each state. Considering the discrete time instants $t = 1, 2, 3, \dots$, the current states of the system can be expressed as q_t .

Resuming the most important details of the model:

$$\lambda = \{A, B, \pi\}$$

T = Length of the observation sequence

N = Number of states used in the model

M = Number of observations symbols in the model

Q = Set of states of the Markov process = q_0, q_1, \dots, q_{N-1}

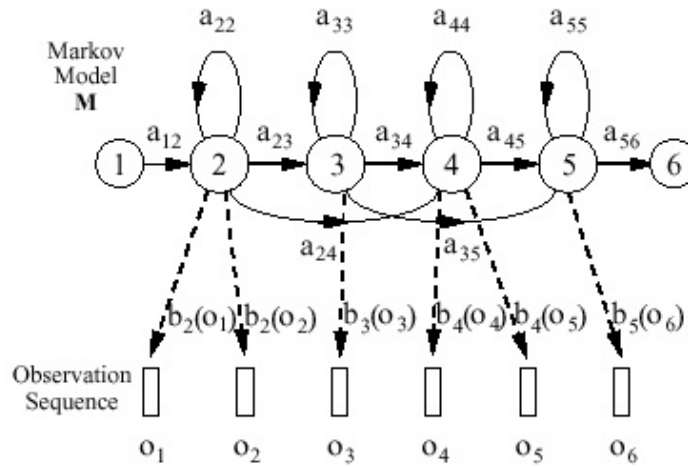
V = Set of possible observations of the Markov process = $0, 1, \dots, M - 1$

O = Observation sequence = $\{O_0, O_1, \dots, O_{T-1}\}$

A = State transition probabilities matrix = $\{a_{ij}\}$
 $= \{P(x_t = j | x_{t-1} = i)\}$ for $1 \leq i, j \leq N$

B = Observation probability matrix = $\{b_i(k)\}$
 $= \{P(o_t = k | x_t = i)\}$ for $1 \leq i \leq N$ & $1 \leq k \leq N$

π = Initial state distribution = $\{\pi_i\}$
 $= P\{x_1 = i\}$ for $1 \leq i \leq N$

Example HMM Model:FIGURE 2.30: HMM example model, with 6 states ($N = 6$) [82].

The Matrix A, B and π form is:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N,1} & a_{N,2} & \cdots & a_{NN} \end{bmatrix} \quad B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1N} \\ b_{21} & b_{22} & \cdots & b_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ b_{M,1} & b_{M,2} & \cdots & b_{MN} \end{bmatrix} \quad \pi = \begin{bmatrix} \pi_{11} \\ \pi_{12} \\ \vdots \\ \pi_{1,N} \end{bmatrix}$$

HMM's Topologies

There are some common topologies widely used in the HMM's applications. Topologies like:

- **Linear HMM:**

This case occurs if the elements $a_{ij} > 0$ where $j = i$ or $j = i + 1$. In this case, each state can rollback to itself or go to the immediately next state.

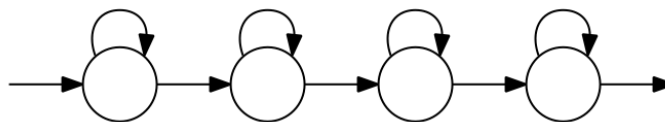


FIGURE 2.31: Linear HMM model [83].

- **Left-to-Right HMM:**

This case occurs if the elements $a_{ij} > 0$ where $j > i$. In this case, each state can rollback to itself or go to the next states.

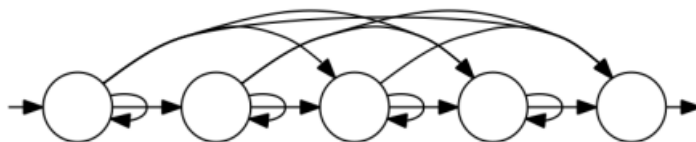


FIGURE 2.32: Left-to-Right HMM model [84].

- **Ergodic HMM:**

In this case, if $a_{ij} > 0$ any state can rollback to itself or can be reached from the other states.

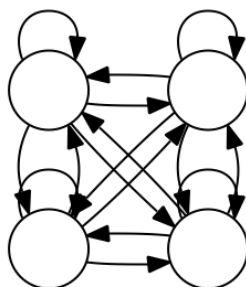


FIGURE 2.33: Ergodic HMM model [85].

The Problems of HMM's

- **The Evaluation Problem**

Considering the Hidden Markov Model λ , and the sequence of observations $O = \{O_0, O_1, \dots, O_{T-1}\}$, to compute the probability $P(O|\lambda)$, it is needed the information from $\lambda = \{A, B, \pi\}$.

Since HMM's have a finite set of states and observations, computing all the observations for every sequence of states to find the probability of the sequence that is needed is a exhaustive computational effort (N^T), where N and T are the number of states and possible observation sequences.

To solve this problem it is needed a algorithm, the Foward Algorithm, known for a lower complexity of N^2T , which is linear, considering N the number of states of the model.

- **The Decoding Problem**

Considering the Hidden Markov Model λ and the sequence of observations $O = \{O_0, O_1, \dots, O_{T-1}\}$, how to find the sequence $S_1, S_2, S_3, \dots, S_T$ that maximizes $P(S|O, \lambda)$?

The common solution is to use a dynamic algorithm for finding the most likely sequence of hidden states, the Viterbi algorithm. While very effective, the algorithm has important drawbacks. One is the non-real time response, because the algorithm is implemented recursively from the last observation in the sequence.

A possible solution is to apply the Viberti algorithm to a partial sequence of observations, making the algorithm independent of the last observation of the sequence.

• The Learning Problem

Considering the Hidden Markov Model λ and the sequence of observations $O = \{O_0, O_1, \dots, O_{T-1}\}$, how to find a Hidden Markov Model $\hat{\lambda}$, identical to λ , that maximizes the probability of observing O ?

In this case, to determine the optimum model where $P(O|\lambda)$ is maximal, there are 3 possible solutions, the Forward-Backward algorithm, the Baum-Welch algorithm, and Viterbi training.

2.9 Related Work

The areas of gesture recognition and HMIs are areas with a great and diverse field of applications and uses. The quantity and variety of examples and implementations, using gesture recognition applied to HMIs, to study and classify, is considerable.

The vast amount of technologies, devices and possible uses, make this area a source of great interest and work to develop, at least in the upcoming years.

This thesis and related research focus on the recognition of gestures, using IR technology, to incorporate in HMIs, more precisely automotive HMIs.

In this section a list of six examples was selected to illustrate the work being developed in the area of gesture recognition, using IR technology, applied to HMIs.

The Gesture Watch

The Gesture Watch [86], shown in Figure 2.34, is a prototype of a mobile wireless device, that is used on the user's wrist. This prototype allows hand gesture recognition to control some functionalities in different devices. The device uses an array of five proximity sensors to achieve the gesture recognition, interpreting a set of gestures to control functions in devices like smartphones, music players and others.

The device uses four main sensors on the top (2 for each reference axis) and one other sensor as trigger detection sensor.

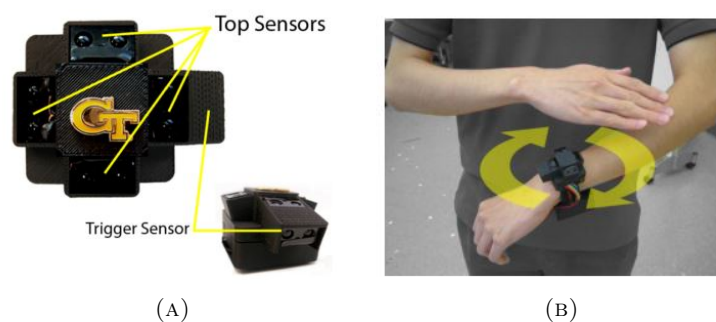


FIGURE 2.34: The Gesture Watch (A) and an example of gesture detected (Clockwise Circle) (B) [86]

GBECI - Gesture-Based Easy Computer Interaction using a Linear Array of Low Cost Distance Sensors

This case of study [87], presents the prototype of a gesture recognition device, as shown in Figure 2.35, that is used to support simple functionalities in HCI (Human-Computer Interaction).

The device uses eight proximity sensors, placed in a single line array, on a horizontal plane like a table or other flat surface.

The system is used to control a set of functionalities, based on the recognition of simple static and dynamic gestures, using the palm of the hand as main input of the system.

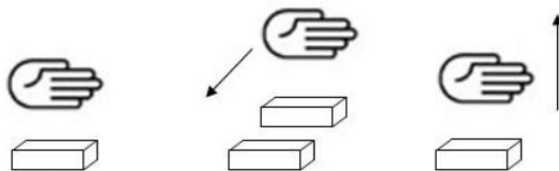


FIGURE 2.35: GBECI gesture detection technique [87]

HoverFlow

The case of study [88], presents the prototype of a gesture recognition device, as shown in Figure 2.36, that is used for ADI(Around Device Interaction), an emerging research topic in the field of the mobile device HMIs.

The device itself uses six proximity sensors, placed in the device edges and in the front face of the device.

The system is used to control a set of functionalities in a remote device using bluetooth connectivity. These functionalities are achieved through the recognition of static and dynamic gestures, using the palm of the hand as main input of the system.

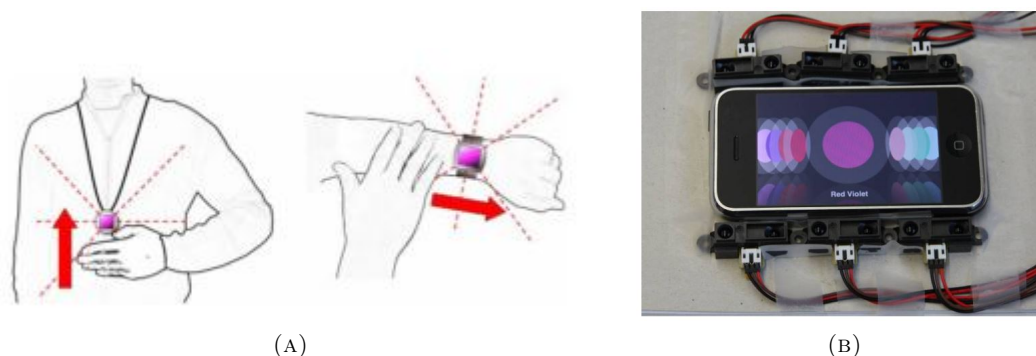


FIGURE 2.36: Hoverflow Concept (A) and Hoverflow Prototype(B)

Contactless Gesture Recognition System Using Proximity Sensors

The case of study [89], presents the prototype of a gesture recognition device, as shown in Figure 2.37, that is used for HMI (Human-Machine Interaction).

The device uses one proximity sensor(receptor) and two IR LEDs (emitters), placed in a single line array. This prototype uses the two emitting LEDs in a phased activity since only one of the LEDs is ON at a specific time. This allows the use of a single receptor for the detection of gestures.

This system is used to control a set of functionalities, based on the recognition of dynamic gestures, using the human hand as main input of the system. Since the simplicity of the system is the key, only 4 gestures are detected, being them, the left-right, right-left, push and pull gestures.

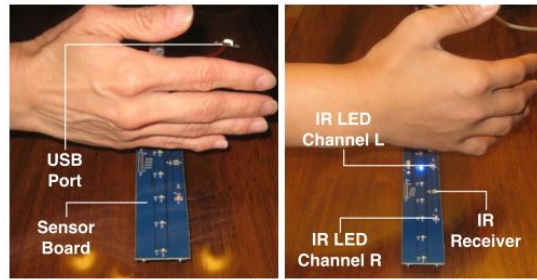


FIGURE 2.37: Contactless Gesture Recognition System prototype [88]

Contactless Hand Gesture Recognition System

The case of study [90], presents the prototype of a gesture recognition device, as shown in Figure 2.38, that is used for HCI (Human-Computer Interaction). The device uses a bi-dimensional array of proximity sensors, with a dimension of three by three.

This system is used to control a set of eight functionalities, emulating the inputs of a mouse and keyboard in a regular computer. The system recognizes a set of dynamic gestures, using the human hand as main input of the system.



FIGURE 2.38: Contactless Hand Gesture Recognition System [90]

A motion gesture sensor using photodiodes with limited field-of-view

The thesis [79], presents the prototype of a low-power and small-size motion gesture sensor (MGS), as shown in Figure 2.39, that is used to support HMI. The device uses just one IR LED and two photodiodes, instead of two IR LEDs and two photodiode, to achieve low-power consumption. The system also uses optical blocks to limit the field-of-view of each photodiode

The system recognizes just a few gestures (swiping gestures), using the human hand/finger as main input of the system.

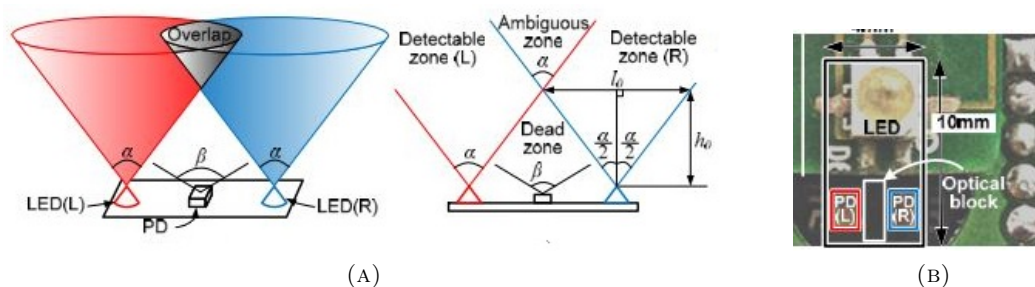


FIGURE 2.39: Motion gesture sensor scheme (A) and prototype(B)

Final Comments

In this chapter were presented important knowledge's mainly related to gesture recognition, HMIs, usability and gesture recognition technologies. The main focus was on the gesture recognition technologies since they were an important component of the critical knowledge needed for further research. Because of this reason, a brief study on capacitive, ultrasound, radar, infrared, laser, imaging and wearable technologies was conducted.

Another important study conducted was the study of the market (by analyzing the available forecasts, case studies and concepts).

Also, in the chapter a brief study covering three algorithm approaches was conducted, namely by studying Threshold-based, DTW and HMM approaches. Finally a brief study on the available related work was conducted.

Chapter 3

System Specification

The previous chapter described some of the most important theoretical concepts and knowledge's related to this thesis, by exploring key concepts like gesture classification, automotive HMI design, gesture detection technologies, usability concepts, automotive HMI market status and important related work.

This chapter presents the research methodology adopted on this dissertation, followed by an overview of the prototyping gesture recognition system as well as software and hardware requirements specification.

Furthermore, the architecture of the system is also presented as well as a brief conceptual design section with the exploration of a set of gestures and HMI concepts.

Finally the data acquisition and validation processes are briefly described and the data processing options are taken into consideration.

3.1 Methodology Specification

In terms of research methodology, this thesis is structured and developed following the Waterfall model. This approach or model is a sequential design process, widely used in systems development processes, being mainly non-iterative, flowing firmly downwards, like a real waterfall.

Following this framework the project is divided in five main phases or steps: Analysis (Requirement Analysis), Design, Implementation, Verification and Maintenance [91–94].

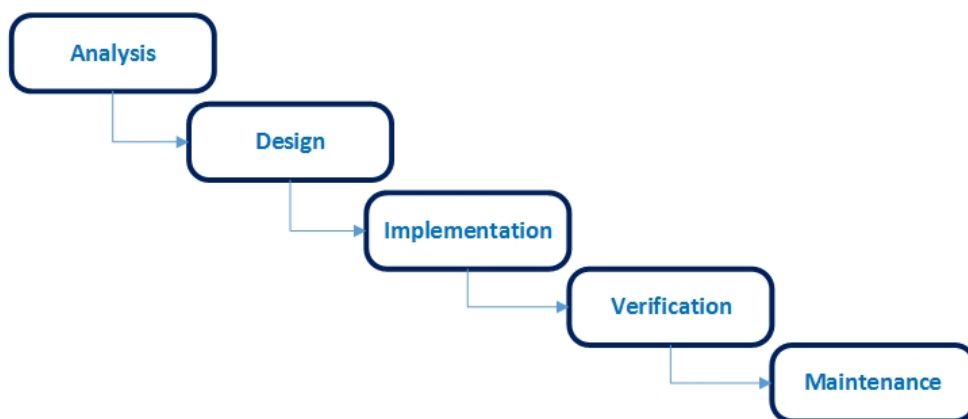


FIGURE 3.1: Waterfall Model Development Steps.

The first phase, Analysis phase, is an important phase where the objectives are to understand the problems, needs and goals to achieve. It is a especially important phase because it is here that the user requirements are defined and understood. This phase is the beginning of the project definition and planning.

Then, the second phase, Design Phase, is important to create the plans, drawings and conceptual algorithms that will describe the final system that will be implemented. It is the phase where the system is designed based on the selected requirements previously analyzed.

In the third phase, Implementation Phase, the implementation process must be defined and executed. If some hardware manufacturing is required the manufacturing should be achieved. The software solutions should be implemented and the hardware components should be also integrated with the software.

In the fourth phase, Verification Phase, the main focus is to define and implement the testing procedures focusing on the validation of the operation process. In this phase, the system is carefully analyzed to check if the system fulfills the customer expectations.

In the fifth phase, Maintenance phase, the system is analyzed to find problems, either due to improper requirements determination, changes in requirements or errors in the design process. This phase is characterized by modifications applied to the system done to solve problems or errors detected.

This framework, was chosen because of advantages like the detection of errors in the project design phase (saving time in the implementation phase), the technical documentation support given to new developers making the maintenance phase easier, the clear definition of milestones to achieve, the more clear estimation of the project costs, and the easier definition of testing scenarios by using the functional specifications and requirements.

3.2 Development Requirements

For the development of the system architecture, several requirements were taken in account. The requirements were divided in two categories, the functional and non-functional requirements. The functional requirements selected for this project were:

- collect gesture data from the gesture recognition device(s);
- process the collected data using gesture recognition algorithms;
- communicate a response for each gesture detected to a central stack/infotainment system using an Ethernet connection (TCP/IP protocol).

′ In terms of non-functional requirements, the requirements were:

- be able to capture valid gesture data, in gestures with different hand positions, speeds, distances and execution times;
- ensure low levels of calibration for the gesture recognition algorithms;
- locally process the gesture data, and communicate the system responses to a central stack/system in a period of time of less than 1 second.
- ensure that, the rate of gesture detection can be equal or above 70%;
- ensure that, for each type of gesture, the rate of recognition can be equal or above 70%;
- create a prototype capable to be integrated and tested in a "real world" driving simulator;
- ensure that a basic set of gestures can be recognized by all the different gesture recognition devices, creating that way a gesture recognition platform independent of the gesture detection device.

3.3 System Overview

Based on the purpose of this thesis, namely the recognition of gestures using IR sensing technology to apply in the design of automotive HMI concepts, a simple overview of the system is depicted in Figure 3.2 using an intuitive block diagram.

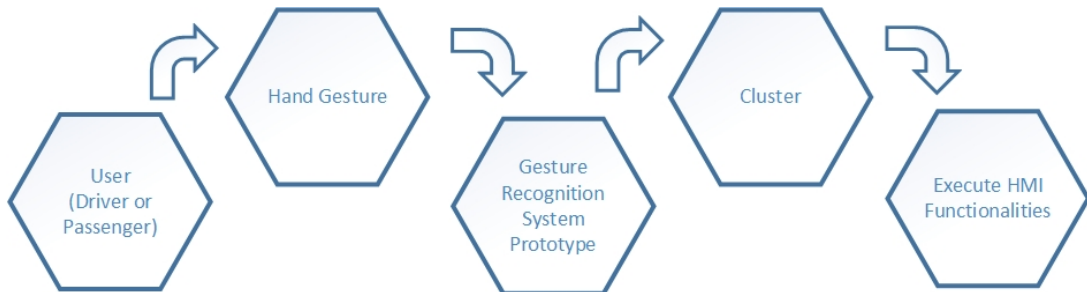


FIGURE 3.2: Global System perspective.

The previous diagram, Figure 3.2, shows the complete process of gesture recognition applied to a custom gesture based HMI. It is possible to see the interactions between user (mainly hand gestures), gesture recognition system, cluster and HMI functionalities.

3.3.1 Detailed Overview

Based on the Figure 3.2 and focusing on the components of a conceptual HMI system, a more complete perspective of this system can be achieved with the diagram in the Figure 3.3.

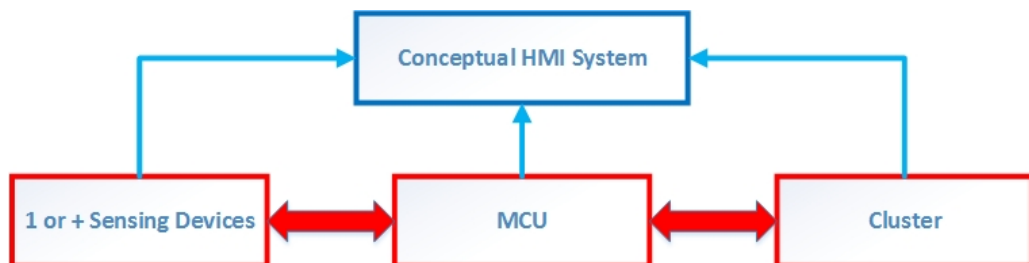


FIGURE 3.3: Simple diagram of the system.

Analyzing the diagram, it is possible to see a gesture-based HMI concept (blue box) and the main components of that HMI concept (red boxes). In terms of components, they are the sensing devices (used for gesture detection), microcontroller unit (MCU) and Cluster. As implementation steps, three steps can be identified watching the components in the diagram being them, the sensing device interface, the gesture detection algorithms and the data communication to the cluster.

3.3.2 Gesture Recognition System Overview

In a more complete diagram, Figure 3.4, it is possible to see a more complete perspective of the conceptual HMI system. A clear description of the gesture recognition system components and functionalities can be also presented.

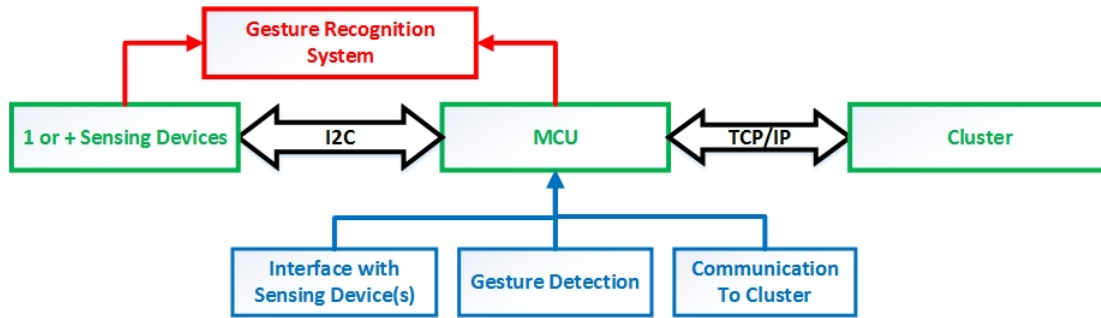


FIGURE 3.4: Complete diagram of the system.

Observing Figure 3.4, the main blocks that represent the functionalities of the system are the sensing device interface, gesture detection and the communication to the remote cluster (blue color boxes).

Also, in this diagram the components of the system are represented, namely the sensing devices, the MCU and the cluster (green boxes). The system is not integrated with the Bosch DSM cluster, but it will be compatible in terms of functionality with that cluster. The cluster was simulated using a workstation running a dedicated server used to treat the messages received from the MCU.

In the diagram, it is also possible to see that the communications protocols will be the TCP/IP, for the interface of the MCU to the Cluster and I2C for the interface between the MCU and the sensing devices. Other communication protocol also used is the serial protocol, to communicate the system response to an auxiliary pc. In this step, the main requirement was the use of TCP/IP protocol since it is the interface used in the Bosch DSM and other clusters and workstations.

3.4 Technology Specification

After the study of important background knowledge, a set of procedures must be taken in order to design and implement the gesture recognition system that is a part of the objectives for this dissertation.

In this procedures of study and selection, it is important to define rules or metrics to define important elements of the project. The next topic will cover the selection of the technology used in this thesis.

3.4.1 Technology Selection Criteria

In this dissertation, it is important to select a technology capable to be used in the detection and recognition of gestures. The technologies already in the market are many and very distinct (section 2.4), and newer and more effective technologies and solutions are being introduced yearly. For the selection of the technology, some metrics or criteria must be defined and applied.

List of criteria to select the technology:

- Availability on the market;
- Detection Range;
- Functionality;

- Time-to-prototype;
- Efficiency;
- Available Support and Knowledge;
- Implementation Size;
- Implementation Costs;

3.4.2 Definition of the technology.

For the selection of the technology, besides the previous definition of the criteria, after some contacts with the project partners, market analysis and project strategy definition, a technology was selected, the Infrared technology. Other technologies were taken in account, especially capacitive and ToF camera technologies, being the first one also explored in the project where this thesis is a part of.

3.4.2.1 Advantages and Disadvantages of IR Technology

IR technology applied to gesture based HMIs, in particular automotive HMIs, can be a good option for simple concepts of interaction.

Since the main options taken in care were IR, Capacitive and ToF camera technologies a small comparison between them is made below.

Specification	IR	Capacitive	ToF Cameras
Cost	Low	Low	Average-High
Size	Small	Small-High	Small-High
Availability	Good	Average	Good
Time-to-prototype	Low	Low	Low-Average
Functionality	1D to 3D detection	1D to 3D detection	3D detection
Detection Range	Good	Average	Good
Performance	Average	Average	Good
Support & Knowledge	Good	Good	Good

TABLE 3.1: Comparison of IR, Capacitive and ToF Camera Technologies

Looking at the table 3.1, it is necessary to note the advantages of IR technology in terms of implementation costs, size, availability, time-to-prototype, functionality, detection range, support and available knowledge.

In terms of disadvantages, the most important are the more limited performance and vulnerability to ambient interferences.

3.5 Hardware Specification

For the development of this dissertation, different hardware elements must be selected. The definition of rules or metrics to define important hardware elements is crucial and it will be covered in detail in the next topics.

3.5.1 IR Sensing Devices

To solve the first research question, "What devices, based on IR technology, should be used for the development of automotive HMI concepts?", a set of steps are needed, being them:

- The market study, in terms of IR devices/sensors for gesture recognition;
- The definition of the criteria to select the devices or sensors;
- The selection of the sensors or devices, based on the chosen criteria;

3.5.1.1 Market Study

For this step, a major criteria was defined, for the study of the sensing devices and their use cases. The criteria was that the technology to use must be based of IR technology. The type of sensors to use should be based on active IR technology, including ToF sensors. So, devices like passive IR sensors or TOF cameras for example, are strategically out of selection.

During the market study another important question was the grade of the sensors. Here the option was to use the available sensors on the market, not only automotive grade sensors, since the final system will be a functional development prototype to be inserted and tested in a driver simulator mockup.

The prototype system should consist in a framework to build a showroom of automotive HMI concepts, so the need of sensors already validated for automotive use is not essential, but it is desired.

3.5.1.2 Criteria to select the Sensing Devices

For this step, a brief list of criteria was created to evaluate and select the sensors. This list is composed by the following criteria:

- Detection Range;
- Functionality (especially axis/dimensions of detection);
- Availability in the market;
- Time-to-prototype;
- Protocol of communication;
- Resolution;
- Linearity;
- Cost;

Following this list, some criteria must be taken in special consideration, criteria like: cost, detection range, functionality, type of communication and availability in the market.

3.5.2 Selection of Sensing Devices

After a study of the market and the definition of the criteria to evaluate and select the sensors, a list of 4 sensors was selected for further study and development. That list is composed by:

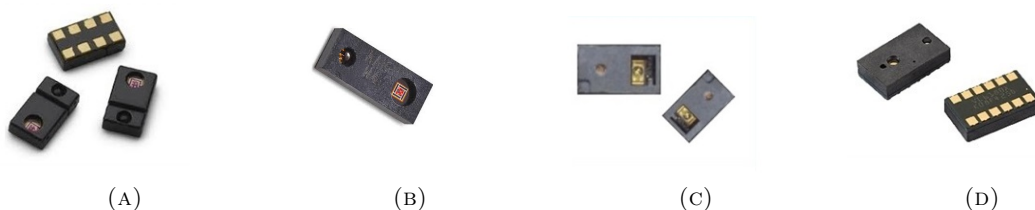


FIGURE 3.5: A - Avago APDS-9960, B - ams TMG4903, C - PixArt PAJ7620U2, D - ST VL6180X

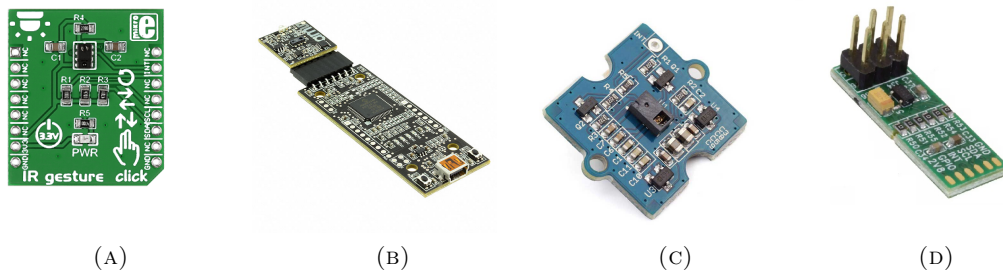
The first sensor is the Avago APDS-9960, Figure 3.5a. This sensor was selected mainly because it allows the detection of 2D gestures as well as proximity detection (allowing for this reason at least 3 types of gestures: horizontal and vertical swipe gestures and in/out gestures (proximity based measurements)).

The second sensor is the ams TMG4903, Figure 3.5b. This sensor was selected because allows 2D-3D gesture detection, as well as only proximity detection. The main differences to Avago APDS-9960 are the resolution, granting higher quality of raw data (16 bits compared to 8 bits in the APDS-9960 sensor), and the higher level of calibration options available.

The third sensor is the PixArt PAJ620U2, Figure 3.5c. This sensor was selected as a natural complement of the Avago APDS-9960 and ams TMG4903. The main source of interest in this sensor is the onboard detection of 9 gestures. Another aspect to take in consideration is that this sensor uses a matrix of 60x60 pixels instead of just four photodiodes.

The fourth sensor is the ST Microelectronics VL6180X, Figure 3.5d. This sensor was selected because it allows proximity detection based on the TOF principle, being a source of interest for the global project. Besides this fact, it is also interesting because can achieve better results and lower interferences than other active IR sensors. With more than one of this sensors it is possible to detect 2D gestures like in the Avago APDS-9960, granting the detection of the same number of gestures.

Because the sensors are already available in the market, on development boards a selection of development boards was also made. The list of development boards is composed by:



(A) (B) (C) (D)
 FIGURE 3.6: A - MikroElektronika IR Gesture click board, B - ams TMG4903 Eval Kit, C - Seeedstudio Groove - Gesture v1.0, D - ST VL6180X SATEL board

- MikroElektronika IR Gesture Click board, 3.6a, with the Avago APDS-9960.
- Evaluation Kit for TMG4903, 3.6b, based on the ams TMG4903 sensor.
- Seeedstudio Groove - Gesture v1.0 board, 3.6c, based on the PixArt PAJ7620U2.
- ST VL6180X SATEL board, 3.6d, based on the ST VL6180X sensor.

3.5.3 Criteria to select the Development Board

Besides the sensors or devices used for gesture recognition, it is important to define the criteria to select the other hardware components to use in the project.

A special item is the hardware development board used to interface with both the cluster and the sensors, and to run the gesture recognition algorithms.

The list of criteria defined for this step is:

- Performance;
- Size;
- Availability;
- Functionality;
- Communication support(Ethernet, I2C, SPI, UART & CAN);
- Mbed development platform support;
- Cost;

In the selection of the hardware and following this list of criteria, it is important to highlight criteria like the performance, availability, functionality, and communication protocols support.

3.5.4 Selection of the Development Board

Respecting the list of criteria previously presented, the development board selected was the EA LPC4088 QuickStart Board, from Embedded Artists.

This board, based on the NXP LPC4088 microcontroller, was selected because of the performance, size, functionality, communication support(native Ethernet, SPI, I2C, CAN and UART interfaces) and especially Mbed development platform support.

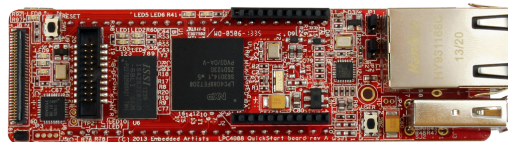


FIGURE 3.7: EA LPC4088 QuickStart Board [95]

Besides this selection, another development board was also used because it was immediately available for development. This board was the NXP LPC1768, which also respects the selection criteria as well the EA LPC4088 board.

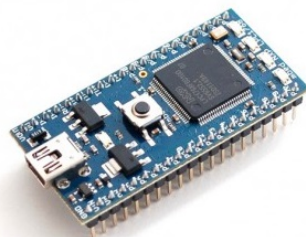


FIGURE 3.8: NXP LPC1768 Board [96]

3.6 Software Specification

After the selection of the sensors, the needs to interface with the sensors and to be able to recognize gestures are immediately raised.

This needs are directly connected to the second research question, "What gestures, based on the selection of IR devices, can be recognized and can be used for the development of automotive HMI concepts?".

To interface with the sensors, to develop the gesture detection algorithms, and to create demo applications, a few steps are needed like the selection of the programming

language(s), the selection of software development platforms, and the selection of the algorithms.

3.6.1 Programming Languages

For the development of this thesis, one or more programming languages must be selected. For this selection, a list of criteria should be defined and respected.

The list of criteria chosen for the selection of the programming languages(s) is:

- Target platform(s);
- Flexibility of the language;
- Performance;
- Official support;
- Developer Community;
- Portability;

As programming languages for this thesis two programming languages were selected, C and C++, mainly because they are supported by the target embedded systems/platforms, have good support to developers, flexibility and large community.

3.6.2 Software Development Platforms

For this type of project, with high dependance on both hardware and software, the software selection is a very important step to take in care. For this step, a set of criteria must be defined.

The list of criteria defined for this step is:

- Cost;
- Official Support;
- Developer community;
- Functionality;
- Time-to-prototype;
- Code portability;
- Learning Curve;

After a brief period of market analysis and based on the list of criteria a set of software tools was selected. The selected software is:

- Keil IDE + MDK-ARM Version 5 Microcontroller Development Kit

This software was selected to develop C/C++ code to create the interface between the sensing devices, development board and cluster, as well as to implement the gesture recognition algorithms.

- Mbed Online IDE + Mbed SDK

This software was selected with the same purposes of Keil. The main difference between both software is that Mbed promotes higher level software implementations, easy code portability between different development boards and platforms, and lower

time-to-prototype. A huge handicap is that Mbed is cloud-based, and with that offline work is impossible.

- Matlab R2015b + Kevin Murphy HMM toolbox

Matlab R2015b software was selected because it was necessary to develop and assess gesture recognition algorithms. The Kevin Murphy HMM toolbox for Matlab was used mainly to test HMM model implementations, mainly for the creation and learning phases of the models.

- Qt Creator 3.5.0 based on Qt 5.5.0

This software was selected for the creation of GUI applications, to test the sensors interface and to collect gesture data for analysis. Another aspect was that this software was a strategic option to use in the project where this thesis is a part of.

3.6.3 Algorithm Development

The gesture recognition algorithms are crucial to develop a gesture recognition system, and in this area a great number of algorithms and implementations is available, so the need to define criteria to proceed with the algorithm selection is crucial.

3.6.3.1 Criteria to select the Algorithms

To select the algorithms to use in the dissertation, a list of criteria can be composed by:

- Complexity;
- Implementation time;
- Performance;
- Reliability;
- Robustness;
- Adaptability;
- Available support;

3.6.3.2 Selection of the Algorithms

For this thesis, based on the selection criteria, two approaches were selected for further study and algorithm development. The two approaches selected are respectively:

- Threshold Approach

The first option to consider was based on a threshold-based approach. This is a simple approach, but with good support for the development of algorithms to recognize simple gestures. Typically simple dynamic gestures can be detected with sufficient detail using this approach. This approach has the potential to be fast to develop, simple and performance effective.

- HMM Approach

Another option, based on machine learning, is the HMM approach. HMM's are a good option when talking of dynamic gestures, being one of the most widely used options in the gesture recognition area. HMM's are typically used because they allow the modeling of a system with time varying series of data, allowing for this reason the

detection of gestures with different shapes and durations.

Notes:

For the selection and development of algorithms three main factors/criteria were taken in account, being them:

- Reliability - the system should be accurate, avoiding false triggering and false gesture classification;
- Robustness - the system should minimize the effects of the noise and other interferences;
- Adaptability - the system should minimize the effects of different gesture postures, speeds and timings;

3.7 System Architecture

To implement a system it is necessary to first project or design the concept of the desired system.

More than just choosing the components and the tools, it is also necessary to take in consideration the system requirements to design a system architecture.

Following a modular approach, the following topics will cover all the design choices made on the system architecture design.

3.7.1 Local System

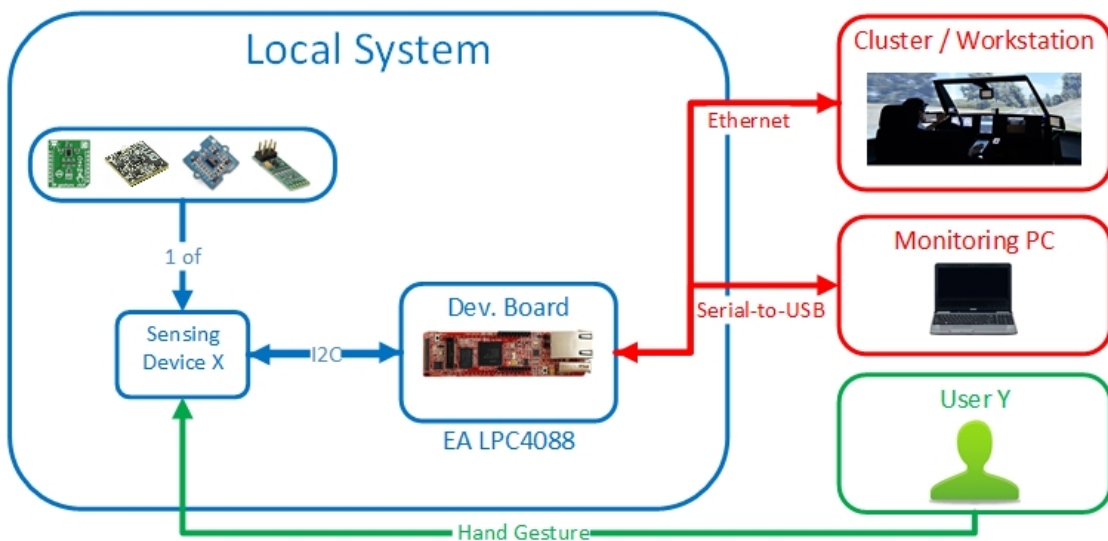


FIGURE 3.9: Local System diagram.

The local system is essentially composed by two main elements: the development board and the gesture recognition devices (Figure 3.9).

It is also possible to identify the other components connected to the system, namely the Cluster/Workstation, used to run a dedicated server to treat the gesture data from the MCU, the local PC for monitoring purposes and the user interaction.

Since the different sensing devices have different specifications and characteristics, the interface to use with them varies. To allow a better understanding of the interfaces for each sensors, this section will be divided in 3, one for each sensor in use.

3.7.2 Avago APDS-9960 and ams TMG4903

- **Avago APDS-9960**

The Avago APDS-9960 device, from Avago Technologies, is a sensing solution that allows gesture detection, proximity detection, ambient light detection, and color sensing.

The Gesture sensing block uses four directional photodiodes to sense the reflected IR energy (sourced by the integrated IR LED) to convert physical motion information (direction, distance or speed) to digital raw data.

The architecture of the sensor allows high sample rate, ambient light subtraction, 8 bit proximity/gesture raw data output and 16 bit ambient light/RGB color output.

- **ams TMG4903**

The ams TMG4903 device, from ams AG, is a sensing solution that allows gesture, proximity and ambient light detection. Furthermore the device allows RGB color sensing, IRBeam optical pattern generation and IR device control.

The Gesture sensor uses four directional photodiodes (North, South, West and East photodiodes) to sense the reflected IR energy (sourced by the integrated IR LED), converting physical motion information (direction, distance or speed) to digital raw data.

The architecture of the sensor allows high sample rate, self-maximizing dynamic range, ambient light subtraction, 16 bit proximity/gesture raw data output and 16 bit ambient light/RGB color output.

3.7.2.1 Interface Strategy

The interface between the sensing devices and the microcontroller is a crucial necessity. In this case, and considering the similarity between the sensors, in terms of output data and structure, a common strategy can be defined for non-specific elements, like the configuring procedures, calibration and even raw data capture.

This strategy is mainly useful for the analysis of the raw data from the sensors, because even with differences in the data output (8-bit and 16-bit), the sensor structure allows a identical interface.

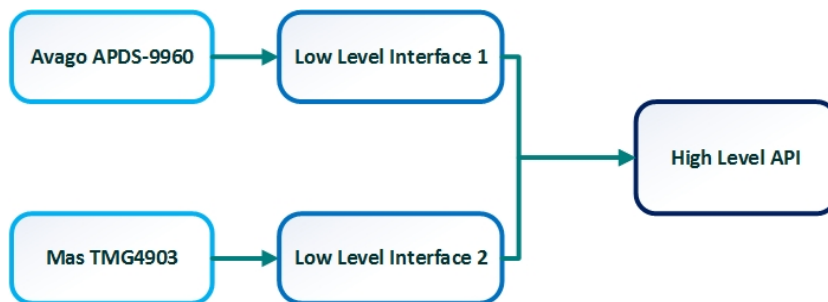


FIGURE 3.10: Device interfaces scheme.

Observing the figure above, Figure 3.10, it is possible to see a specific low level interface to each sensing device, but a common API shared by both sensing devices. This fact is mainly due to the similarities in the devices themselves.

For a better comprehension, the next topics will cover important components of the interface design by dividing them in working blocks.

Main Interface Block

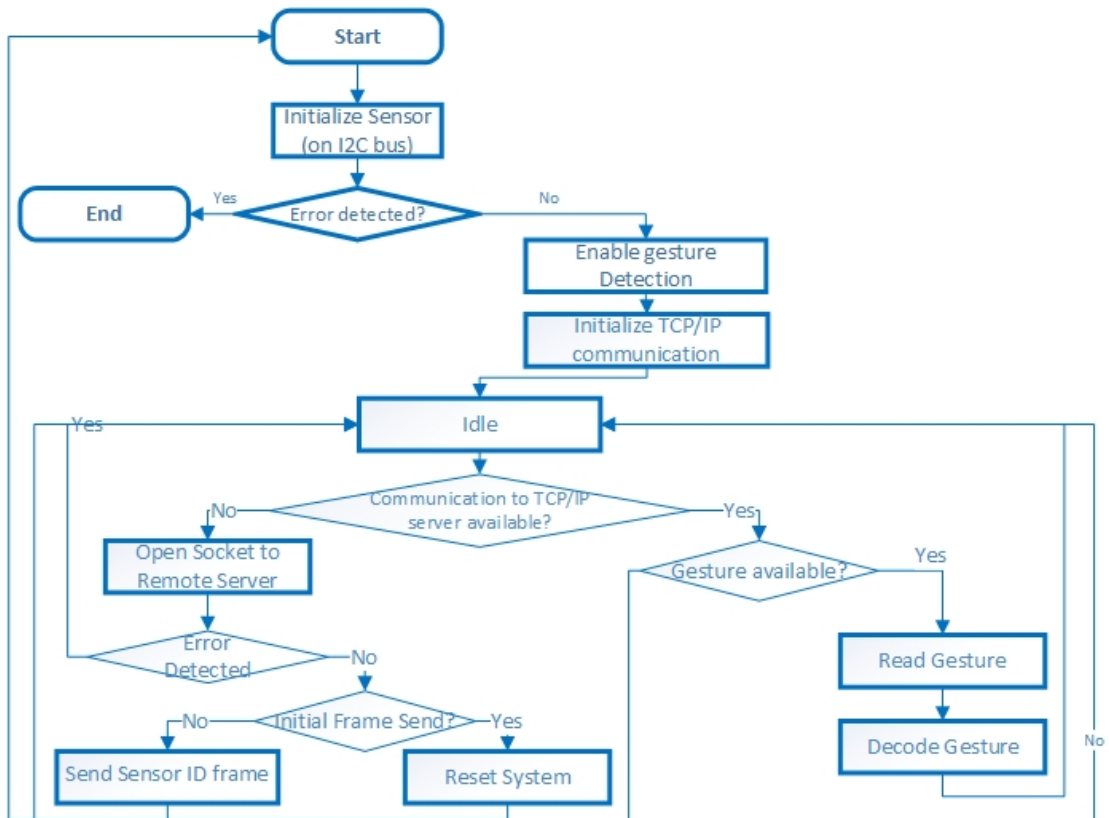


FIGURE 3.11: Main device interface flowchart.

Considering the Figure 3.11, it is possible to see a fairly simple block, where the sensors and network initialization is done. Furthermore, it is possible to see that the gesture detection and classification is done in two blocks, where the "Read Gesture" block is responsible for the data acquisition and the "Decode Gesture" is responsible for the gesture classification. Another aspect to consider is the process of communication to the server, explained ahead in detail in the section 4.1.2.

Detection Block



FIGURE 3.12: Gesture detection block.

Both sensors have status registers where specific gesture flags are raised upon the activation of the gesture detection state machine, so a basic procedure is to read those flags or as alternative, wait for an interrupt to occur upon the first dataset of collected data from a gesture.

The Figure 3.12 shows this procedure, and by analyzing the blocks, it is possible to see that the status registers are read and the gesture flags extracted to check if a gesture is available.

Read Gesture Block

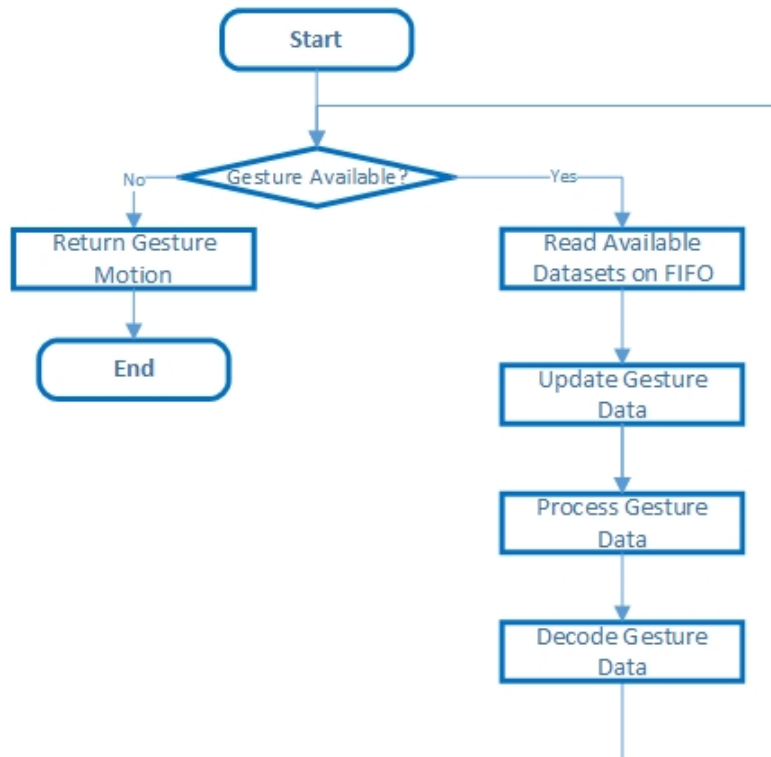


FIGURE 3.13: Gesture data read block.

In the Figure 3.13, it is possible to recognize how the process of data acquisition is developed.

Basically both sensors have 32 datasets FIFO's (128 bytes or 256 bytes considering Avago APDS-9960 sensor or ams TMG4903 sensor) where the gesture data is stored.

The process of data acquisition passes by reading that temporary FIFO's ("Read Available Datasets on FIFO" block) to auxiliary memory, organizing the data by each photodiode ("Update Gesture Data" block).

Then, by analyzing and treat the data from the Up/North, Down/South, Left/West and Right/East photodiodes ("Process Gesture Data" block) it is possible to determine the direction or nature of the movement ("Decode Gesture Data" block).

After this step the gesture data is processed to identify the gesture previously performed by a user.

Process Gesture Data Block

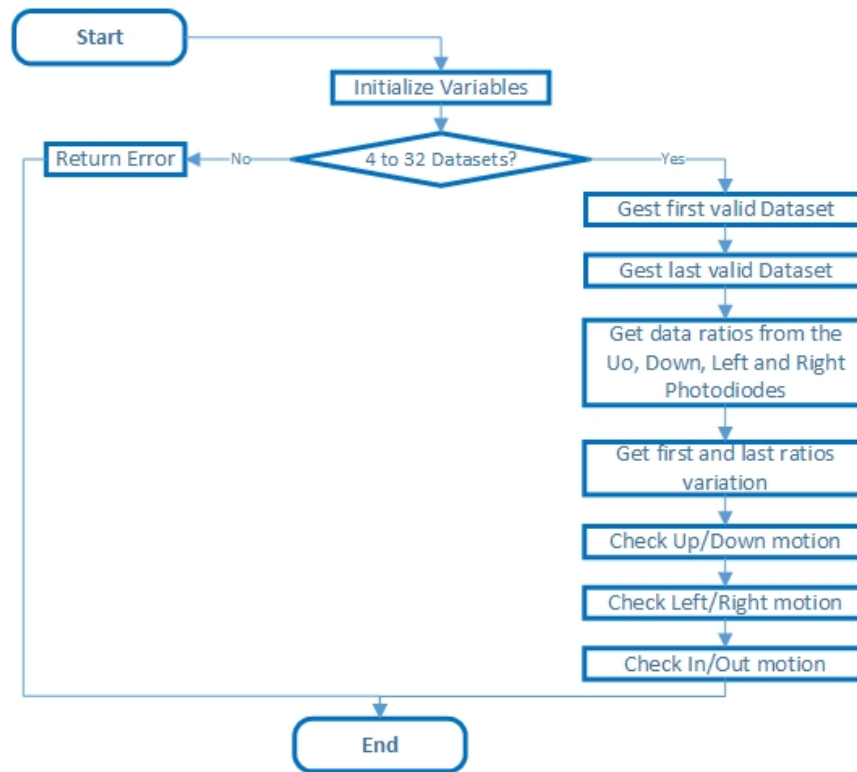


FIGURE 3.14: Process Gesture Data block.

Figure 3.14 shows the "Process Gesture Data" block, where the data collected from the gesture is analyzed. The first step to process the data is to collect the first and last gesture datasets that are valid (above thresholds). The Figure 3.15 shows how the data collected is treated.

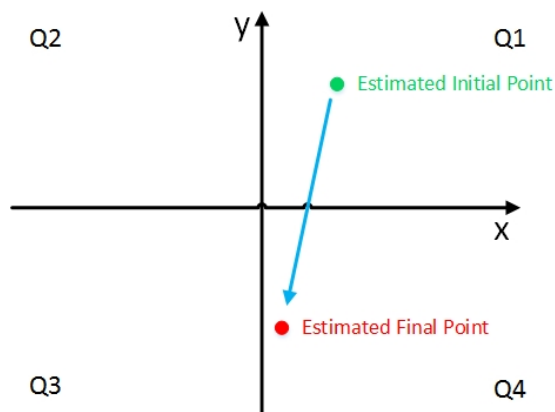


FIGURE 3.15: Dataset collection example.

To estimate the gesture nature, an estimation of the initial and final positions is developed. The first and last position ratios, in the x-axis and y-axis, are calculated using the first and last datasets, more precisely using up, down, left and right first and final readings from the photodiodes.

After this step, the ratios variation is analysed and stored, to be used in the next

phase to check the type of motion under detection, namely Up, Down, Left, Right, In and Out movements.

Decode Gesture Data Block

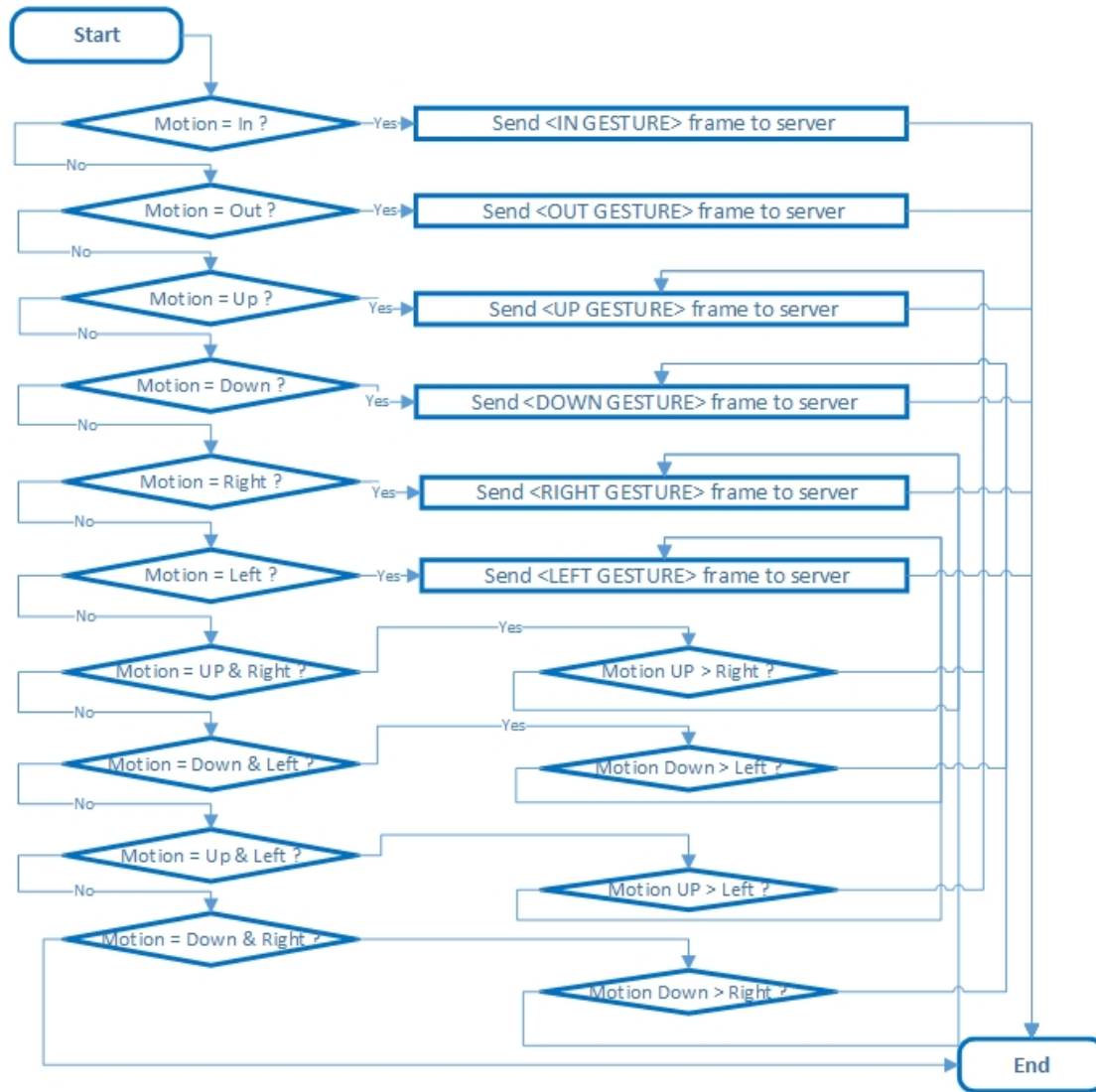


FIGURE 3.16: Decode Gesture Data block.

The figure above, Figure 3.16, shows the "Decode Gesture Data" block. In this block, the motion data previously collected and treated, is used to determine the type of movement detected.

Also in this block the communication to the server is achieved, following a communication protocol explained in more detail in the section 3.7.5.

3.7.3 PixArt PAJ7620U2

The PixArt PAJ7620U2 device, from PixArt Imaging Technologies, is a sensing solution that allows gesture detection, proximity detection, object brightness and size detection and image detection.

The device uses a 60x60 pixels sensing array, generating a image capable of being accessed via SPI interface. Besides this image capture mode, the sensor has a gesture detection/cursor mode using I₂C interface.

In the gesture detection mode, there are 9 native gestures capable of being detected by the sensor (move up, down, right, left, forward, backward, circle-clockwise, circle counter-clockwise, and wave).

The practical range of detection goes from 5 to 15cm, the operating viewing angle (FOV) is about 60°, and the detection speed can go from 60 °/s to 600 °/s or from 60 °/s to 1200 °/s (normal or gaming mode).

3.7.3.1 Interface Strategy

The interface between the sensing device and the microcontroller is a necessity to take in consideration. This interface will be important to configure the device and extract the gesture data from the device itself. This device has a more simplistic interface since the raw data from the gestures is processed internally, being the most important step the identification of the internal flags raised upon the detection of each gesture.

Main Interface Block

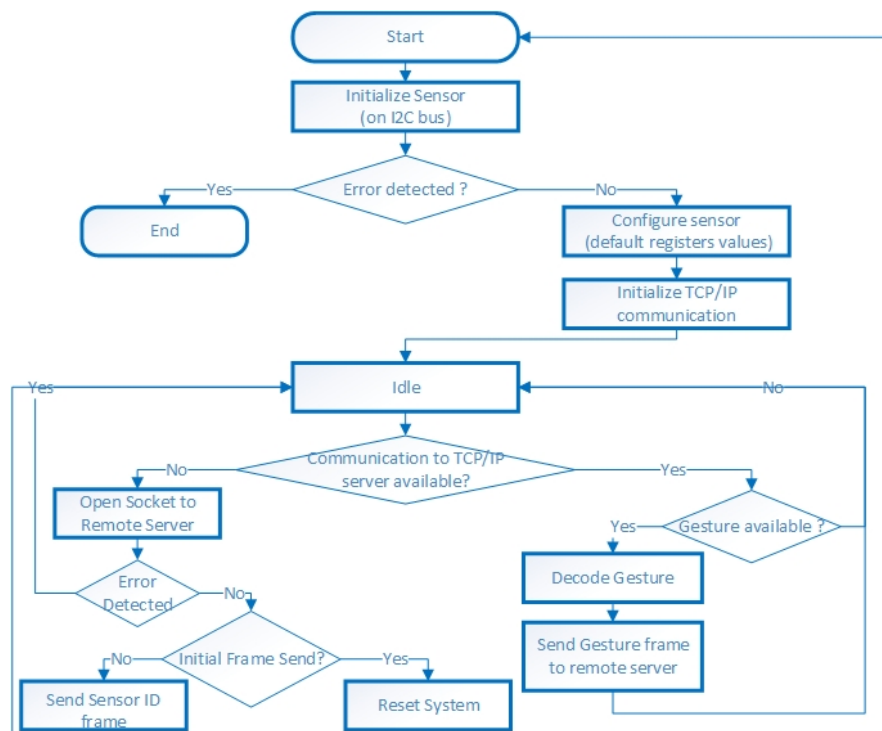


FIGURE 3.17: Main device interface flowchart.

This block is the main block of the interface code for the PixArt PAJ7620U2 device. Like it is possible to see, in Figure 3.17, the main steps are the device initialization,

network initialization, gesture detection and communication to server. This block also illustrates the method used to maintain the system communicating to the server.

Decode Gesture Block

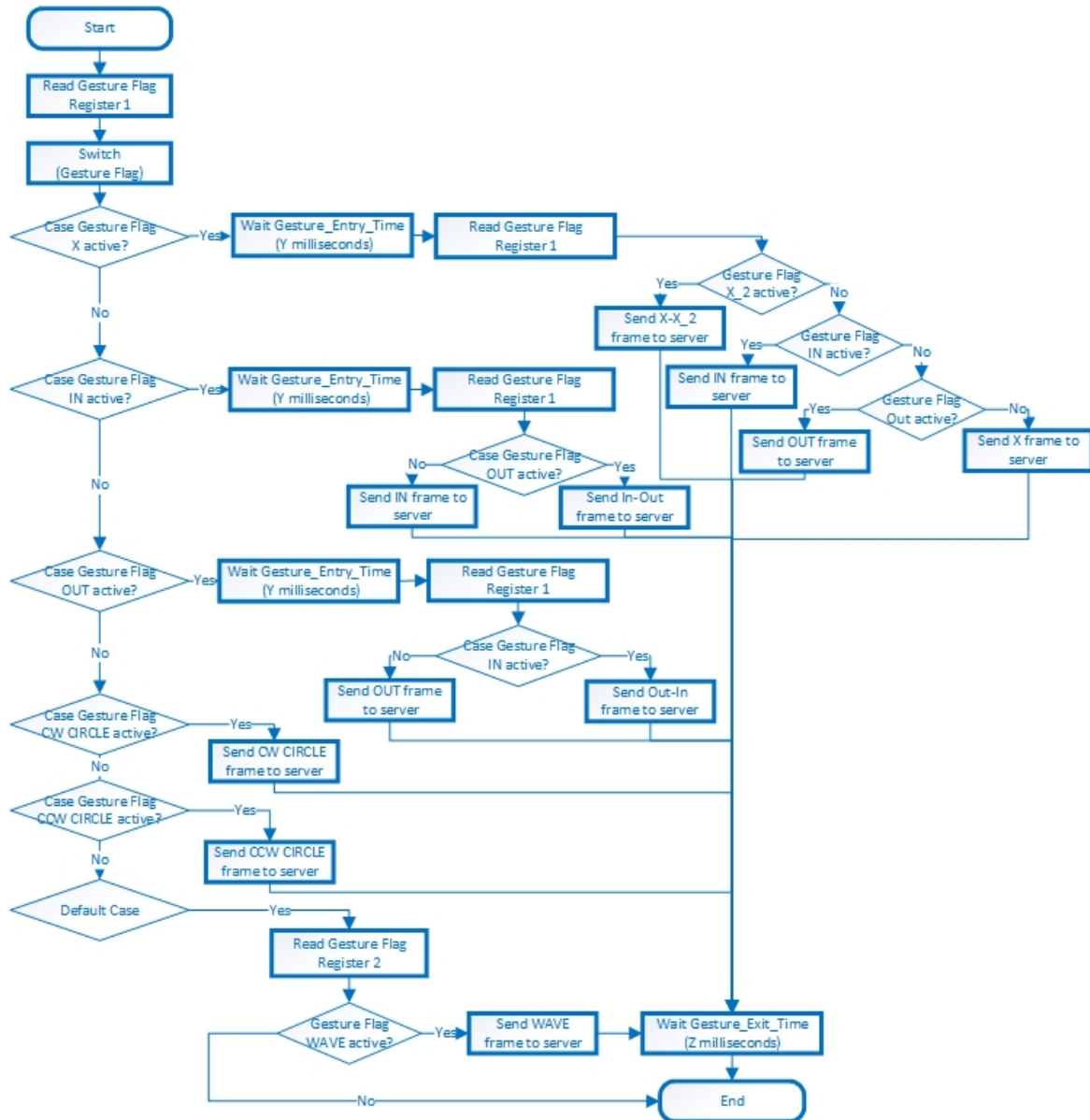


FIGURE 3.18: Gesture detection routine flowchart.

This block is the coding block where the gesture detection occurs. Since the gesture detection, is mainly done on the device itself, the algorithm reads only output registers to proceed with the gesture identification (see Figure 3.18).

Since the device detects 9 base gesture (Up,Down,Right,Left,In,Out,Clockwise Circle, Anti-Clockwise Circle and Wave), a strategy was used to detect 6 additional gesture (Left-Right,Right-Left,Up-down, Down-Up, In-Out and Out-In). The strategy consists in the use of two timed readings instead of only one reading, granting that way the detection of bidirectional gestures. Note that, in the diagram, the "Flag X" represents

four cases (Right, Left, Up, Down flags) and the "FLAG X_2" the same four cases, identifying that way the cases like Left-Right, Right-Left, Up-Down and Down-Up gestures. Another detail is that the "Y" and "Z" represent time intervals between output readings. This block also identifies the communication procedure to the server, explained ahead in the section 3.7.5.

3.7.4 ST VL6180X

The ST VL6180X device, from ST Microelectronics, is a sensing solution based on the ST's patented FlightSense technology. This technology is based on TOF principles making the measurement of the distance independent of the target reflectance, turning the measurement of targets with different colors and surfaces more reliable.

The sensing device can sense both ambient light and proximity. With the measurement of proximity and the combination of multiple sensors, physical motion information (direction, distance or speed) and reliable gesture detection can be achieved.

3.7.4.1 Interface Strategy

The interface between the sensing device and the microcontroller is crucial needed and this interface will be important to configure the device(s) and extract the gesture data from the device(s) itself. This case of study use four sensing devices in a matrix (in a form of a cross). The devices are used to calculate the proximity of an object, in the four points. The interface guarantees that this distances are used to estimate the nature of a gesture.

Main Interface Block

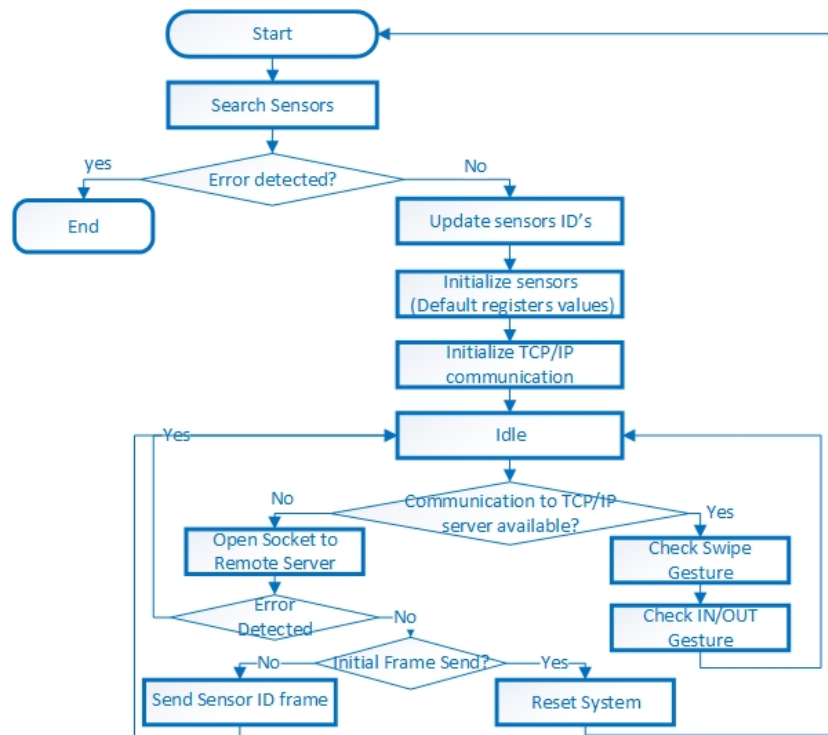


FIGURE 3.19: Main device interface flowchart.

This block, is the main block of the interface code for the ST VL6180X sensing device. Analyzing the Figure 3.19, the main steps are the device array initialization, network initialization, gesture detection and communication to server. This block also contains the method to maintain the communication to the server.

Decode Gesture Block

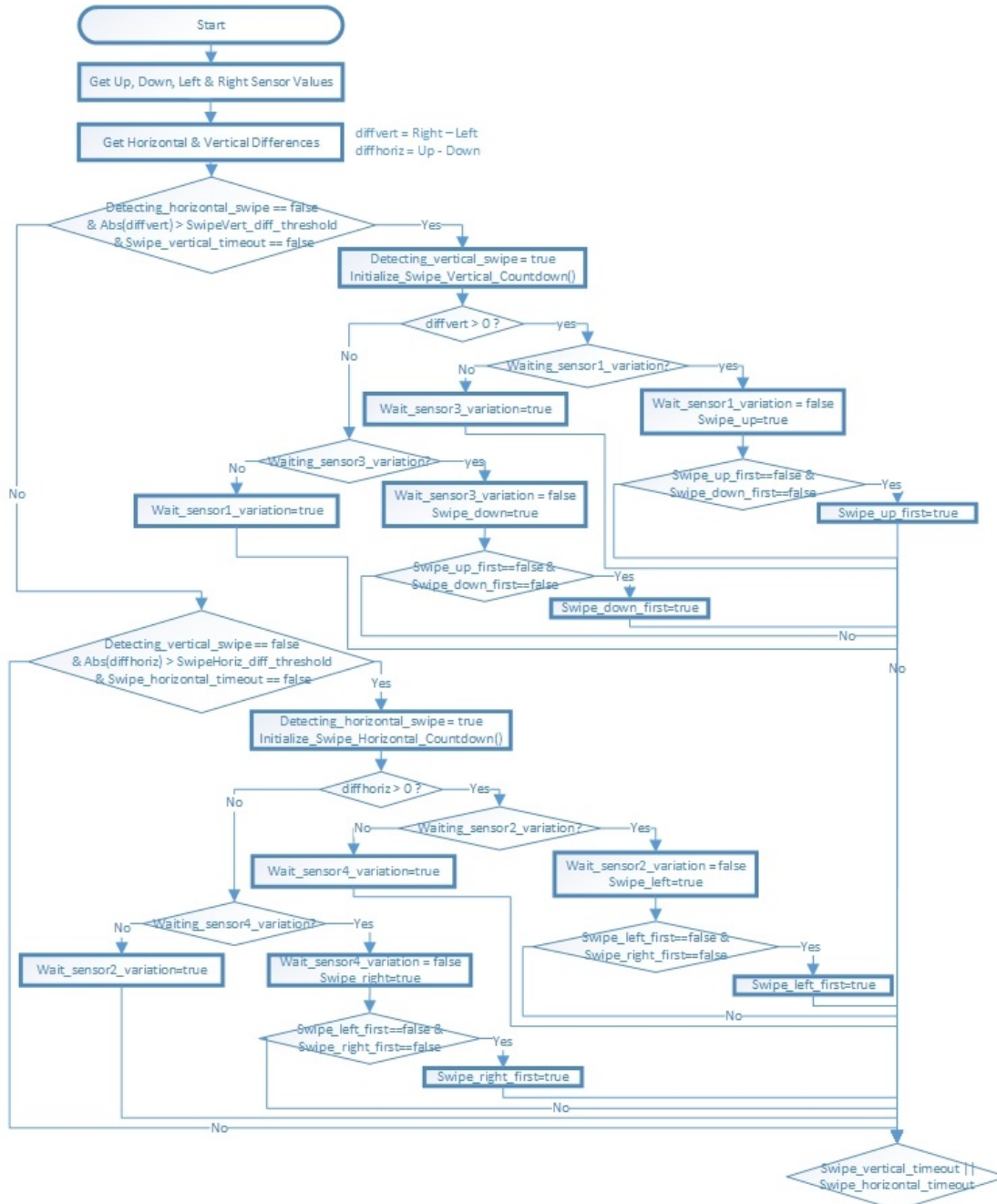


FIGURE 3.20: Gesture detection routine flowchart - Detection Component.

This block, present in Figure 3.20, of the interface for the ST VL6180X is responsible only to detect gestures. The block uses a state machine logic, based on the detection of proximity in the array of sensors.

Using this proximity values, an axis based logic movement, and two possible time-outs, the gestures can be detected for further classification.

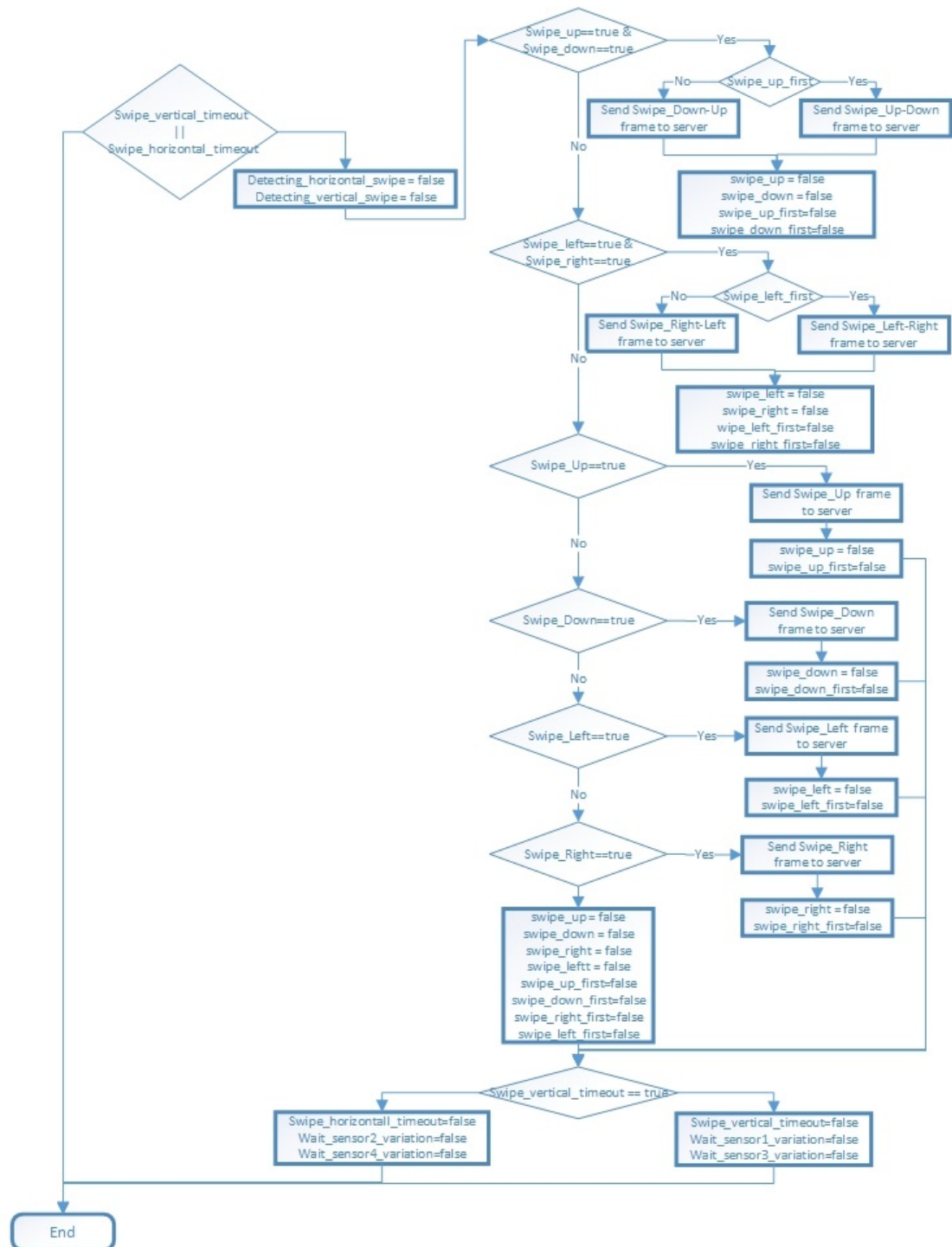


FIGURE 3.21: Gesture detection routine flowchart - Classification Component.

This block consists in a simple coding block activated after a vertical or horizontal gesture timeout, that is responsible for the gesture classification and communication of that classification to the server. For this reason, the block is the main component of this interface.

3.7.5 Remote Server

The Local system, mentioned before in section 3.7.1, is responsible for the gesture detection, processing and recognition.

Other components are important like the remote server used to interface with the local system, simulating the integration of the system in a a driving simulator, like the Bosch DSM. It is important to understand that the server is only a tool to test and validate the system for a future integration is a driving simulator (real world simulator).

To create this server two factors were important: the definition of a communication protocol and the definition of the server architecture. The next topics will cover this two elements in more detail.

3.7.5.1 Communication Protocol

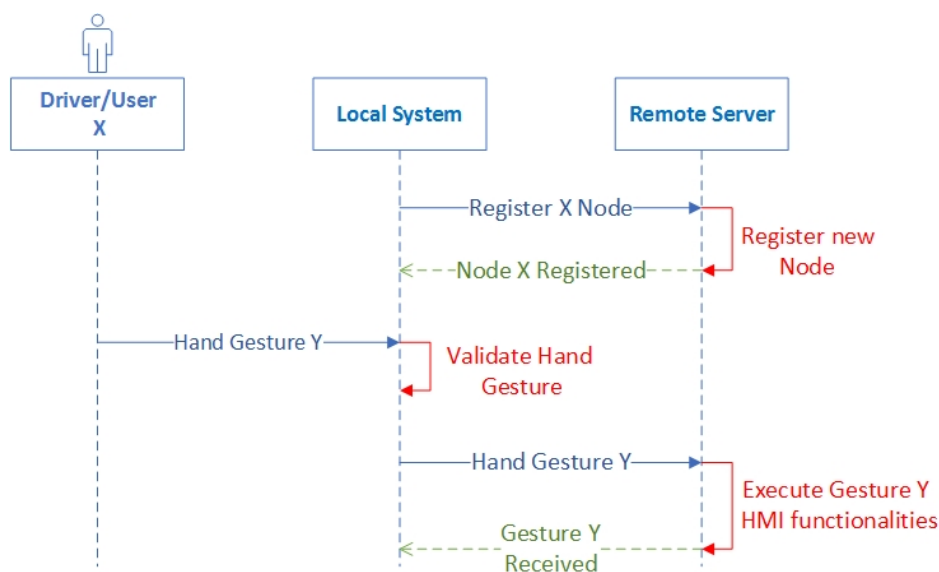


FIGURE 3.22: Remote Server Sequence Diagram.

Analyzing Figure 3.22, it is possible see to the main interactions between user, local system and remote server. The protocol of communication to use should have two type of frames, the register node frames and the gesture frames.

- Register Node Frame Model:
"registernode=Gesture_Sensor_X\n"
- Gesture Frame Model:
"to=Destination_ID from=Gesture_Sensor_X message=<GESTURE_Y>\n"

The register node frame is a simple frame used to register new devices connected to the server where the "Gesture_Sensor_X" represents the ID of the sensing device.

The gesture frames are the frames used to communicate new gestures detected by the sensing device. In this frames the "Destination_ID" represents the destination module, "Gesture_Sensor_X" represents the ID of the sensing device and the "GESTURE_Y" represents the ID of the gesture detected.

3.7.5.2 Remote Server Architecture



FIGURE 3.23: Remote Server Architecture.

Analyzing Figure 3.23, it is possible to see the architecture of the server. The server has a main thread responsible for the initialization of the server and for new client handling.

The server has also N client (N gesture recognition systems) threads to handle with the client requests, where each thread serves a unique client when it is connect to the system.

3.7.5.3 Main Thread



FIGURE 3.24: Main Thread Architecture.

Analyzing the figure 3.24, it is possible see to the structure of the main server thread. This thread is responsible for the network initialization and for adding new clients(gesture recognition systems) to the server, by adding new client threads to the server.

3.7.5.4 Client Thread

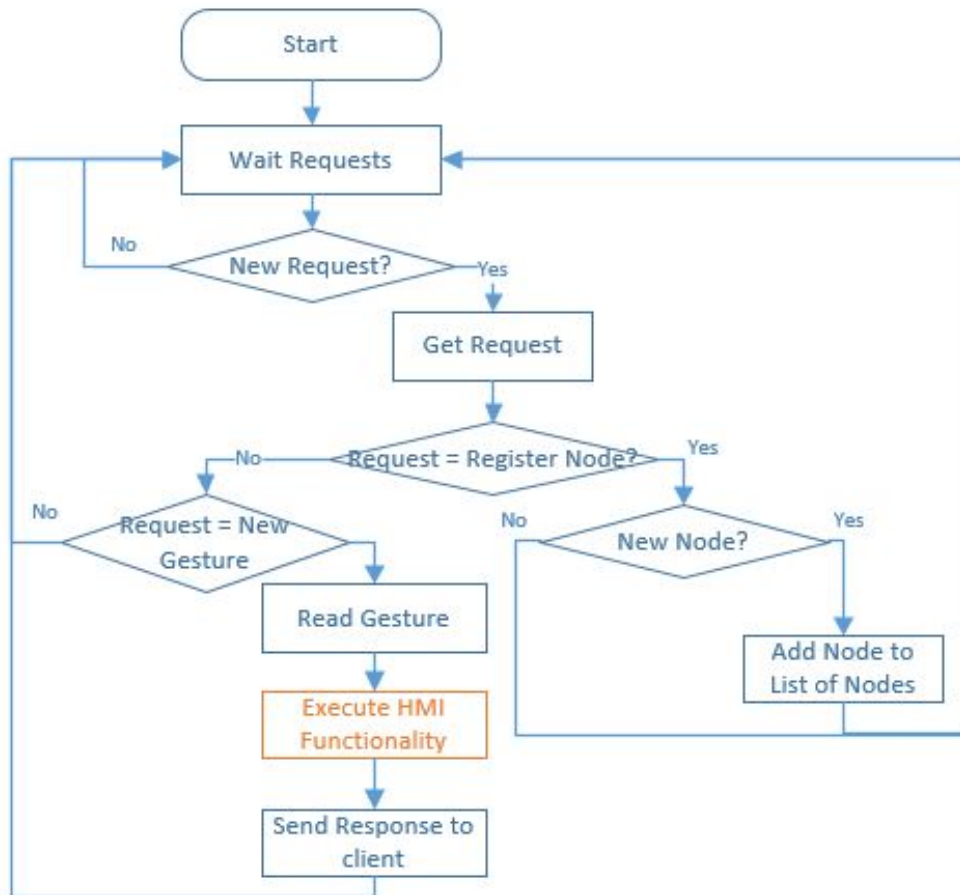


FIGURE 3.25: Client Thread Architecture.

The local system(prototype) should be tested in terms of integration with a external element such as a driving simulator, or in this case, a workstation. This element is important to test the local system functionality.

Paying close attention to figure 3.25 it is possible to see the structure of the client threads. These threads are responsible for adding new clients(new gesture recognition systems) to the list of known clients as well as reading the client requests and sending the response to the clients.

Note that the orange rectangle, represents a HMI feature or functionality, not implemented in the server.

3.7.6 GUI Application

To help the sensing device test and monitoring, a GUI application was developed. The application uses a standard serial interface, being compatible with wired and wireless communication(Bluetooth Serial).

Another aspect is that the application should support more than one sensing device, by developing a common protocol of communication to interface with different sensing devices.

The definition of the communication protocol and the definition of the application architecture are presented in the next topics.

3.7.6.1 GUI Application Mockup



FIGURE 3.26: Remote Server Architecture.

This application mockup, present in Figure 3.26, represents the elements of the desired GUI. The GUI uses 5 main areas: data plot area, log areas (general data log and gesture log), transmit data area, Gesture image area and variable monitoring area.

3.7.6.2 Communication Protocol

To make the application work for different sensing devices and different data inputs, a protocol was designed to guarantee the data processing.

- 1 - Gesture Frame Model:
"<Gesture_X>\n"
- 2 - Light, Color and Proximity Frame Model:
"\$X<value>\n"
- 3 - Gesture Data Frame Model:
"#<channel><value>\n"
- 4 - Application Response Frame Model:
"><message>\n"

Considering the first case, the `Gesture_X` represents the gesture detected by the sensing device. Then considering the second case, `X` represents `A`, `R`, `G` and `B` (for light, red, green and blue values). Also on this case `<value>` represents an integer value from 0 to 65535.

In the third case `<channel>` represents the photodiode ID value (from 1 to 4) and `<value>` represents an integer value from 0 to 65535. Looking at the fourth case, `<message>` represents a generic message between the application and the sensing device.

3.7.6.3 GUI Application Architecture



FIGURE 3.27: GUI Application Architecture.

Looking closely to Figure 3.27 it is possible to see that the GUI application is structured following a thread base architecture allied to a timed selection of routines. The threads mainly deal with the initialization and the data acquisition. The three timeout routines deal with the GUI updating work.

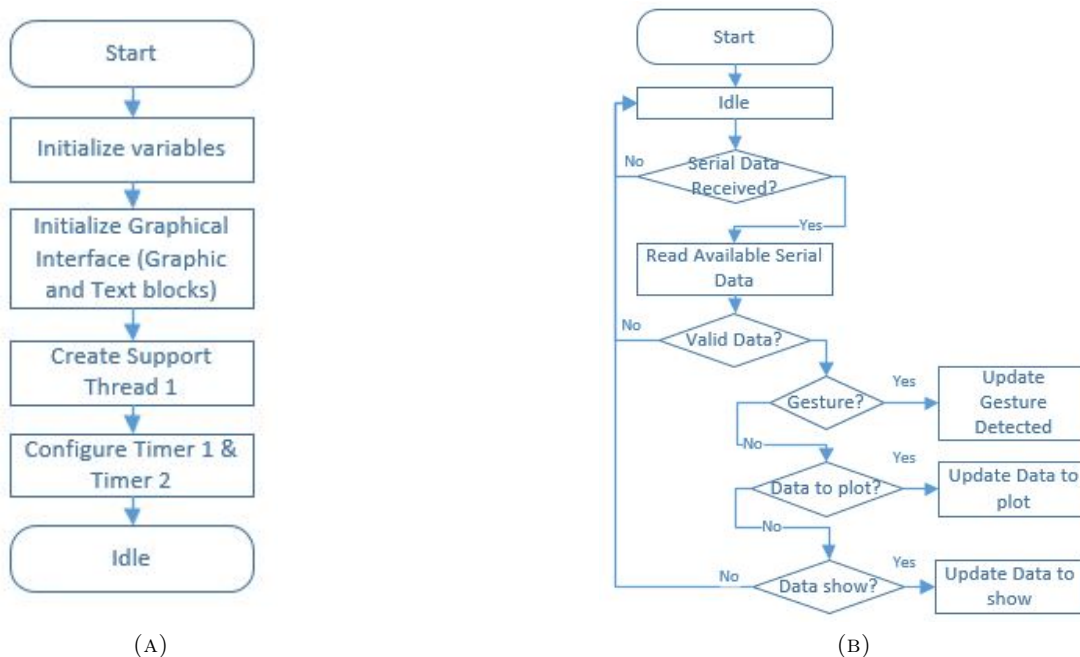


FIGURE 3.28: Main Thread (A) and Support Thread(B).

Analyzing the Figure 3.28(A) it is possible to observe the main thread structure. In this thread the GUI is initialized and all the initializations, support threads creation and support routines calls take place. Seeing figure 3.28(B) it is possible to see the support thread structure. This thread is responsible for the data acquisition, implementing the communication protocol, seen in section 3.7.6.2.

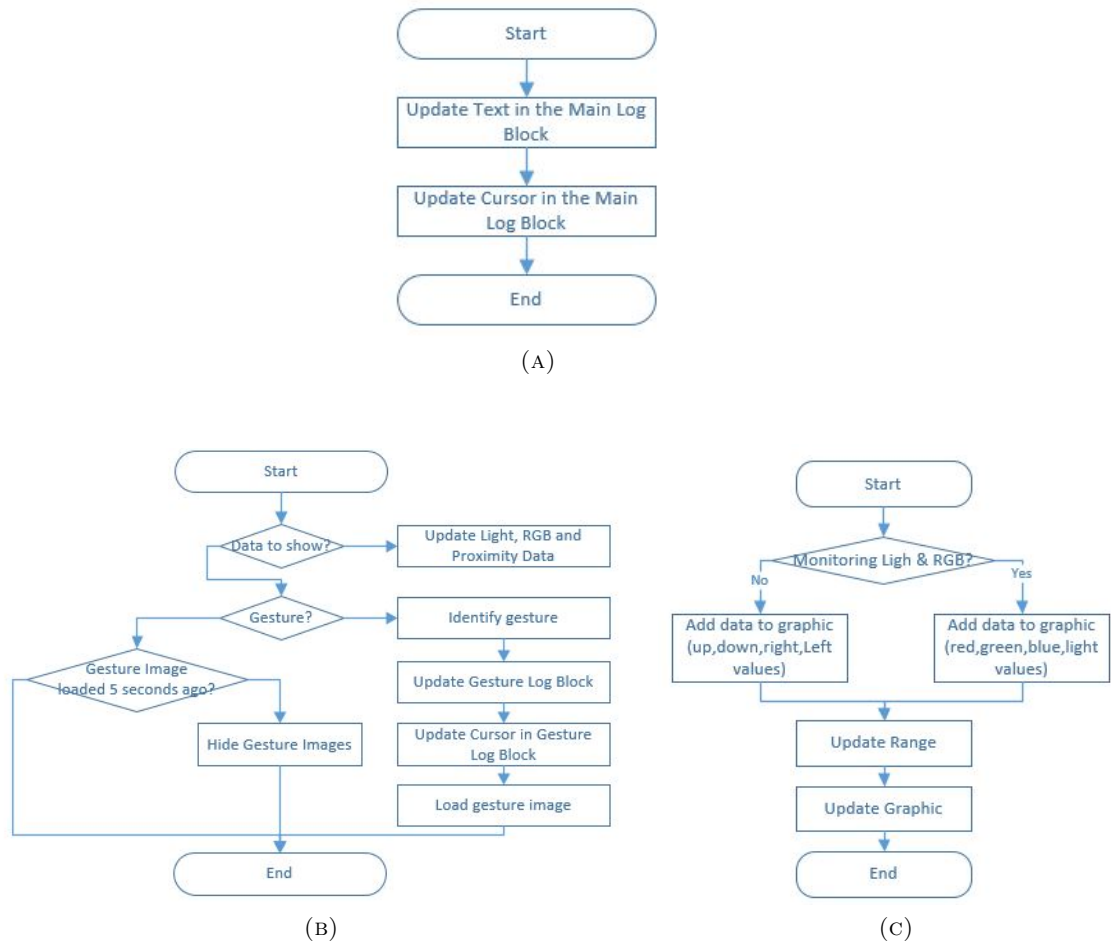


FIGURE 3.29: Timeout Routine1 (A), Timeout Routine2 (B) and Timeout Routine3 (C).

By observing Figure 3.29(A) it is possible to conclude that this routine called upon a timer counting timeout is responsible for updating the main text log block.

Then, Figure 3.29(B) shows another routine responsible for updating the variables under monitoring, the gesture log and gesture image.

Finally, Figure 3.29(C) shows the routine responsible for the data updating in the graph area, both for color and gesture monitoring. The range update is necessary to adjust the graph representation to the new values.

3.8 Conceptual Design

Conceptual Gestures

For the development of this automotive HMI concepts, a set of gestures must be recognized. This set of gestures should be "realistic" because the recognition of the gestures must be granted using IR sensors.

A list of possible gestures to be selected and recognized can have as gestures:

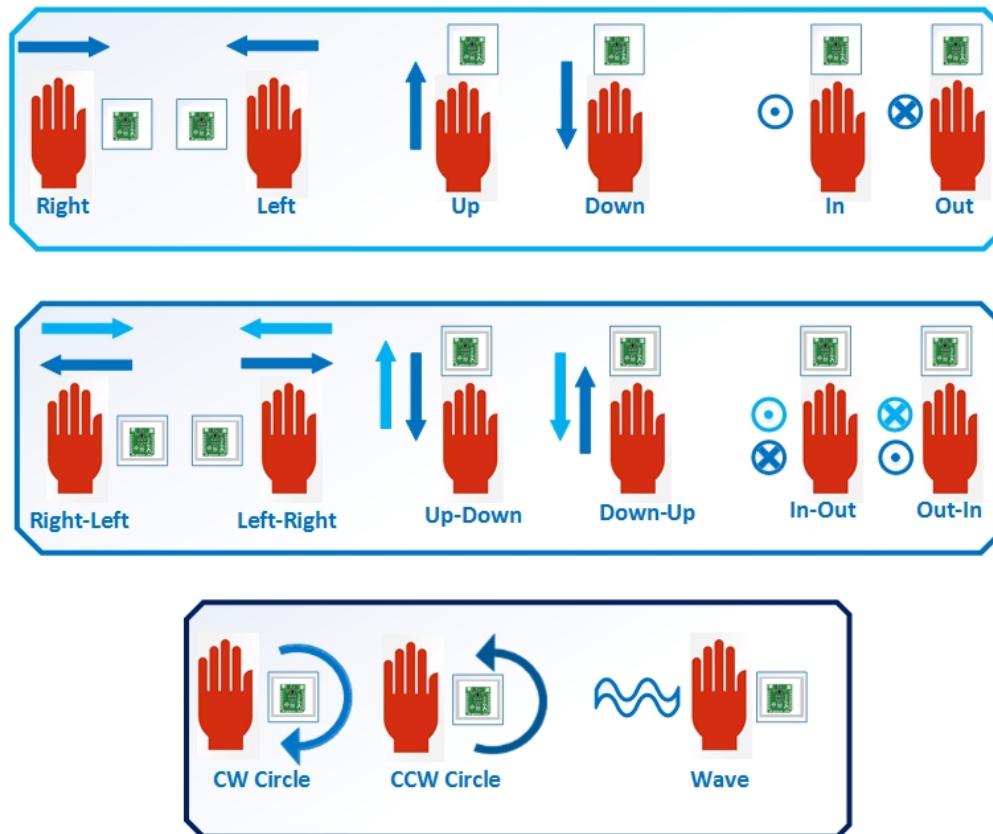


FIGURE 3.30: Basic selection of gestures

Looking at Figure 3.30, the upper block of gestures represents the 6 high priority gestures to implement, the middle block represents the middle priority gestures to implement (developed from the upper block gestures) and the lower block of gestures is the lower priority set of gestures to implement.

Conceptual HMI Design

During the study of possible concepts related to automotive HMI's, a small review of possible concepts was conducted for further analysis. The list of possible concepts can be divided in two types, the concepts outside the car and inside the car.

Concepts Outside the Car

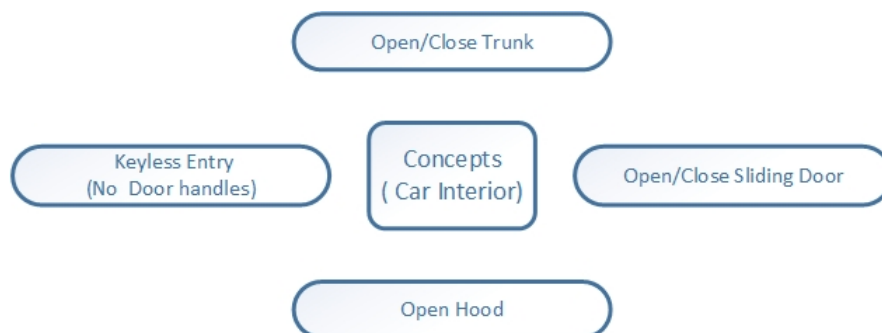


FIGURE 3.31: Basic diagram of concepts outside the car.

Concepts Inside the Car

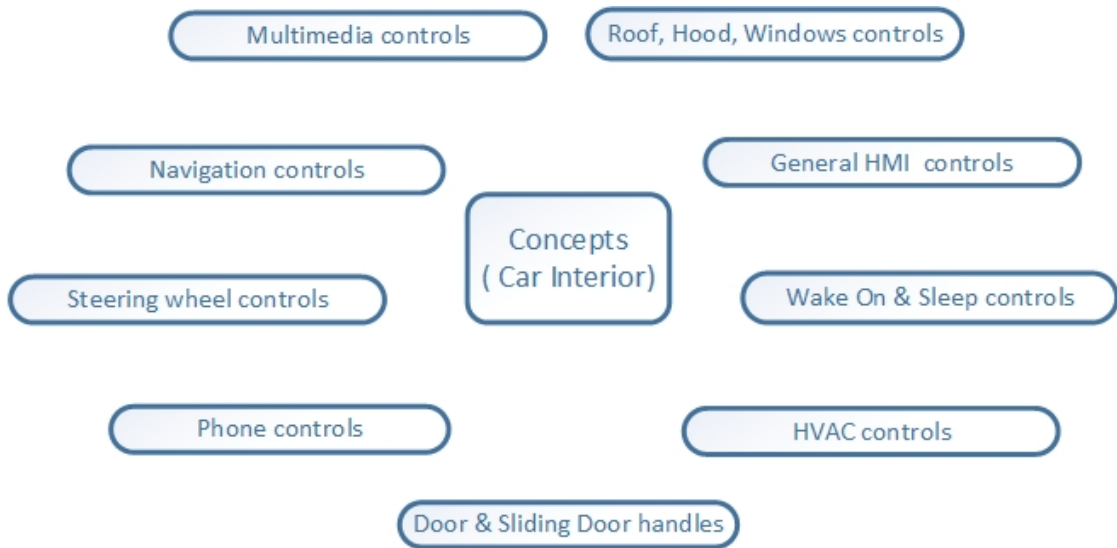


FIGURE 3.32: Basic diagram of in-car concepts.

In the figures above, Figure 3.31 and 3.32, a small list of concepts, both on the exterior and interior of the car, is presented. This list is a result of a market analysis and a personal analysis.

This list of concepts is a base of knowledge to be further studied and validated in the future using the gesture detection techniques used in this project. Furthermore, this list of concepts are used to develop the gesture detection algorithms and test procedures.

3.9 Data Acquisition and Processing

The quantity and quality of data that can be collected from the sensing devices is a key subject for further developments in this dissertation. This section tries to explain what type of data can be collected from the sensing devices and also what type of data treatment can be developed to achieve the desired results for this dissertation.

The following diagram, depicted in Figure 3.33, shows the steps and elements relevant in the process of data acquisition and treatment.

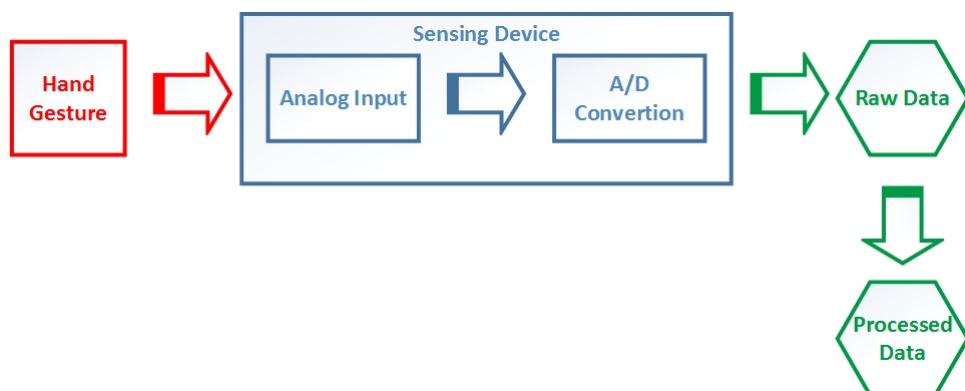


FIGURE 3.33: Data Acquisition and Processing

3.9.1 Raw Data Processing

IR sensing devices are far from perfect and the data collected from them generally correspond to noisy and untreated data. Since factors like temperature, humidity, material reflectivity and other infrared light sources are good examples of sources of interference, to correctly use the sensing devices data, some cares need to be taken in account.

Based on the sensors previously selected, the output data from the sensors (raw data) is scaled in fixed values, from 8 to 16 significant bits, so the range of values goes from 0-255 and from 0-65535, respectively.

Taking the raw data directly from the sensor is always an option, but to work with more adjusted data, a process of filtration and or linearization should be used.

3.9.2 Processed Data Treatment

After the processing done on the raw data and considering as purpose of the system the detection of gestures, higher level processing approaches should be used on this process. For this objective two options were taken into account, the threshold model and the HMM model.

Threshold model

Using this approach the treatment of data is intuitive. The most important steps are the definition of the threshold values to control elements like the gesture and non-gesture classification, the valid gesture timing and even the number of valid samples to classify a set of data as a recognized gesture.

This process of threshold selection is an iterative process, based on more practical and subjective tests, in order to make the system robust to a desired type of environment, a set of trials and errors must exist to calculate the most suitable threshold values.

HMM Model

Using the HMM approach, the working scenario is different. Here, the treatment of data is done on the training and testing of the HMM classifier. After that, the classifier only receives new data, and processes the classification based on the training previously done. The HMM approach is a good options for dynamic hand gestures, because the gestural process is a time-varying process and a system with a variable set of data samples for each gesture. The following scheme, Figure 3.34, shows how the data is treated in the process.

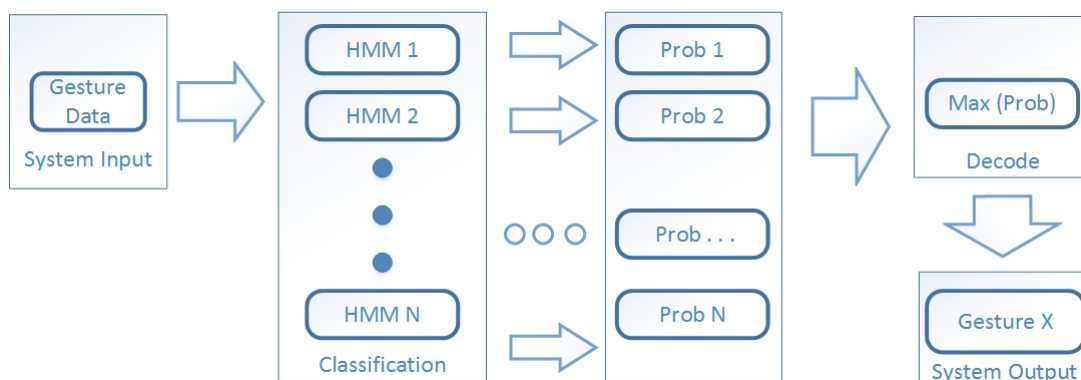


FIGURE 3.34: HMM Approach for gesture recognition

When analyzing the figure above, Figure 3.34, it is possible to see that for each gesture an HMM classification is done.

The input of data to the HMM classifiers is a result of the user gesture, because for each gesture a sequence of time-varying data is generated. The sequential data is processed by the HMM classifier and for each case a likelihood probability is generated.

Based on the likelihood results, the highest likelihood case is tested to pass for a threshold value, confirming if the gesture can be considered valid or not. If the gesture data passes the threshold value then the result will be a valid gesture X, where X represents the gesture ID.

3.9.3 HMM Learning Procedure

HMM classifiers should pass by a phase of supervised training/learning. This training allows the discovery of the HMM parameters for each HMM model.

To do this training process, essential to the creation of the classifiers, a database of gesture data must be created using different persons and gesture types.

To capture real data, the gesture recognition system records the data and then the data is analyzed in Matlab using Kevin Murphy HMM toolbox, where the HMM classifiers are trained with the captured data. For convenience the data is recorded in a dedicated text file, to easily allow a simple access and graphical representation.

3.9.4 Data Validation

Since the HMM approach is based on supervised learning, a database of gesture data must be created and validated.

In the reality the database of gestures should be split in two parts, where around 70% of the data should be used for training data and 30% should be used as testing data. This percentages are a result of the literature analysis, and also in the fact that the total number of samples is not large, so the option should be to create a larger database of training and a smaller database for testing, privileging the database consistence and considering less the test data quantity.

3.9.5 Validation Criteria

For the validation of the Database a few criteria should be taken in consideration, being them:

- **Sensitivity** - metric that measures the probability that a test will indicate a true positive case among the positive cases of a system. This metric is also called the true positive rate or probability of detection, since measures the proportion of positives that are correctly identified.

$$Sensitivity = \frac{TruePositives}{TruePositive+FalseNegative} = \frac{TP}{TP+FN}$$

- **Specificity** - metric that measures the probability that a test will indicate a true negative case among the negative cases of a system. For this reason, the specificity is also called the true negative rate.

$$Specificity = \frac{TrueNegative}{TrueNegative+FalsePositive} = \frac{TN}{TN+FP}$$

- **Accuracy** - metric that measures the probability that a test will indicate a true negative case and a true positive case among the cases of study.

$$Accuracy = \frac{TruePositive+TrueNegative}{TruePositive+FalsePositive+TrueNegative+FalseNegative} = \frac{TP+TN}{TP+FP+TN+FN}$$

- **Confusion Matrix:**

In the area of machine learning and specially considering the problem of statistical classification it is common to use tools like the confusion matrix's, also known as error matrix's (see Figure 3.35).

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

FIGURE 3.35: Confusion Matrix example [97]

These matrix's have a layout of data that easily allows the visualization of the performance of an algorithm.

In these tables/matrix's, each column of the matrix represents the instances in a predicted class and each row represents the instances in an actual class or vice-versa. In figure 3.36, it impossible to see a more practical example of a confusion matrix.

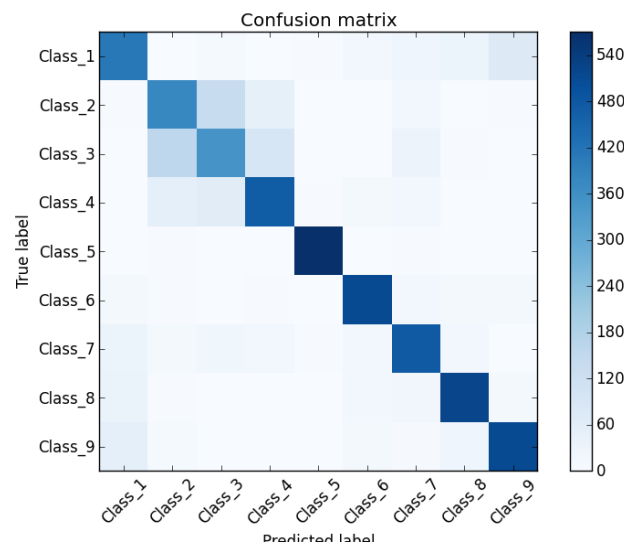


FIGURE 3.36: Confusion Matrix example [98]

3.10 Validation Processes Specification

In this section one of the most important phases and processes is presented, the verification and validation process. There are numerous definitions of verification and validation, but considering an IEEE standard definitions, verification can mean the:

- act of reviewing, inspecting, testing, checking, auditing, or otherwise establishing and documenting items, processes, services or documents considering specific requirements;
- process of determining whether the products of a given phase of the product development life cycle fulfill the requirements established during the previous phase;

On the other side, validation is defined as the:

- evaluation of a product at the end of the product development process to ensure compliance with the user requirements;
- "end-to-end" verification process;

The evaluation and verification methods should be practical and rigorous to make a good analysis of the system. For a more organized procedure, a set of validation steps should be designed. This steps are explained in the next topics.

3.10.1 Experimental Setup

3.10.1.1 Test Configurations

The project validation and verification will be done under specific procedures and metrics. The testing procedures will be done in a controlled in-lab environment, with controlled conditions, such as controlled luminosity, temperature and distraction elements.

The advantages of this type of approach include good experimental control, easy data recording, safety, security and efficiency. On the other side, the main disadvantage of this approach is the lack of "real world" data, but the use of real world procedures is not easy to achieve and is not a priority for this dissertation.

The overall experiments will be performed in a room where the user is completely alone and isolated from external distractions such as noise, other persons or any kind of major distractions.

The procedure will be controlled by a supervisor with visual access to the user. The supervisor will have as function the monitoring of all the collected data by the system and the monitoring of the user to check if the user is performing correctly all the desired test tasks.

3.10.1.2 Test Equipment

For all the testing procedures, namely initial tests, integration tests, and final tests the list of hardware to use will consist in a set of elements such as:

- The prototype of the gesture recognition system, that is composed by the following items:
 - Development board(s);
 - Gesture recognition device(s);
 - Micro SD card reader or Bluetooth-Serial Dongle;

- A testing platform prototype, used for calibration and testing purposes;
- A cluster or local pc running a remote server to handle with the gesture detection;
- A laptop computer, to remotely access and analyze the gesture data previously collected;
- A camera device, to remotely capture the user performance doing the required gestures;
- An oscilloscope, to measure the gesture recognition code execution time for the different gestures;
- A multimeter, to monitor the system and to track possible hardware problems;

The list of software to use will consist in a set of elements such as:

- Serial and TCP/IP monitoring Terminal;
- Keil IDE + MDK-ARM Version 5;
- Mbed Online IDE;
- Matlab R2015b;
- Dedicated TPC/IP server (developed on Qt 5.5.0)
- Dedicated GUI application(developed on Qt 5.5.0), with:
 - Serial interface;
 - Gesture recognition animations;
 - Gesture data recording;
- Dedicated GUI application for demonstration purposes(developed on Qt 5.5.0);

3.10.2 Test Methods

3.10.2.1 Initial tests

This phase comprehends all the initial/individual testing and integration, in terms of the elements such as the development board, sensing devices, interface code and software tools.

Starting with the development board, besides the study of functionalities, tests and demos should be conducted using the selected software tools. The tests should cover the functionality of the software tools and the test of the development board peripherals and interfaces. Now, considering the gesture recognition devices/sensors, their testing procedure should be done in two steps, being them:

- communication/interface tests, where the interface code between the board and sensing devices is tested;
- functional tests, where the capabilities of the sensors are tested, as well as the final interface codes.

3.10.2.2 Integration tests

This phase is formed by a set of tests, used to test the individual components integration on the system and also the system integration with the cluster simulating the integration in an automotive driving simulator. Moreover the tests will be conducted on a dedicated bench test platform.

The tests to conduct will consist on:

- communication tests using the gesture recognition system and the cluster; the tests consist in the development of demos, with information packets to test the information exchange operations;
- functional testing, where the system is tested in terms of efficiency of detection; in this tests a small group people will be used to register real data for a better analysis of the system efficiency. This step will be divided in two phases:
 - set of simple gesture recognition tests, where the data is collected for further analysis using an auxiliary memory(micro SD card) or the remote pc trough bluetooth-serial interface;
 - basic execution time measurements, trough simple time measurement using the development board I/O peripherals and an oscilloscope.

3.10.2.3 Final tests

This phase is the final step in all the testing procedures. This phase is formed by a battery of tests, that is designed to have 10 to 30 people as elements of study. The test should be based in a set of users with different ages and sexes, simulating the variety of "real world" users. This subjects of study should perform 50 times the same gesture, if possible with 3 hand positions and speeds, but with no time limit for the test execution, simulating a real world utilization of the system.

3.11 Prototype Testing Platform

Nowadays, the variety of sensing devices is considerable and many of them are good enough out of the box for simple non-critical applications. Besides this fact, to know the practical response of the sensing devices and to achieve the best response to a specific use case, the sensing devices should be tested and calibrated for the particular system and configuration where they are used. For the initial implementation and testing phase, the development of a prototyping measurement and calibration platform was considered.

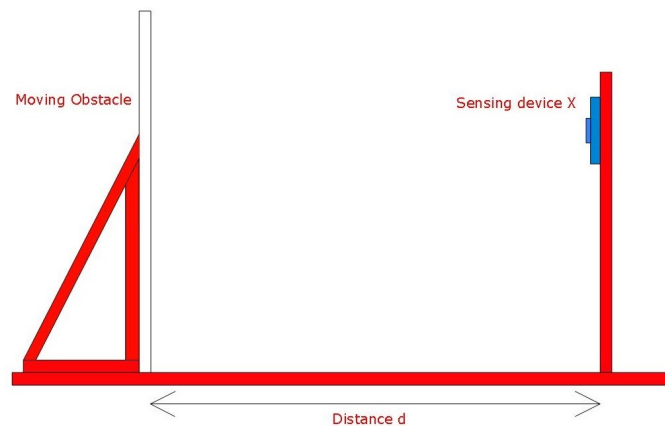


FIGURE 3.37: Prototype Testing Platform Mockup

This platform, characterized by a great simplicity of design and operation, is inspired in other existing testing platforms [99–101], and can be used to trace the response of different sensors in terms of proximity measurements. The designed platform, see Figure 3.37, contains a moving element(obstacle), to use in the proximity(distance) variation and a support where the sensing devices are fixed.

Chapter 4

System Implementation

In the fourth chapter, it is described and explained the implementation phase of this dissertation. Here, important implementation choices and details are presented and discussed.

As shown on the last chapter, the system is composed by a local system, composed by a MCU and a set of sensors, a cluster and a remote pc, used only on the monitoring process. Still on the chapter three, the hardware and software tools and the global system architecture were presented. Then a brief conceptual design was presented as well as the data acquisition process and the system validation process.

The implementation process of this thesis, can be divided in terms of hardware and software. In terms of hardware, the main focus goes to the hardware test/measurement platform used for testing the gesture recognition devices.

In terms of software, the main focus goes to the presentation of important software implementations, previously mentioned in the last chapter.

4.1 System Architecture Implementation

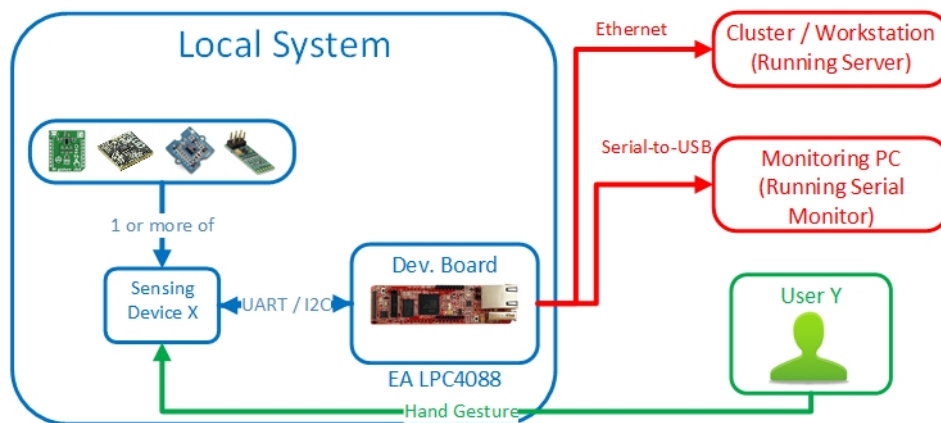


FIGURE 4.1: Local System simple diagram.

Looking at Figure 4.1, it is possible to observe the structure already explained on the last chapter, by presenting a system composed by a local system, a remote server and a monitoring pc.

To maintain the same line of thinking, the same structure used in the section 3.7 will be used in this section, meaning that the following topics will follow the same presentation order. In order to do this, the implementation will be divided in the local system, remote server and GUI application topics.

4.1.1 Local System

4.1.1.1 Avago APDS-9960 and ams TMG4903 Interfaces

Main Routine

```

void main(void) {
...
while(1)
{
if(RunningMode==true)
{
if (GSensor.isGestureAvailable()) {
res=GSensor.readGesture();
DecodeGesture(res);
}
if(pc.readable())
{
serial_print_info();
}
}
else
{
if(OpenSock(Socket,(char *)Ethernet_SERVER_IP,Ethernet_Port) == false)
{
RunningMode = false;
}
else {
if(SendSimpleSocket(Socket,(char *)eth_init)!=true) mbed_reset();
RunningMode = true;
}
}
}
...
}

```

LISTING 4.1: Main Routine Highlights

Considering the extract of code above, it is not possible to see where the sensors and network initialization is done, mainly because these steps are dependent on each sensors and not especially relevant to understand this routine. Instead, it is possible to see that the gesture detection and classification are divided in two main steps, explained during design phase, using the method "isGestureAvailable" and "readGesture". Another aspect to consider is the communication strategy explained before. Basically, the system checks if it is connected via TCP/IP to the remote server, and if not tries to connect, sending at first an initialization frame. If this fails the system suffers a reset, to try to make a new connection. Another aspect to consider is the serial interface method "readable" used to check if a new serial request is available, and the "serial_print_info" used to output relevant information from the system.

Gesture Detection Routine

```

bool isGestureAvailable()
{
...
if( !ReadDataByte(APDS9960_GSTATUS, val) ) {
return ERROR;
}
/* Shift and mask out GVALID bit */
val &= APDS9960_GVALID;
...
}

```

LISTING 4.2: Gesture Detection Routine

Looking at the extract of code above, the strategy previously presented in the design phase is followed. The strategy is to read the available gesture flag, and check if new gesture data is available. Since the flag is available in a status register, this register is read and interpreted. In this step, the difference in the implementation between both sensing device interfaces is only the register being read.

Read Gesture Routine

```
int readGesture() {
    ...
    // If there is data on the FIFO than read each block of 4 datasets (4 bytes)
    if( fifo_level > 0 ) {
        bytes_read = ReadDataBlock(APDS9960_GFIFO_U,(uint8_t*)fifo_data,(fifo_level*4));
        if( bytes_read == -1 ) return ERROR;
    }
    // Check if f one or more datasets (1 dataset = 4 bytes) are available
    if( bytes_read >= 4 ) {
        for( i = 0; i < bytes_read; i += 4 ) {
            if(SERIAL_MONITORING) {
                pc.printf("#1%d\n",fifo_data[i]);
                pc.printf("#2%d\n",fifo_data[i+1]);
                pc.printf("#3%d\n",fifo_data[i+2]);
                pc.printf("#4%d\n",fifo_data[i+3]);
            }
            gesture_info.up_data[gesture_info.index] = fifo_data[i];
            gesture_info.down_data[gesture_info.index] = fifo_data[i + 1];
            gesture_info.left_data[gesture_info.index] = fifo_data[i + 2];
            gesture_info.right_data[gesture_info.index] = fifo_data[i + 3];
            gesture_info.index++;
            gesture_info.total_gestures++;
        }
        // Process gesture data and if a gesture flag is raised decode the gesture
        if( processGestureData() ) {
            decodeGesture();
        }
        // Reset auxiliar variables
        gesture_info.index = 0;
        gesture_info.total_gestures = 0;
        ...
    }
}
```

LISTING 4.3: Gesture Processing Routine

Analyzing the extract of code above, the strategy previously showed in the design phase is strictly implemented. The strategy is to read the gesture data FIFO's from each sensor (maximum of 128 and 256 bytes for AVAGO APDS9960 and ams TMG4903). To achieve this, in the AVAGO case showed above, at least a data set of 4 bytes is read (1 byte for each photodiode) and saved to auxiliary data structures. In the case of ams TMG4903 the main difference is in the number of bytes read, because since the sensor has a resolution of 14 bits, each data set is composed by 8 bytes (2 bytes for each photodiode), so at least 8 bytes must be read each time. For this reason the auxiliary data structure of ams TMG4903 has a different size (128 vs. 256 bytes).

Gesture Data Processing Routine

```
bool processGestureData()
{
    ...
    // Calculate the first and last ratios in x and y axis (left/right ratio and up/
    // down ratio)
    UD_ratio_first = ((u_first - d_first) / (u_first + d_first)) * 100;
```



```

LR_ratio_first = ((l_first - r_first) / (l_first + r_first)) * 100 ;
UD_ratio_last = ((u_last - d_last) / (u_last + d_last)) * 100;
LR_ratio_last = ((l_last - r_last) / (l_last + r_last)) * 100;

// Determine the variation/difference between both ratios
UD_delta = UD_ratio_last - UD_ratio_first;
LR_delta = LR_ratio_last - LR_ratio_first;

// Accumulate the variations/differences in both axis
gesture_UD_delta += UD_delta;
gesture_LR_delta += LR_delta;

// If the variation in y axis (up/down) is greater then the sensitivity threshold
if( gesture_UD_delta >= GESTURE_SENSITIVITY_UD ) gesture_UD_count = 1;
else if( gesture_UD_delta <= -GESTURE_SENSITIVITY_UD ) gesture_UD_count = -1;
else gesture_UD_count = 0;

// If the variation in x axis (left/right) is greater then the sensitivity thresh
// old
if( gesture_LR_delta >= GESTURE_SENSITIVITY_LR ) gesture_LR_count = 1;
else if( gesture_LR_delta <= -GESTURE_SENSITIVITY_LR ) gesture_LR_count = -1;
else gesture_LR_count = 0;

/* Determine IN/OUT gesture */
if( !gesture_UD_count && !gesture_LR_count ) {
    if( (abs(UD_delta) < GESTURE_SENSITIVITY_IO) && (abs(LR_delta) < GESTURE_SENS
ITIVITY_IO) ) {
        if( (!UD_delta) && !LR_delta ) gesture_IN_count++;
        else if( UD_delta || LR_delta ) {
            gesture_out_count++;
        }
        if( (gesture_IN_count >= IN_THRESHOLD) && (gesture_OUT_count >= OUT_THRES
HOLD) ) {
            if( !UD_delta && !LR_delta ) gesture_status = IN_STATE;
            else if( UD_delta && LR_delta ) gesture_status = OUT_STATE;
            return true;
        }
    }
} else {
    if( (abs(UD_delta) < GESTURE_SENSITIVITY_IO) && (abs(LR_delta) < GESTURE_SENS
ITIVITY_IO) ) {
        if( !UD_delta && !LR_delta ) {
            gesture_IN_count++;
        }
        if( gesture_IN_count >= IN_COUNT ) {
            gesture_UD_count=0, gesture_LR_count=0, gesture_UD_delta=0, gesture_
LR_delta=0;
        }
    }
}
return false;
...
}

```

LISTING 4.4: Gesture Data Processing Routine

Looking at the extract of code above, it is possible to see that steps like the calculation of the first and last valid gesture samples are omitted. Instead it is shown how that data is treated. Basically the strategy for each one of the sensors is to:

- Calculate the first and last valid horizontal and vertical photodiodes ratios (up-down and left-right ratios);
- Calculate the vertical and horizontal photodiodes ratio variations;
- Accumulate the variations, in the horizontal and vertical axis, and verify if they are above the sensitivity thresholds, updating the control variable for up/down and left/right movements (`gesture_UD_count` and `gesture_LR_count`);

- In the IN/OUT gesture classification, verify if the variations are under the threshold and update the counter for either in or out gestures. If the counter "gesture_IN_count" is above the IN_THRESHOLD (15) and "gesture_OUT_count" is under the OUT_THRESHOLD(2) the gesture is classified before further decoding.

If on the in or out gestures the right/left or up/down counters are not zero (swipe gesture being detected), the absolute variations of the ratios ("UD_delta" and "LR_delta") are analyzed and if lower than a sensitivity threshold ("GESTURE_SENSITIVITY_IO"), the in gesture counter can be updated and the other gesture being detected can suffer a reset (if "gesture_IN_count" > IN_COUNT).

Gesture Decode Routine

```
bool decodeGesture()
{
    ...
    /* Return if near or far event is detected */
    if( gesture_status == IN_STATE ) {
        gesture_motion = DIRECTION_IN;
        return true;
    } else if ( gesture_status == OUT_STATE ) {
        gesture_motion = DIRECTION_OUT;
        return true;
    }
    /* Determine swipe direction */
    if( (gesture_UD_count == -1) && (gesture_LR_count == 0) ) gesture_motion = DIRECTION_UP;
    else if( (gesture_UD_count == 1) && (gesture_LR_count == 0) ) gesture_motion = DIRECTION_DOWN;
    else if( (gesture_UD_count == 0) && (gesture_LR_count == 1) ) gesture_motion = DIRECTION_RIGHT;
    else if( (gesture_UD_count == 0) && (gesture_LR_count == -1) ) gesture_motion = DIRECTION_LEFT;
    else if( (gesture_UD_count == -1) && (gesture_LR_count == 1) )
    {
        if( abs(gesture_UD_delta_) > abs(gesture_LR_delta_) ) gesture_motion = DIRECTION_UP;
        else gesture_motion = DIRECTION_RIGHT;
    }
    else if( (gesture_UD_count == 1) && (gesture_LR_count == -1) )
    {
        if( abs(gesture_UD_delta) > abs(gesture_LR_delta_) ) gesture_motion = DIRECTION_DOWN;
        else gesture_motion = DIRECTION_LEFT;
    }
    else if( (gesture_UD_count == -1) && (gesture_LR_count == -1) ) {
        if( abs(gesture_UD_delta_) > abs(gesture_LR_delta) ) gesture_motion = DIRECTION_UP;
        else gesture_motion = DIRECTION_LEFT;
    }
    else if( (gesture_UD_count == 1) && (gesture_LR_count == 1) ) {
        if( abs(gesture_UD_delta_) > abs(gesture_LR_delta_) ) gesture_motion = DIRECTION_DOWN;
        else gesture_motion = DIRECTION_RIGHT;
    }
    else {
        return false;
    }
    ...
    switch(gesture_motion) {
        ...
        case DIRECTION_RIGHT:
            if(SendSimpleSocket(Socket, (char *)eth_right_gesture)!=true) mbed_reset();
            pc.printf("RIGHT\n");
            break;
    }
}
```

```

    ...
}
    ...
return true;
}

```

LISTING 4.5: Gesture Decode Routine

Observing the extract of code above, it is possible to see how the gestures are decoded. In the case of IN/OUT gestures the decoding is done just by detecting the value of the gesture states `IN_STATE` and `OUT_STATE`, previously updated, on the `processGestureData()` routine.

In the case of the other gestures, the states of the gestures are analyzed (“gesture_UD_count” and “gesture_LR_count”) and by direct comparing the values with the reference values for each one of the quadrants, the identification is done. To solve redundant cases, the variation ratios are also analyzed (“gesture_UD_delta” and “gesture_LR_delta”).

In order to communicate the gestures detected, the routine also incorporates the communication layer by reporting the decoded gesture either by serial interface (just for monitoring) or TCP/IP interface (to the cluster running remote server). A special note goes for the communication to the server, where if the communication fails, the system suffers a reset, to try the reconnection to the server.

4.1.1.2 PixArt PAJ7620U2 Interface

Main Routine

```

...
while(1)
{
    if(RunningMode==true)
    {
        if (is_Gesture_Available()) // Read Registers 0x43 and 0x44 - Gesture Flags
        {
            Decode_Gesture_Complete(data);
            if(pc.readable())
            {
                serial_print_info();
            }
        }
    }
    else
    {
        if(OpenSock(Socket,(char *)Ethernet_SERVER_IP,Ethernet_Port) == false)
        {
            RunningMode = false;
        }
        else {
            if(SendSimpleSocket(Socket,(char *)eth_init)!=true) mbed_reset();
            RunningMode = true;
        }
    }
}
...

```

LISTING 4.6: Main Routine Highlights

Considering the extract of code above, the sensors and network initializations are omitted. Instead, it is possible to see that the gesture detection and classification is divided in two main steps, explained during system architecture design process, using

the method "is_Gesture_Available()" and "Decode_Gesture_Complete".

It is also possible to see the communication strategy already explained both on the design phase and on the Avago APDS-9960/ams TMG 4903 interface in section 4.1.1.1. Furthermore it is also possible to see the communication to the monitoring pc, via serial interface, with the objective of sending debug and testing data.

Gesture Decode Routine

```

void decode_Gesture_Complete(uint8_t data_main)
{
...
switch (data_main)
{
case GES_RIGHT_FLAG:
wait_ms(GES_REACTION_TIME);
PAJ7620U2_ReadReg(0x43, 1, &data_main);
if(data_main == GES_LEFT_FLAG) {
pc.printf("_RIGHT-LEFT_ \n");
if(SendSimpleSocket(Sock,(char *)eth_right_left_gesture)!=true) mbed_reset();
wait_ms(GES_QUIT_TIME);
}
else if(data_main == GES_FORWARD_FLAG) {
pc.printf("_IN_ \n");
if(SendSimpleSocket(Sock,(char *)eth_in_gesture)!=true) mbed_reset();
wait_ms(GES_QUIT_TIME);
}
else if(data_main == GES_BACKWARD_FLAG) {
pc.printf("_OUT_ \n");
if(SendSimpleSocket(Sock,(char *)eth_out_gesture)!=true) mbed_reset();
wait_ms(GES_QUIT_TIME);
}
else {
pc.printf("_RIGHT_ \n");
if(SendSimpleSocket(Sock,(char *)eth_right_gesture)!=true) mbed_reset();
wait_ms(GES_QUIT_TIME);
}
break;
...
case GES_CLOCKWISE_FLAG:
pc.printf("_CW-CIRCLE_ \n");
if(SendSimpleSocket(Sock,(char *)eth_cw_circle_gesture)!=true) mbed_reset();
wait_ms(GES_QUIT_TIME);
break;
...
...
}

```

LISTING 4.7: Gesture Decode Routine

Analyzing the extract of code above, it is possible to see part of the decode process in the non-interrupt based gesture detection approach.

Following this strategy already explained in the system architecture design process, the approach is to read the gesture detection flags present in two registers (0x43h and 0x44h). Furthermore the strategy passes by 1 or 2 timed readings controlled by 1 or 2 main delays, the "GES_REACTION_TIME" and the "GES_QUIT_TIME".

The "GES_REACTION_TIME" is a delay inserted in the detection of sequential gestures, and the "GES_QUIT_TIME" is responsible for a pause between each gesture detection. This approach easily allows the detection of up to 15 gestures, either them being simple or sequential gestures.

4.1.1.3 ST VL6180X Interface

Main Routine

```

...
while(1) {

if(RunningMode==true)
{
// Decode Swipe Gestures
Check_Decode_Gestures();

//Decode IN/OUT Gestures
if( OutControl(2) == true ) {
print_Out_Gesture();
} else if( InControl(4) == true ) {
print_IN_Gesture();
}

if(pc.readable())
{
serial_print_info();
}
} else {
if(OpenSock(Sock,(char *)Ethernet_SERVER_IP,Ethernet_Port) == false)
{
RunningMode = false;
}
else {
if(SendSimpleSocket(Sock,(char *)eth_init)!=true) mbed_reset();
RunningMode = true;
}
}
}
}
...

```

LISTING 4.8: Main Routine Highlights

Considering the extract of code above, the sensors and network initializations are not showed. Instead, it is possible to see that the gesture detection and classification is divided in two main steps, the processing of swipe gestures (using `Check_Decode_Gestures()` routine) and the processing of In/Out gestures (using `OutControl(X)` and `InControl(Y)`, where X and Y represente the sensor being used in the monitoring).

The detection of In/Out gestures is achieved almost directly, based of the sensor proximity result(estimated from raw data to millimeters), with the detection of swipe gestures being explained in detail in the next section.

In the extract code above it is also possible to see the communication strategy already explained before in the last topics and on the system architecture design.

```

void SwipeControls(void const *args)
{
...
diffhoriz = distance2 - distance4;
diffvert = distance1 - distance3;

// CALCULATE DIFFERENCE IN THE 2 VERTICAL SENSORS ( 1 AND 3), ACTIVATE TIMEOUT
// AND GESTURE FLAGS
if( detecting_horizontal==false && !WaitCountDown_SV && (abs_num(diffvert) > Swi
peVert_diff_threshold) )
{
detecting_vertical=true;
CountDown_SV.attach(&CountDown_SV_Callback, SwipeVert_timeout);
if (diffvert > 0)
{

```

```

    if (wait_sensor_1_variation==true) {
        wait_sensor_1_variation=false;
        swipe_up=true;
        if(swipe_up_first==false && swipe_down_first==false) swipe_up_first=true;
    }
    else {
        wait_sensor_3_variation=true;
    }
}
else
{
    if (wait_sensor_3_variation==true) {
        wait_sensor_3_variation=false;
        swipe_down=true;
        if(swipe_up_first==false && swipe_down_first==false) swipe_down_first=true;
    }
    else {
        wait_sensor_1_variation=true;
    }
}
}
...

// CLEAN AUXILIAR VARIABLES & PRINT GESTURE TYPE IF DETECTED, AFTER GESTURE TIMEOUT
if(WaitCountDown_SH || WaitCountDown_SV)
{
    detecting_horizontal=detecting_vertical=false;
    // PRINT DETECTED GESTURE
    if(swipe_up && swipe_down)
    {
        if(swipe_up_first) {
            pc.printf("\n-----\n\tSWIPE UP-DOWN\n-----\n");
            if(SendSimpleSocket(Sock,(char *)eth_up_down_gesture)!=true) mbed_reset();
        } else {
            pc.printf("\n-----\n\tSWIPE DOWN-UP\n-----\n");
            if(SendSimpleSocket(Sock,(char *)eth_down_up_gesture)!=true) mbed_reset();
        }
        swipe_up=false, swipe_down=false, swipe_up_first=false, swipe_down_first=false;
    }
    ...
    else if(swipe_up)
    {
        pc.printf("\n-----\n\tSWIPE UP\n-----\n");
        if(SendSimpleSocket(Sock,(char *)eth_up_gesture)!=true) mbed_reset();
        swipe_up=false, swipe_up_first=false;
    }
    ...
}
...
}

```

LISTING 4.9: Gesture Decode Routine

Looking at the extract of code above, it is possible to understand the strategy used in the detection of gestures. Basically the distance data is analyzed, by calculating the variation in the pairs of horizontal and vertical sensors, then the difference is compared to the vertical and horizontal thresholds. In the extract of code it is possible to analyze the vertical swipe detection, where basically the strategy is to detect the sensor being covered by a hand or finger, and activating a counting interval. If the opposite sensors is covered in the counting interval a swipe is detected. Sequential swipe gestures can also be detected if executed in the counting interval. The extract of code also shows that the communication of a possible gesture, done after the counting sequence, is used in the gesture detection. In this last step if the communication fails, the system suffers a forced reset, to reinitialize the system.

4.1.2 Remote Server Interface

Main Thread

```

...
while(1)
{
    ...
    clisockfd = accept(sockfd, (struct sockaddr *)&cli_addr, &clilen);
    if (clisockfd < 0) {
        perror("ERROR on accept");
        return(1);
    }
    fprintf(stdout, "Client Connected %d\n", clisockfd);
    pthread_create(&vec_threads[count++], NULL, ClientHandler, (void*) &clisockfd);
    ...
}
...

```

LISTING 4.10: Main Thread Highlights

Considering the extract of code above, it is possible to observe the main function of this thread, that is basically to connect new clients to the server. In this case, this thread uses the system call "accept" to extract the first connection request on the queue of pending connections, creating a new connected socket. Then, this thread creates a new thread to answer the new client, in this case, a new local system (gesture recognition system).

Client Threads

```

void * ClientHandler(void* socketFD) {
    ...
    while(getline(&line, &bufSize, socIN) >= 0) {
        ...
        if(!strcmp(request,"registernode")) {
            if(!search_node(base,op_str)) {
                base=insert_node(base,op_str);
                fprintf(stdout, ">> registernode - New connection with device %s.\n",op_str);
            } else {
                fprintf(stdout, ">> registernode - Device %s already in the list.\n",op_str);
            }
        } else if(!strcmp(request,"info")) {
            list_nodes(base);
        } else if(!strcmp(request,"to")) {
            fprintf(stdout, ">> to:\n");
            strcpy(destination,op_str);
            str_cut('=',destination,op_str);
            str_cut(' ',destination,aux);
            fprintf(stdout,"Destination: %s\n",destination);

            strcpy(source,op_str);
            str_cut('=',source,op_str);
            str_cut(' ',source,aux);
            strcpy(messange,op_str);
            fprintf(stdout,"Source: %s\n",source);
            fprintf(stdout,"Message: %s\n",messange);

            if((!strcmp(destination,"upperstack") || !strcmp(destination,"centralstack")) &&
                (!strcmp(source,"avago_gesture_sensor") || !strcmp(source,"pixart_gesture_sensor")
                || !strcmp(source,"ams_gesture_sensor") || !strcmp(source,"st_gesture_sensor"))) ) {
                if(!strcmp(messange,"<GESTURE_UP>")) {
                    fprintf(stdout, ">> <GESTURE UP DETECTED>\n");
                    fprintf(socOUT, "<G-UP - OK>\n");
                    Test_HMI_Function();
                }
                ...
                else if(!strcmp(messange, "<GESTURE_WAVE>")) {

```

```

        fprintf(stdout, ">> <GESTURE WAVE DETECTED>\n");
        fprintf(socOUT, "<G-WAVE - OK>\n");
        Test_HMI_Function();
    } else {
        fprintf(stdout, ">> <INVALID GESTURE DETECTED>\n");
        fprintf(socOUT, "<G-INV - KO>\n");
    }
}
}
...
}
...
}

```

LISTING 4.11: Client Threads Highlights

Looking to the extract of code above, it is possible to observe the main function of this thread, that is basically to decode the frames or messages from the clients to the server (in this cases the local systems(s)).

Basically 3 functionalities were implemented: the registration of a new node (local system), the request for information related to the registered nodes, and the decode of the messages/frames related to the detected gestures. In this case, for every detected gestures just a dummy HMI function is executed "Test_HMI_Function", but in a real application the functionalities should vary with each gesture and context of operation.

This approach of simulation was chosen, mainly because the front end GUI applications to use in the project are still in development, so the major interest was to test the system integration with a TCP/IP server.

4.1.3 GUI Application Application Interface

Main Thread

```

...
pthread_create(&thread1, NULL, dowork, NULL);
...
setupRealtimeData(ui->customPlot);
connect(&gestureTimer, SIGNAL(timeout()), this, SLOT(doGestureWork()));
gestureTimer.setSingleShot(false);
gestureTimer.start(1000); // Interval of 1s

connect(&TextTimer, SIGNAL(timeout()), this, SLOT(doGUIwork()));
TextTimer.setSingleShot(false);
TextTimer.start(0);

...
ui->label_8->setStyleSheet("QLabel { color : blue; }");
...
ui->pushButton_5->setStyleSheet("QPushButton { color : blue; }");
...

```

LISTING 4.12: Main Thread Highlights

Considering the extract of code above, it is possible to identify that this thread is responsible to create the support thread, configure 3 timer routines and proceed with the main GUI initializations.

In this case, the three timers are configured to run continuously and with different timing intervals. One of the timers is initialized in the "setupRealtimeData" routine, the "dataTimer", and the others are directly initialized in this routine.

The "dataTimer" is configured to send a timeout signal to call the callback "realTimeDataSlot", to update the data in the graphic. This timer is configured for a counting of 0 milliseconds, which in practice means that the callback is called when it

is possible.

The "gestureTimer" sends a timeout signal at each second and this signal is used to call the callback routine, responsible to update the data being shown relative to the execution of gestures (both text or gesture symbols).

The other timer, "TextTimer", should send a timeout signal to call the callback routine as soon as possible, because it is configured to generate a timeout for 0 milliseconds, but in practice the callback is called only when possible. This routine is mainly responsible for updating the data being displayed on the monitoring data area.

The extract of code below, shows the "setupRealtimeData" timer configuration. Like it is possible to see this timer is configured to run continuously and generate a timeout signal whenever possible. As stated before, this routine is used to update the data in the graphical area.

```
// make left and bottom axes transfer their ranges to right and top axes:
connect(customPlot->xAxis, SIGNAL(rangeChanged(QCPRange)), customPlot->xAxis2,
        SLOT(setRange(QCPRange)));
connect(customPlot->yAxis, SIGNAL(rangeChanged(QCPRange)), customPlot->yAxis2,
        SLOT(setRange(QCPRange)));

// setup a timer that repeatedly calls MainWindow::realtimeDataSlot:
connect(&dataTimer, SIGNAL(timeout()), this, SLOT(realTimeDataSlot()));
dataTimer.start(0); // Interval 0 means to refresh as fast as possible
```

LISTING 4.13: setupRealtimeData Routine - Timer and Configuration

Support Thread

```
...
while(1){
s.ReadData(data_serial);
if( strlen(data_serial)>0 && data_serial[0]!='#' && data_serial[0]!='$')
{
new_data=true;
if(!strcmp(data_serial,"UP\n")){
gesture_detected_type=1;
qDebug() << "Up Gesture detected";
}
...
else if(!strcmp(data_serial,"WAVE\n")){
gesture_detected_type=9;
qDebug() << "Wave Gesture detected";
} else {
qDebug() << data_serial;
}
}
else if( strlen(data_serial)>3 && data_serial[0]=='$')
{
...
if(data_serial[1]=='A') strcpy(ALS,&data_serial[2]);
else if(data_serial[1]=='R') strcpy(RED,&data_serial[2]);
...
} else {
if(strlen(data_serial)>=3)
{
qDebug()<<"Data received: " << data_serial <<endl;
if(data_serial[1]=='1') new_data=true, value0 = bytesToInt(&data_serial[2]);
else if(data_serial[1]=='2') new_data=true, value1 = bytesToInt(&data_serial[2]);
...
}
}
...
}
...
}
```

LISTING 4.14: Support Thread Highlights

Looking at the extract of code above, it is possible to observe that this routine is responsible for parsing the data received via serial interface, decoding the type of message received from the local system.

This thread is responsible for implementing the communication protocol, explained before in the system architecture design, identifying in this case frames of gestures, monitoring data like ambient light level, proximity and RGB color values, and photodiodes data. These data frames are then used in the plotting and monitoring sections of the application.

Support Routine 1

```

...
if(new_data==true) {
    ui->textEdit_2->setText(ui->textEdit_2->toPlainText() + data_serial);
    c.movePosition(QTextCursor::End);
    ui->textEdit_2->setTextCursor(c);
    data_serial[0]='\0';
    new_data=false;
}
...

```

LISTING 4.15: Support Routine Highlights

Analyzing the extract of code above, it is possible to observe that this routine is responsible to treat the data received via serial interface, inserting the received data in the main text log area, and updating the cursor in that same log area.

Support Routine 2

```

...
if(new_als_rgd==true)
{
    ...
    if(var_monitor==true)
    {
        ui->label_11->setText(ALS);
        ui->label_12->setText(RED);
    }
}
if(gesture_detected_type)
{
    switch(gesture_detected_type)
    {
        case 1:
            // UP GESTURE
            ui->label_6->setText("UP GESTURE DETECTED");
            ui->textEdit_3->setText(ui->textEdit_3->toPlainText()+"\nUP GESTURE
            DETECTED");
            c_commands.movePosition(QTextCursor::End);
            ui->textEdit_3->setTextCursor(c_commands);
            load_up_image();
            gesture_detected_type=0;
            state=true;
            break;
        ...
        case 9:
            // WAVE GESTURE
            ui->label_6->setText("WAVE GESTURE DETECTED");
            ui->textEdit_3->setText(ui->textEdit_3->toPlainText()+"\nWAVE GESTU
            RE DETECTED");
            c_commands.movePosition(QTextCursor::End);
            ui->textEdit_3->setTextCursor(c_commands);
            load_wave_image();
    }
}

```

```

        gesture_detected_type=0;
        state=true;
        break;
    ...
}
} else if(state==true)
{
    ui->label_6->setText("Waiting for gesture input ...");
    hide_up_image();
    ...
    hide_wave_image();
    ...
}
...
if( count_5s == 1 ) value_monitor_1=value0, value_monitor_2=value1, value_monitor
_3=value2, value_monitor_4=value3;
...

```

LISTING 4.16: Support Routine Highlights

Analyzing the extract of code above, it is possible to observe that this routine is responsible for parsing the data received via serial interface, inserting the received data in the monitoring text log areas, and updating the cursor in that log area. More than that, this routine also coordinates the animations of the gestures detected, by informing the user of the detected gestures (either by text or image), and after 5 seconds hides all the gesture detected information.

Support Routine 3

```

...
if(monitor_all==true) {
    add_value_type_1(key, value0);
    add_value_type_2(key, value1);
    add_value_type_3(key, value2);
    add_value_type_4(key, value3);
}
else{
    if(monitor_u==true) add_value_type_1(key, value0);
    if(monitor_d==true) add_value_type_2(key, value1);
    if(monitor_l==true) add_value_type_3(key, value2);
    if(monitor_r==true) add_value_type_4(key, value3);
}
...
// make key axis range scroll with the data (at a constant range size of 8):
ui->customPlot->yAxis->setRange(150,300,Qt::AlignBottom);
ui->customPlot->xAxis->setRange(key+0.25, 8, Qt::AlignRight);
ui->customPlot->replot();
...

```

LISTING 4.17: Support Routine Highlights

Observing the extract of code above, it is possible to note that this routine is responsible for processing and updating new data, previously received, in the graphical area.

This routine decodes the monitoring option in the graphic area, either by joining the photodiode readings to the graphic or the light and RGB color readings. Another responsibility of this routine is the resizing of the graphical area and the re-plot of the graphic.

4.2 Prototype Testing Platform

The prototype testing platform, first presented in 3.11, was implemented like was planned, like it is possible to see in the figure blow, Figure 4.2.

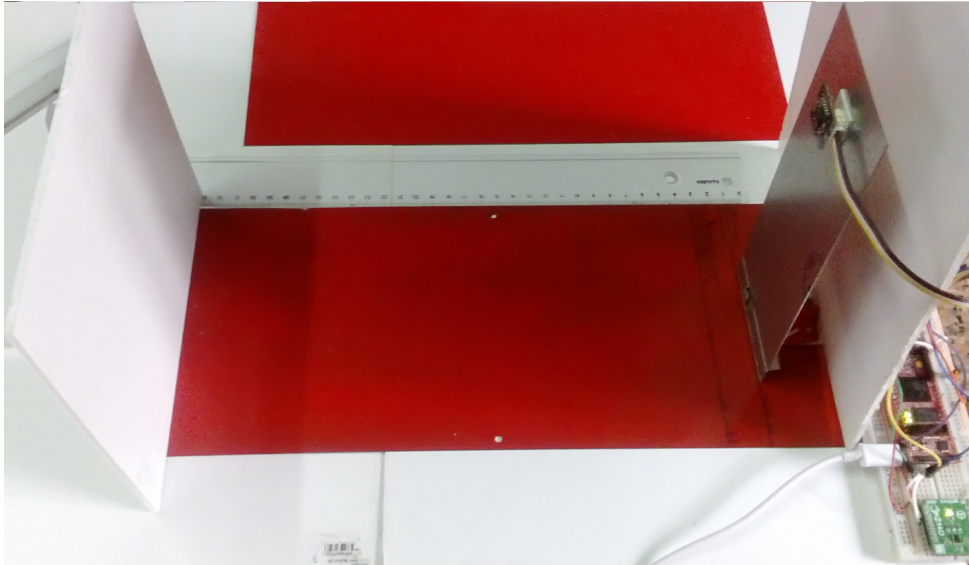


FIGURE 4.2: Prototype Testing Platform Mockup

Prototype Materials

For the construction of base frame and supports, the materials used were galvanized steel sheets. For the obstacle/object used for proximity measurements, the material used was a 5mm thick K-line sheet.

K-line material, which consists in a two layer of bright white paper separated by a high density polyethylene foam core, is characterized by a high light reflectance, due to the white color and material in question. Therefore, this material was selected as an option to proceed with the initial proximity and gesture tests.

Measurement Procedures

To trace each sensors response, in terms of proximity measurements, the procedure taken was the collection of proximity measurements for each half centimeter, giving that way two samples for each centimeter.

Since the sensors have different measurement ranges, the maximum range measured varied with each sensor, from 15 cm to 40 centimeters.

Considering the different measurement ranges, the effective number of data samples varied from 30 to 80 samples.

Chapter 5

Validation and Results

In this chapter, all the experimental results obtained after the implementation and integration phase of the system are described.

Initially the tests will be based on the analysis of the sensing devices and the testing of the different modules of the system. In this step, it is necessary to test if the individual elements are working and the system functioning and behavior is acceptable. Then if the tests are successful, the integration of the system can be achieved.

In this integration tests, the system viability is analyzed to check if the system ensures the functionalities set by the requirements.

After the integration tests, and since the system will not be integrated in a real driving simulator, a battery of final in-lab tests will be designed and implemented. The realization of this tests is extremely important to analyze the system performance and reliability.

5.1 Calibration Tests

The first set of tests to conduct is based on the testing of the different sensing devices response and basic behavior. These tests are a important opportunity to collect information to be used in the development of the gesture detection system, since it can be used to define the basic configurations for the sensing devices.

5.1.1 Avago APDS-9960

The Avago APDS-9960 allows a great level of customization and calibration options, both for proximity and gesture detection. In the calibration of the sensor several variables were taken such as the LED drive current, light pulse wavelength, light pulse number, proximity gain, LED boost current, wait time (non gesture timing), proximity and gesture detection thresholds.

Calibration Tests

After a first battery of tests several choices were taken for further study. To maximize the range of detection the IR led drive current was set to 100 mA and the led boost current to 300%, making the detection range from nearly 15cm up to 40 cm (however less than 30 cm is considered optimal). Besides this configuration others were selected, for example the wait time between readings, set to the minimum of 2.78 ms (to maximize the data acquisition).

During the main calibration steps, settings such as the light pulse wavelength, light pulse number and proximity gain were also tested.

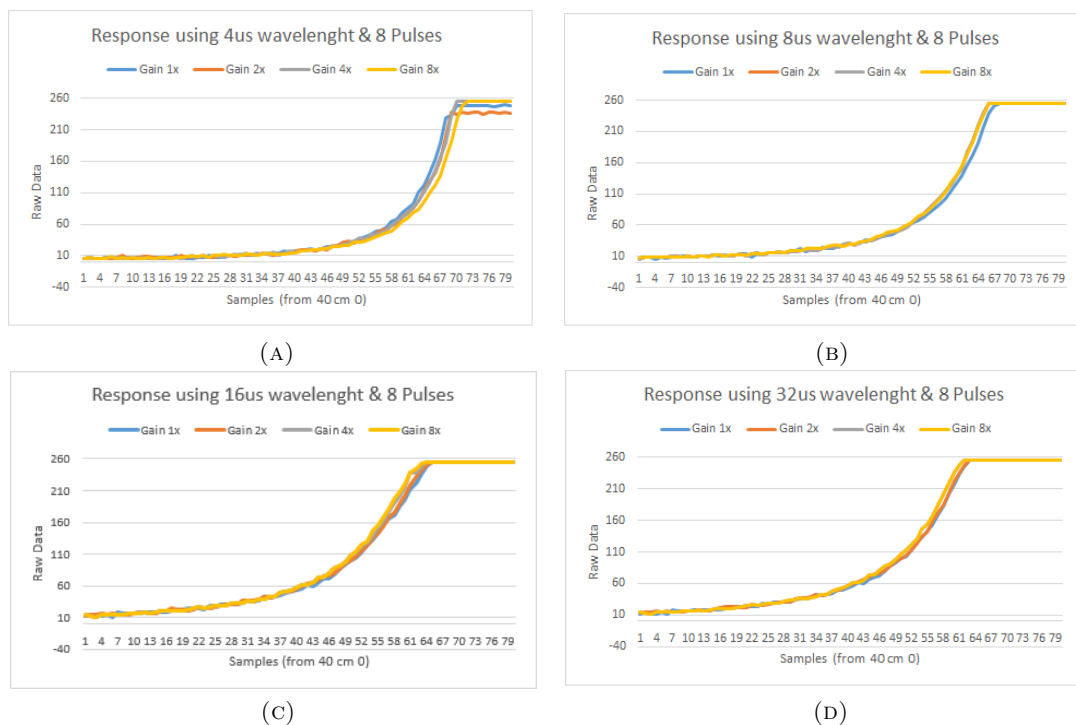


FIGURE 5.1: Functional test using as fixed parameter the number of pulses per measurement and as variable the pulse wavelengths

These tests, as displayed in Figure 5.1, are presented as means to observe the sensing device response. After the tests, the final number of pulses was set to 8 (available from 1 to 64), and the final light pulse wavelength was set to $8\mu\text{s}$. Like it is possible to show for this configuration, the sensor output saturation occurs approximately at 7 to 7.5cm. Since the distance of detection between 0 to 20cm was an expected result, this configuration was selected.

Another configuration selected was the proximity gain, with a value of 4x, allowing that way a superior linearity compared with 1x, 2x and 8x.

5.1.2 ams TMG4903

The ams TMG4903 is a sensing device with a high level of customization and calibration options for proximity, ambient light, color and gesture detection.

In the calibration phase of this sensor several variables were taken in account, such as the IR led drive current, light pulse wavelength, light pulse number, proximity gain, detection wait time (non gesture timing), proximity and gesture detection thresholds.

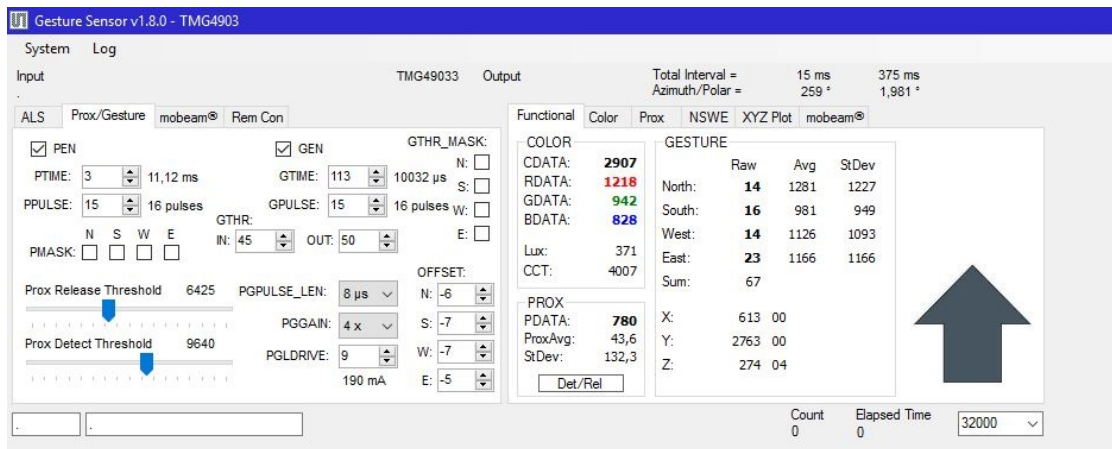
Calibration Tests

Since the sensing device comes in a hardware development kit and also has a dedicated testing GUI application, all the initial tests were done using that application.

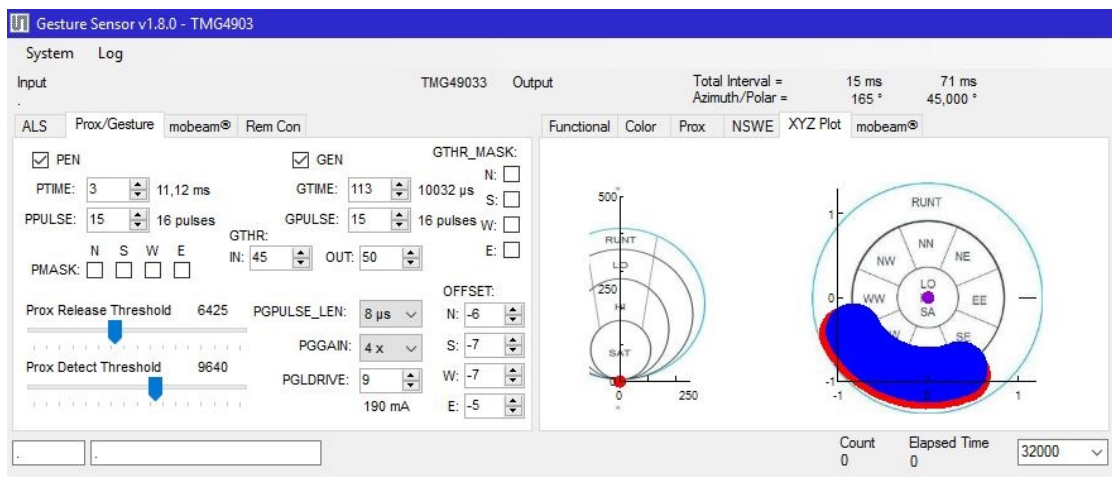
After a first set of tests several choices were taken for further work. To maximize the range of detection the LED drive current was set to 190 mA, making the detection range from nearly 10cm up to 25 cm (less than 15 cm as optimal working range). Besides this configuration others were selected, for example, the wait time set to the minimum of

2.78 ms (to maximize the data acquisition).

During the main calibration steps, settings such as the light pulse wavelength, light pulse number and proximity gain were also tested.



(A)



(B)

FIGURE 5.2: Functional tests using ams GUI application for TMG4903 sensing device.

This test, described in Figure 5.2, is shown as sample of the GUI application working procedure. Based on the application setting like IR led drive current, light pulse wavelength, light pulse number, proximity gain, detection wait time, proximity thresholds were configured, tested and later defined as inputs for the development of the sensing device interface code, gesture detection algorithms and for the final calibrations.

5.1.3 PixArt PAJ7620U2

The PixArt PAJ7620U2 is a sensing device more limited in terms of calibration and optimization settings, mainly because it is a closed hardware solution, almost seen like a black box in terms of working process. To proceed with the device calibration, the available configurations options were tested.

Besides the reduced number of settings capable to be tuned, settings like proximity gain, proximity high/low hysteresis thresholds, and working mode (normal and gaming)

were especially tested.

The next figure, Figure 5.3, shows part of the proximity gain testing.

Calibration Tests

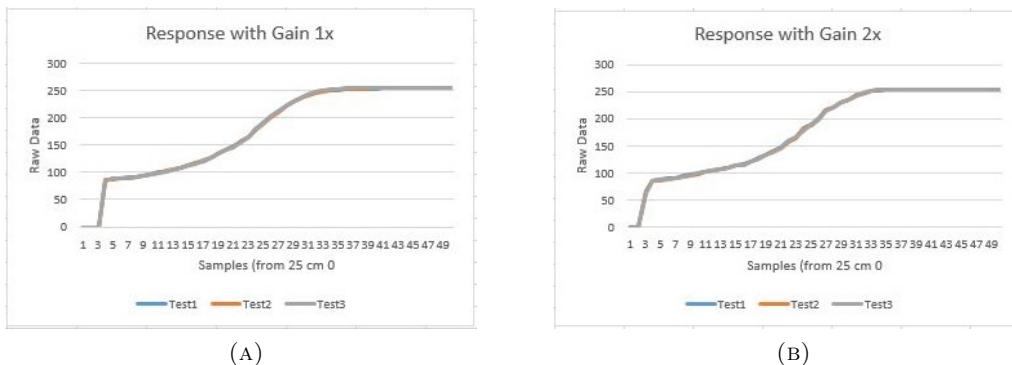


FIGURE 5.3: Battery of 3 tests for 1x or 2x proximitygains

Considering the handicap of this device in terms of customization and parametrization, especially in the configuration of parameters related to the detection of the object brightness and object size, in the gesture detection after the tests were done several choices were taken.

The gain was set to 2, the high/low hysteresis thresholds were maintained as default and the working mode set to game mode, allowing 60 °/s up to 1200 °/s of detection capability.

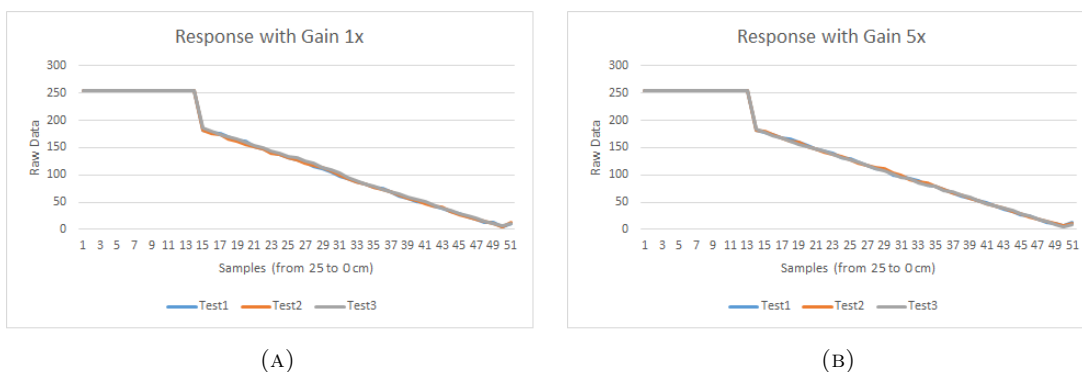
5.1.4 ST VL6180X

Since the ST VL6180X allows a high level of customization and calibration settings, mainly for the detection of proximity and ambient light, a set of tests should be made to collect data from the sensor response under different configurations.

In the calibration of the sensor several variables were taken into account such as the proximity thresholds, the integration period, the crosstalk compensation rate, the inter-measurement period and others.

The following tests are a sample of the realized tests, in this case, only testing the proximity measurement for various proximity gains.

Calibration Tests



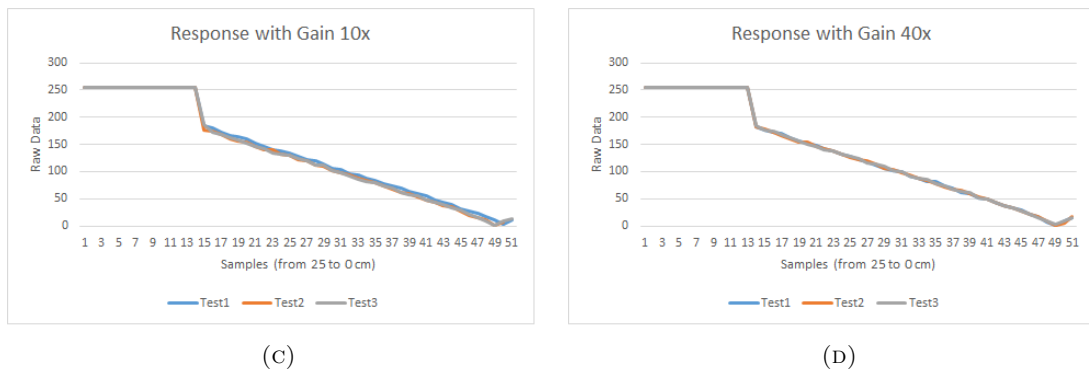


FIGURE 5.4: Sensor response with proximity gain 1x (A), 5x (B), 10x (C) and 40x (D)

After a first set of tests several choices were taken for further study, like the proximity gain set to 5x, or the inter-measurement time set to 10 milliseconds.

These choices made the practical measurement range up to 20 cm (up to 10 cm with very reliable detection and up to 18 cm with still reliable results, but more dependent of factors like temperature).

5.2 Initial Tests

Before starting the test of the system, it is necessary to analyze the behavior of each unit of the system. As stated before, the system is mainly formed by a local system and remote server, furthermore a GUI application is used for monitoring.

The testing of the local system is, in this case, divided in the testing of each sensing device, where a identical set of tests is conducted in every sensor.

The testing of the server is conducted following a single client approach and multi-client approach, where the communication protocol is tested.

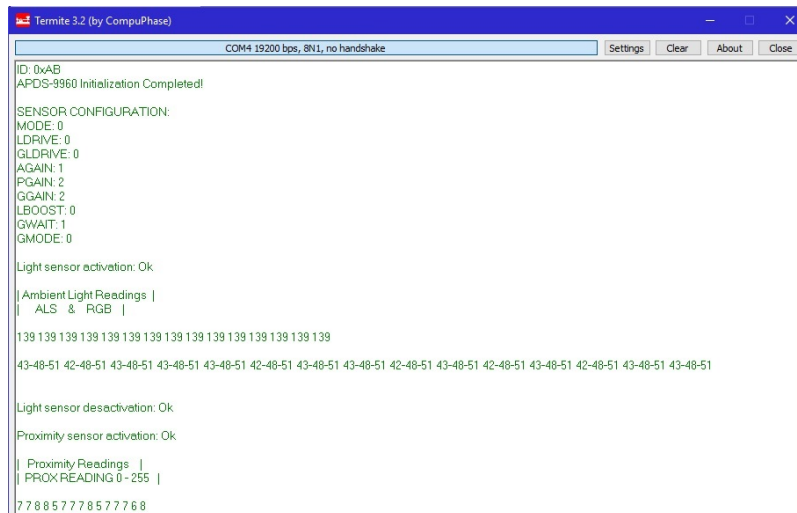
The GUI application is tested, both in terms of communication protocol and graphical response, using in this case as connected sensing device, the Avago APDS 9960.

5.2.1 Local System

The test of the local system components is achieved by following a modular strategy, testing each sensing devices and the development board used.

Avago APDS-9960

Test: To test the API developed of this sensor, a initial test consists in configuring and collecting data from the sensing device. This data is then used by the interface board, and sent to the monitoring laptop pc, where is used for monitoring.



```

Termit 3.2 (by CompuPhase)
COM4 19200 bps, 8N1, no handshake
Settings Clear About Close

ID: 0xAB
APDS-9960 Initialization Completed!

SENSOR CONFIGURATION:
MODE: 0
LDRIVE: 0
GLDRIVE: 0
AGAIN: 1
PGAIN: 2
GGAIN: 2
LBOOST: 0
GWAIT: 1
GMODE: 0

Light sensor activation: Ok

| Ambient Light Readings |
| ALS & RGB |
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
43-48-51 42-48-51 43-48-51 43-48-51 43-48-51 42-48-51 42-48-51 43-48-51 43-48-51 42-48-51 42-48-51 43-48-51 43-48-51

Light sensor deactivation: Ok

Proximity sensor activation: Ok

| Proximity Readings |
| PROX.READING 0 - 255 |
77 88 57 77 85 77 78 57 77 68

```

FIGURE 5.5: Avago APDS-9960 initial interface test.

Result: The test was successful, because the sensing device was well configured and the ambient light, rgb color and proximity values were successfully collected.

- ams TMG4903

Test: To test the API of this sensor, a initial test consists in configuring and collecting data from the sensing device. This data is then processed by the interface board, and sent to the monitoring laptop pc, where is used for monitoring.



```

Termit 3.2 (by CompuPhase)
COM9 19200 bps, 8N1, no handshake
Settings Clear About Close

I2C device found at address 0x72
TMG ID: 0xB8
TMG REV ID: 0x02
ENABLE OK
TIME CONF OK
CONFIG REGS OK
GESTURE REGS OK
OFFSET CONFIG OK
CALIBRATION CONFIG OK
GEST CONFIG OK

TMG4903 Initialization Completed!

SENSOR CONFIGURATION:
MODE: 0x00
LDRIVE: 0x09
GLDRIVE: 0x09
AGAIN: 0x02
PGAIN: 0x02
GGAIN: 0x02
GWAIT: 0x00
GMODE: 0x00

Light sensor activation: Ok

| Ambient Light Readings |
| ALS & RGB |
611 610 610 610 610 610 610 610 610 610 610 609 609 609
181-232-198 181-232-198 181-232-198 181-232-198 181-232-198 181-232-198 181-232-198
181-232-198 181-232-198 181-232-198 181-232-198 181-232-198 181-232-198 181-232-198
181-232-198 181-232-198 181-232-198 181-232-198 181-232-198 181-232-198

Light sensor deactivation: Ok

Proximity sensor activation: Ok

| Proximity Readings |
| Prox = SensorRead |
65313 38 41 38 38 35 44 35 42 35 41 31 32 35 37

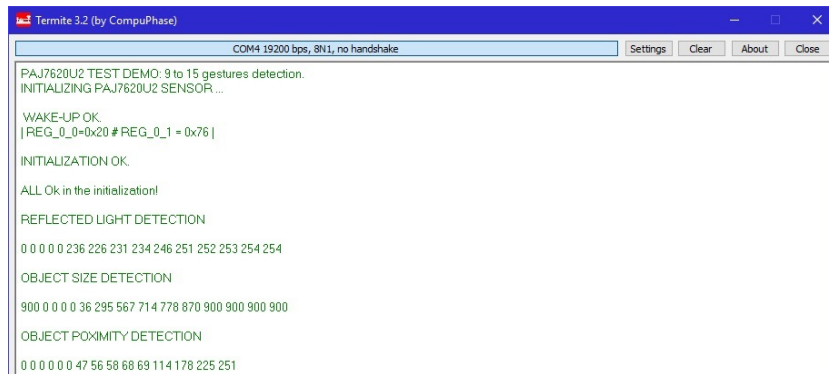
```

FIGURE 5.6: ams TMG4903 initial interface test.

Result: The test was successful, because the sensing device was well configured and the ambient light, rgm color and proximity values were successfully collected.

• PixArt PAJ7620U2

Test: To test the API of this sensor, a initial test consists in configuring and collecting data from the sensing device. This data is then used by the interface board, and sent to the monitoring laptop pc, where is used for monitoring.



```

Termite 3.2 (by CompuPhase)
COM4 19200 bps, 8N1, no handshake
PAJ7620U2 TEST DEMO: 9 to 15 gestures detection.
INITIALIZING PAJ7620U2 SENSOR ...

WAKE-UP OK
|REG_0_0=0x20#REG_0_1 = 0x76|

INITIALIZATION OK.

ALL Ok in the initialization!

REFLECTED LIGHT DETECTION
0 0 0 0 0 236 226 231 234 246 251 252 253 254 254

OBJECT SIZE DETECTION
900 0 0 0 0 36 295 567 714 778 870 900 900 900 900

OBJECT PROXIMITY DETECTION
0 0 0 0 0 47 56 58 68 69 114 178 225 251

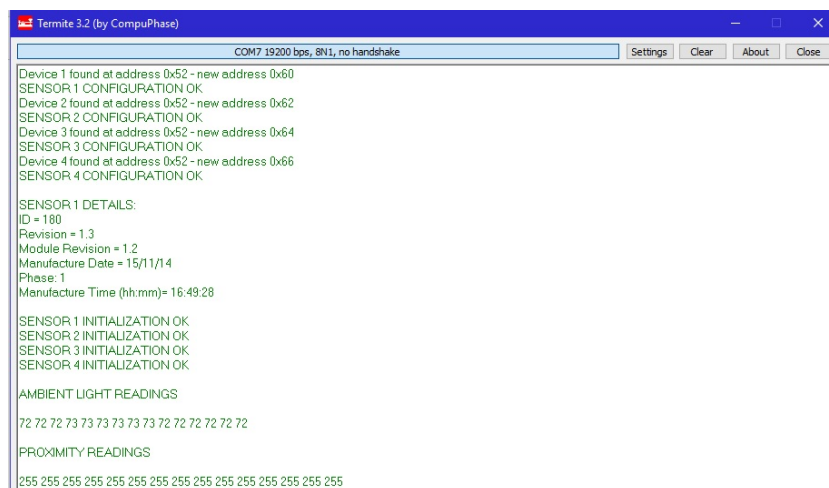
```

FIGURE 5.7: PixArt PAJ7620U2 initial interface test.

Result: The test was successful, because the sensing device was well configured and the object reflected light, object size and object proximity values were successfully collected.

• ST VL6180X

Test: To test the API of this sensor, a initial test consists in configuring and collecting data from the sensing device. This data is then used by the interface board, and sent to the monitoring laptop pc, where is used for monitoring.



```

Termite 3.2 (by CompuPhase)
COM7 19200 bps, 8N1, no handshake
Device 1 found at address 0x52 - new address 0x60
SENSOR 1 CONFIGURATION OK
Device 2 found at address 0x52 - new address 0x62
SENSOR 2 CONFIGURATION OK
Device 3 found at address 0x52 - new address 0x64
SENSOR 3 CONFIGURATION OK
Device 4 found at address 0x52 - new address 0x66
SENSOR 4 CONFIGURATION OK

SENSOR 1 DETAILS:
ID = 180
Revision = 1.3
Module Revision = 1.2
Manufacture Date = 15/11/14
Phase: 1
Manufacture Time (hh:mm)= 16:49:28

SENSOR 1 INITIALIZATION OK
SENSOR 2 INITIALIZATION OK
SENSOR 3 INITIALIZATION OK
SENSOR 4 INITIALIZATION OK

AMBIENT LIGHT READINGS
72 72 72 73 73 73 73 73 72 72 72 72 72

PROXIMITY READINGS
255 255 255 255 255 255 255 255 255 255 255 255 255 255

```

FIGURE 5.8: ST VL6180X initial interface test.

Result: The test was successful, because the sensing devices were well configured and the ambient light and proximity values were successfully collected.

5.2.2 Remote Server

The test of the remote server is achieved by following a modular strategy, testing first the response of the sever in terms of functionality using one single client and then doing a multi-client test, testing the multi-thread response.

Protocol Test: To test the implementation of the server, responsible for the treatment of the gesture data frames received from the local system, a first test consisted in testing the communication protocol, not using the local system as client, but using a localhost client.

<pre>>> to: Destination: upperstack Source: avago_gesture_sensor Message: <GESTURE_UP> >> <GESTURE UP DETECTED> >> to: Destination: upperstack Source: avago_gesture_sensor Message: <GESTURE_DOWN> >> <GESTURE DOWN DETECTED> >> to: Destination: upperstack Source: avago_gesture_sensor Message: <GESTURE_LEFT> >> <GESTURE LEFT DETECTED> >> to: Destination: upperstack Source: avago_gesture_sensor Message: <GESTURE_RIGHT> >> <GESTURE RIGHT DETECTED> >> to: Destination: upperstack Source: avago_gesture_sensor Message: <GESTURE_IN> >> <GESTURE IN DETECTED> >> to: Destination: upperstack Source: avago_gesture_sensor Message: <GESTURE_OUT> >> <GESTURE OUT DETECTED> >> to: Destination: upperstack Source: avago_gesture_sensor Message: <GESTURE_CW_CIRCLE> >> <GESTURE CW CIRCLE DETECTED> ●●●</pre>	<pre>to=upperstack from=avago_gesture_sensor message=<GESTURE_UP> >> <GESTURE UP DETECTED> to=upperstack from=avago_gesture_sensor message=<GESTURE_DOWN> >> <GESTURE DOWN DETECTED> to=upperstack from=avago_gesture_sensor message=<GESTURE_LEFT> >> <GESTURE LEFT DETECTED> to=upperstack from=avago_gesture_sensor message=<GESTURE_RIGHT> >> <GESTURE RIGHT DETECTED> to=upperstack from=avago_gesture_sensor message=<GESTURE_IN> >> <GESTURE IN DETECTED> to=upperstack from=avago_gesture_sensor message=<GESTURE_OUT> >> <GESTURE OUT DETECTED> to=upperstack from=avago_gesture_sensor message=<GESTURE_CW_CIRCLE> >> <GESTURE CW CIRCLE DETECTED> to=upperstack from=avago_gesture_sensor message=<GESTURE_CCW_CIRCLE> >> <GESTURE CCW CIRCLE DETECTED> to=centralstack from=st_gesture_sensor message=<GESTURE_OUT> >> <GESTURE OUT DETECTED> to=centralstack from=st_gesture_sensor message=<GESTURE_LEFT_RIGHT> >> <GESTURE LEFT-RIGHT DETECTED> to=centralstack from=st_gesture_sensor message=<GESTURE_LEFT_RIGHT> >> <GESTURE LEFT-RIGHT DETECTED></pre>
(A)	(B)

FIGURE 5.9: Server(A) - Client(B) communication protocol test.

Result: The test was successful, because the server recognized the gesture data frames/messages, validating the decoding process. Also to note that the correct server responses were sent to the client like it was expected.

Multi-Client Test: To test the implementation of the server, responsible for the treatment of the gesture data frames received from the local system, a second test consisted in testing the communication protocol to more than one client. For demonstration purposes 3 clients were used int the test.

```

Server is running
Client Connected 4
Client Connected 5
Client Connected 6
>> registernode - New connection with device avago_gesture_sensor
>> registernode - New connection with device st_gesture_sensor
>> registernode - New connection with device pixart_gesture_sensor
>> to:
Destination: upperstack
Source: avago_gesture_sensor
Message: <GESTURE_UP>
>> <GESTURE UP DETECTED>
>> to:
Destination: upperstack
Source: st_gesture_sensor
Message: <GESTURE_UP>
>> <GESTURE UP DETECTED>
>> to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: <GESTURE_UP>
>> <GESTURE UP DETECTED>

```

```

artfaria@artfaria-System-Product-Name: ~
artfaria@artfaria-System-Product-Name:~$ telnet localhost 5000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^J'.
registernode=avago_gesture_sensor
to=upperstack from=avago_gesture_sensor message=<GESTURE_UP>
>> <GESTURE UP DETECTED>

```

```

artfaria@artfaria-System-Product-Name: ~
artfaria@artfaria-System-Product-Name:~$ telnet localhost 5000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^J'.
registernode=st_gesture_sensor
to=upperstack from=st_gesture_sensor message=<GESTURE_UP>
>> <GESTURE UP DETECTED>

```

```

artfaria@artfaria-System-Product-Name: ~
artfaria@artfaria-System-Product-Name:~$ telnet localhost 5000
Trying 127.0.0.1...
connected to localhost.
Escape character is '^J'.
registernode=pixart_gesture_sensor
to=upperstack from=pixart_gesture_sensor message=<GESTURE_UP>
>> <GESTURE UP DETECTED>

```

FIGURE 5.10: Server(A) - Clients(B) communication test.

Result: The test was successful, because the server recognized the gesture data frames/messages, validating the decoding process. Also to note that the correct server responses were sent to the 3 clients like it was expected.

5.2.3 GUI Application

The test of the GUI Application is achieved following a modular strategy, testing first the response of the application in terms of functionality (by testing the communication protocol), then a graphical test is done to test the graphical plotting and the gesture recognition feedback.

Protocol Test: To test the implementation of the GUI application and the dedicated communication protocol, responsible for the treatment of the gesture data frames received from the local system, a test was created to check the data being received.

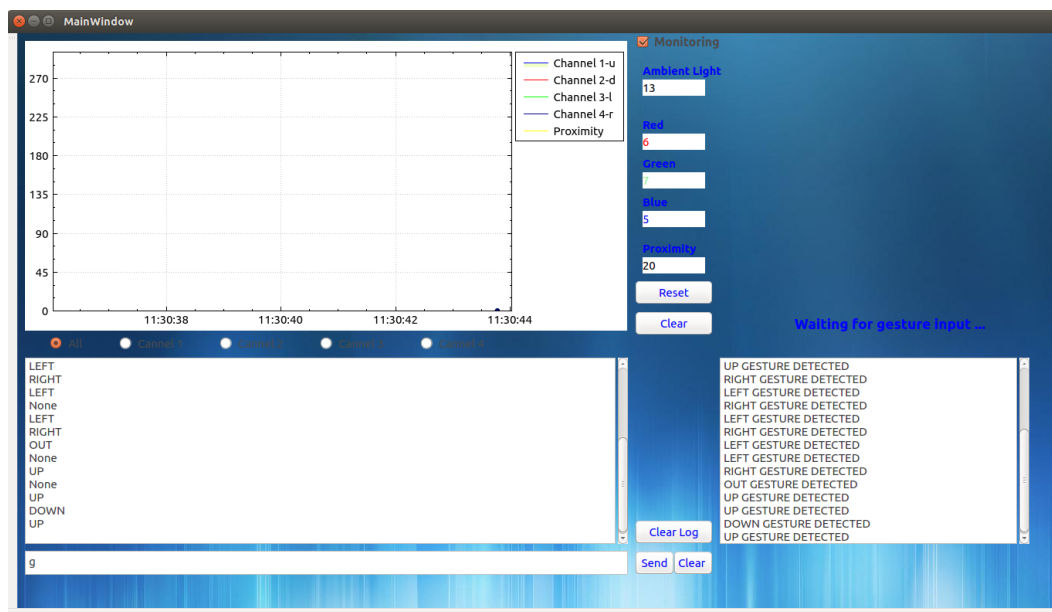


FIGURE 5.11: GUI application communication test.

Result: The test was successful, because the GUI application recognized the data frames/messages correctly, validating that way the parsing and decoding processes. It is possible to see the data being monitored, the gesture detection, and general serial port logs.

- **Plotting Test:** Also to test the implementation of the GUI application, a test was designed to test the sensing device response in the graphic plot area. The test covers the decoding of the gesture data as well as the plotting capability of the application.

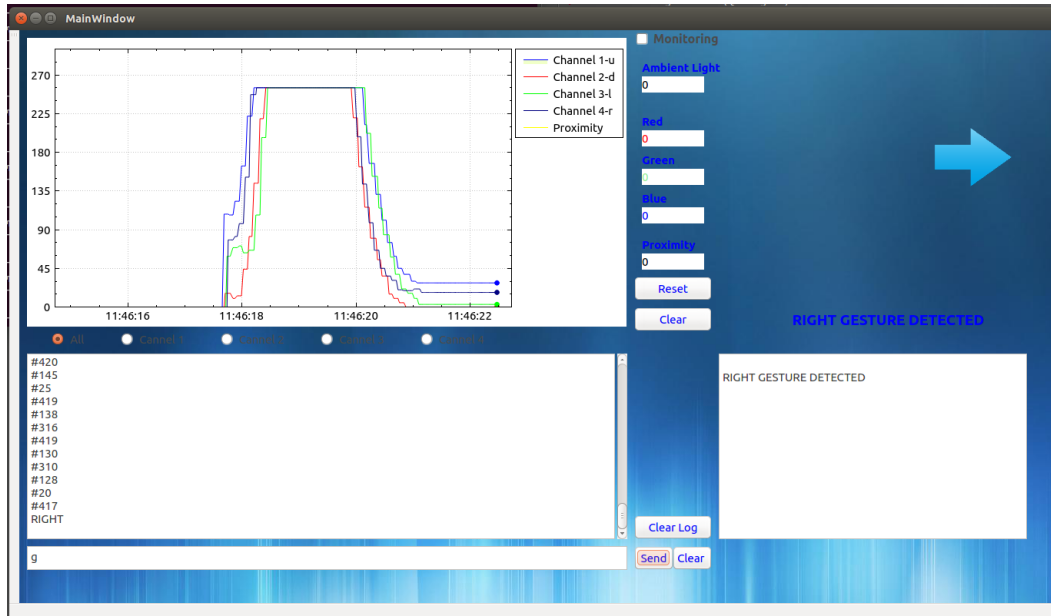


FIGURE 5.12: GUI application plotting test.

Result: The test was successful, because the GUI application recognized the data frames/messages correctly, and at the same time the data plotting was successful. It is possible to see that the plot is not very smooth, a fact that is explained by the nature of the gesture and the sensing device response (sensing device used was the AVAGO APDS-9960). Also in this test, the gesture feedback is correctly done by the right blue arrow, that identifies a right gesture detected.

5.3 Integration Tests

After the first set of tests, testing the individual working order of each system main components, further tests were necessary. The next tests conducted were the integration tests, testing the working order of the local system as well as its integration with the remote cluster that runs the remote server.

5.3.1 Avago APDS-9960

Test: To test the integration of the local system, using the AVAGO APDS-9960 sensing device, with the cluster running the dedicated server, a test was developed to validate the detection of gestures and the communication between both elements.

(A)

(B)

FIGURE 5.13: AVAGO APDS-9960 integration test.

Result: Looking to Figure 5.13, it is possible to see that the test was successful, because the local system recognized the gestures executed and the communication to the server also worked, will all the data frames being well transferred and decoded.

5.3.2 ams TMG4903

Test: To test the integration of the local system, using the ams TMG4903 sensing device, with the cluster running the dedicated server, a test was conceived to validate the detection of gestures and the communication between both elements.

(A)

(B)

FIGURE 5.14: ams TMG4903 integration test.

Result: Analyzing the Figure 5.14, it is possible to see that the test was successful, because the local system recognized the gestures executed and the communication to the server also worked, will all the data frames being well transferred and decoded.

5.3.3 PixArt PAJ7620U2

Test: To test the integration of the local system, using the PixArt PAJ7620U2 sensing device, with the cluster running the dedicated server, a test was conceived to validate the detection of gestures and the communication between both elements.

(A)

```

anifa@anifa-System-Product-Name:~/NetBeansProjects/server
~/farlab/rfariis-System-Product-Name:~/netbeansProjects/server$ ./server
Client Connected 4
>> registernode - Device already in the list
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_LEFT>"
>> <GESTURE LEFT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_UP_DOWN>"
>> <GESTURE UP-DOWN DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_DOWN_UP>"
>> <GESTURE DOWN-UP DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_LEFT_RIGHT>"
>> <GESTURE LEFT-RIGHT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_RIGHT_LEFT>"
>> <GESTURE RIGHT-LEFT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_OUT>"
>> <GESTURE OUT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_IN_OUT>"
>> <GESTURE IN-OUT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_IN>"
>> <GESTURE IN DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_OUT>"
>> <GESTURE OUT DETECTED>

```

(B)

```

Termité 3.2 (by CompuPhase)
COM4 19200 bps, 8N1, no handshake
PAJ7620U2 TEST DEMO - 9:15 GESTURES
INITIALIZING PAJ7620U2 SENSOR ...
WAKE-UP OK
I#REQ_0:0x000 - #REQ_0:1 - 0x76
INITIALIZATION OK
LPC4088 MAC address is 00:1e:c0:09:16:55
LPC4088 IP address is 192.168.1.3
Network Gateway is 192.168.1.1
Network Mask is 255.255.0.0
Socket connected to (192.168.1.4) on port (9999)
MESSAGE SENT: registermode-pixart_gesture_sensor
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_LEFT>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_UP_DOWN>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_LEFT_RIGHT>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_RIGHT_LEFT>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_OUT>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_IN_OUT>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_IN>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_RIGHT>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_UP>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_DOWN>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_IN>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_RIGHT>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_WAVE>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_CW_CIRCLE>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_RIGHT>"
MESSAGE SENT: to-upperstack from-pixart_gesture_sensor message="<GESTURE_CW_CIRCLE>"

```

FIGURE 5.15: PixArt PAJ7620U2 integration test.

Result: Analyzing the Figure 5.15, it is possible to see that the test was successful, because the local system recognized the gestures executed and the communication to the server also worked, will all the data frames being well transferred and decoded.

5.3.4 ST VL6180X

Test: To test the integration of the local system, using the ST VL6180X sensing devices, with the cluster running the dedicated server, a test was conceived to validate the detection of gestures and the communication between both elements.

(A)

```

anifa@anifa-System-Product-Name:~/NetBeansProjects/server
~/farlab/rfariis-System-Product-Name:~/netbeansProjects/server$ ./server
Client Connected 4
>> registernode - Device already in the list
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_LEFT>"
>> <GESTURE LEFT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_UP_DOWN>"
>> <GESTURE UP-DOWN DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_DOWN_UP>"
>> <GESTURE DOWN-UP DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_LEFT_RIGHT>"
>> <GESTURE LEFT-RIGHT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_RIGHT_LEFT>"
>> <GESTURE RIGHT-LEFT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_OUT>"
>> <GESTURE OUT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_IN_OUT>"
>> <GESTURE IN-OUT DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_IN>"
>> <GESTURE IN DETECTED>
to:
Destination: upperstack
Source: pixart_gesture_sensor
Message: "<GESTURE_OUT>"
>> <GESTURE OUT DETECTED>

```

(B)

```

Termité 3.2 (by CompuPhase)
COM7 19200 bps, 8N1, no handshake
Device 1 found at address 0x52 - new address 0x60
Device 2 found at address 0x52 - new address 0x62
Device 3 found at address 0x52 - new address 0x64
Device 4 found at address 0x52 - new address 0x66
Found at address 0x60
Found at Address 0x62
Found at Address 0x64
Found at Address 0x66
LPC4088 MAC address is 00:1e:c0:08:84:4f
LPC4088 IP address is 192.168.1.5
Network Gateway is 192.168.1.1
Network Mask is 255.255.0.0
Socket connected to (192.168.1.4) on port (9999)
MESSAGE SENT: registermode-st_gesture_sensor
MESSAGE SENT: to-upperstack from-st_gesture_sensor message="<GESTURE_LEFT>"
MESSAGE SENT: to-upperstack from-st_gesture_sensor message="<GESTURE_RIGHT>"
MESSAGE SENT: to-upperstack from-st_gesture_sensor message="<GESTURE_UP>"
MESSAGE SENT: to-upperstack from-st_gesture_sensor message="<GESTURE_LEFT_RIGHT>"
MESSAGE SENT: to-upperstack from-st_gesture_sensor message="<GESTURE_DOWN>"
MESSAGE SENT: to-upperstack from-st_gesture_sensor message="<GESTURE_RIGHT_LEFT>"
MESSAGE SENT: to-upperstack from-st_gesture_sensor message="<GESTURE_UP_DOWN>"
MESSAGE SENT: to-upperstack from-st_gesture_sensor message="<GESTURE_DOWN_UP>"

```

FIGURE 5.16: ST VL6180X integration test.

Result: Analyzing the Figure 5.16, it is possible to see that the test was successful, because the local system recognized the gestures executed and the communication to the server worked, will all the data frames being well transferred and decoded.

5.4 Final Tests

After the initial tests and the integration tests, for the final testing procedures, a set of tests was conducted to validate the gesture recognition functionality. These tests were conducted using in-lab conditions, with controlled conditions, such as temperature, humidity and light exposure. This option of testing was chosen mainly for the unavailability of a driving simulator mockup.

5.4.1 Test Settings

For the final set of tests, a group of 10-30 people was planned to be used, being equally divided by genre, but with a diverse age distribution.

For the real set of final tests developed in this thesis, 12 people were initially invited (9 male and 3 female subjects), but only 8 have done all the testing scenarios (6 male and 2 female). Another fact is that the age of the subjects was similar, with an interval from 23 to 31 years old. Two conditions maintained were the number of gesture samples captured for each gesture type, with 50 gesture samples by each gesture type and subject, and the different hand positions, distances and gesture speeds (this last requirement was not fully granted in all the individuals).

These facts were not the expected case, but the availability of subjects of study was limited to proceed with the previously designed testing conditions.

The tests were made inside laboratory, with a temperature interval from 18-24 °C and with an ambient light level/illuminance of approximately 211 lux (in the light level interval of a dark to sunny day [102, 103]).

The subjects of study passed through a learning period, adjusted to each subject, where the system was introduced, including the gestures capable to be detected, the sensing device position and orientation.

5.4.2 Test Results

Avago APDS-9960

Test\Gesture	Up	Down	Right	Left	In	Out
Best	88	86	94	90	72	68
Worst	70	74	78	76	52	44
Average	79.33	82.83	85.67	84.17	61.67	56.17

TABLE 5.1: Avago APDS-9960 battery of tests

The final set of tests, that is compressed in Table 5.1, show the performance of the Avago APDS-9960 sensing device and the implemented algorithms. It is possible to see that the In and Out gestures have a rate of recognition lower than 70% (a minimum established reference value), namely 61.67% and 56.17%. This mainly happens due to the algorithm design and calibration, but also because of the user gesture timings and hand positions.

In the case of the vertical swipe gestures (Up and Down), the rate of recognition was lower than the horizontal gestures (Right and Left), mostly because the hand positions of the users were especially varied in the vertical swipe gestures. Even with this variation, the results were acceptable, with a range of 79.33% to 82.83% in vertical swipes and 84.17% to 85.67% in horizontal swipes.

ams TMG4903

Test \ Gesture	Up	Down	Right	Left	In	Out
Best	82	90	94	90	68	64
Worst	64	68	68	72	48	42
Average	70.83	73.17	79.67	78.17	56.92	49.67

TABLE 5.2: ams TMG4903 battery of tests

The final set of tests using the ams TMG4903 sensing device, depicted in Table 5.2, show the performance of the implemented algorithms.

It is possible to observe that the In and Out gestures have a rate of recognition also lower than 70%, namely 56.92% and 49.67%. This happens mainly due to the algorithm design and calibration, but also because of the user hand positions and gesture timings.

Considering the vertical swipe gestures (Up and Down), the rate of recognition was lower than the horizontal gestures (Right and Left), due to the users hand positions. Despite this difference, the results were satisfactory, with a range of 70.83% to 73.17% in vertical swipe gestures and 78.17% to 79.67% in horizontal swipes.

PixArt PAJ7620U2

Test \ Gesture	Up	Down	Right	Left	In	Out	CW Circle	CCW Circle	Wave
Best	92	94	98	96	78	76	92	90	86
Worst	70	74	78	80	46	32	62	66	58
Average	81.33	82.17	87.33	85.67	57.33	50.17	74.17	78.67	71.92

Test \ Gesture	Up-Down	Down-Up	Right-Left	Left-Right	In-Out	Out-In
Best	88	84	88	90	64	48
Worst	66	68	74	78	46	26
Average	77.33	75.17	81.33	82.92	51.50	38.67

TABLE 5.3: PixArt PAJ7620U2 Battery of tests results

The final set of tests, that is resumed in Table 5.3, using the PixArt PAJ7620U2 sensing device, show the performance of the implemented algorithms.

Simple Gestures

It is possible to observe that the In and Out gestures have a rate of recognition also lower than 70%, namely 57.33% and 50.17%. The main reason for these results is due to the algorithm design, but also because of the user gesture timings and hand positions.

Considering the vertical swipe gestures (Up and Down), the rate of recognition was lower than the horizontal gestures (Right and Left), mainly because the users hand positions. Despite the difference, the results were not so distant, with a range of 85.67% to 87.33% in horizontal gestures and 81.33% to 82.17% in vertical gestures.

Considering the clockwise and anti-clockwise circle gesture, the rate of recognition was above 70% (74.17% and 78.67%), which was desired. Another fact is that the wave gesture also register a rate of recognition above 70% (71.92%).

Sequential Gestures

In the case of sequential gestures, it is possible to observe that the vertical sequential swipe gestures had a performance lower than horizontal gestures (77.33% and 75.17% vs. 81.33% and 82.92%), again due to the implemented algorithms, hand positions, and gesture timings. In terms of sequential In-Out and Out-In gestures, the rate of recognition was 51.50% and 38.67%, being lower than the simple In or Out gestures (57.33% and 50.17%) and also much lower than the reference value (70%).

ST VL6180X

Test\Gesture	Up	Down	Right	Left
Best	88	88	92	90
Worst	64	68	68	70
Average	73.67	75.50	78.33	80.17

Test\Gesture	Up-Down	Down-Up	Right-Left	Left-Right
Best	82	88	88	86
Worst	60	66	68	70
Average	71.33	73.17	75.92	77.33

TABLE 5.4: ST VL6180X Battery of tests results

The final set of tests using the ST VL6180X sensing device, presented in Table 5.4, show the performance of the implemented algorithms.

Simple Gestures

Considering the vertical swipe gestures (Up and Down), the rate of recognition was lower than the horizontal gestures (Right and Left), due to the sensing devices displacement, varied user hand positions and gesture speeds. For this reason, the results were acceptable, with a range of 73.67 to 75.50 in horizontal swipe gestures and 78.33 to 80.17 in vertical swipe gestures. The In or Out gestures were not present in the test, because this sensor was used for direct proximity measurements, and not in the detection of In or Out gestures.

Sequential Gestures

In the case of sequential gestures, it is possible to observe that the vertical sequential swipe gestures had a performance lower than horizontal gestures (71.33 and 73.17 vs. 75.92 and 77.33), again due to the implemented algorithms, calibrations, and gesture timings.

Showroom Demo Application:

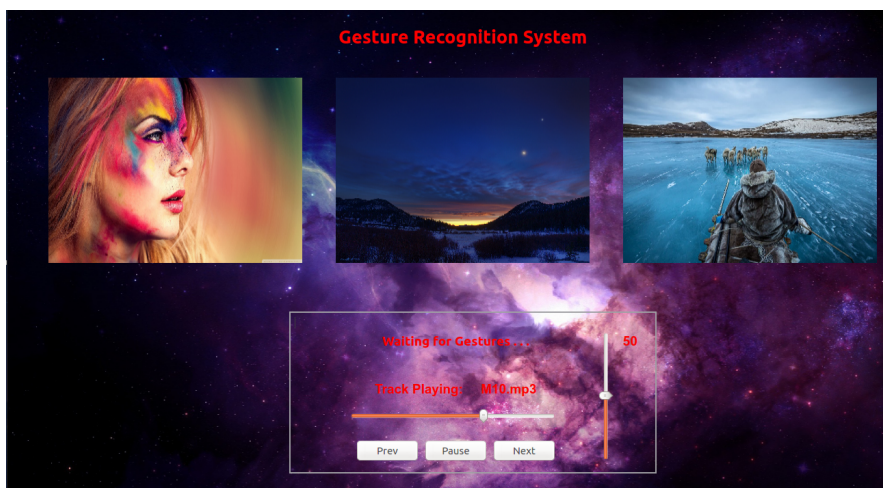
As a complement to the final tests, for demonstration purposes a demo application was created. The application in question is capable of recognizing the input of the gesture recognition system, using serial interface (alternative options could be the use of Bluetooth or TCP/IP communication), and then create a gesture based interface capable of controlling a image gallery and a music playlist.

The real usability of the application is limited, but it serves as an example of possible interactions using gestures and HMI functionalities.

The application can recognize up to 15 gestures, mapping these gestures to 8 functionalities, namely music track control(previous track, next track, start track and stop track), volume control (increase/decrease volume) and image gallery navigation (previous and next set of images).



(A)



(B)

FIGURE 5.17: Demo application screenshots.

Like it is possible to see, in Figure 5.17, the application can trace the gestures being detected, can update the images being displayed, and can control the music playing in the background (music name is displayed in the application text box (in the bottom)).

5.4.3 Tests Analysis

Considering the tests developed in this section, it is possible to conclude that the sensing devices used lead to acceptable results, however certain types of gestures were not detected with the expected performance, especially the In, Out, In-Out and Out-In gestures.

It is possible to observe a low similarity between the results achieved with Avago APDS-9960 and ams TMG4903, which was not so expected since the device interface and gesture algorithms are similar, but in the case of the Avago APDS9960, the linearity

of the sensing device is superior, despite a lower resolution, as well as the distance where the sensor have its output saturated.

Another aspect is that PixArt PAJ7620U2 registered the best overall test results, but with very similar results to the Avago APDS9960, even with low customization and calibration settings, being the best option not only in terms of performance but especially in gesture types detected.

ST VL6180X and ams TMG4903 registered the weakest results, but still above the reference values expected. In the case of the ST VL6180X, a possible source of performance loss was the sensing devices displacement chosen for the testing.

5.4.4 HMI Concepts Analysis

Based on the test results, in the selection of concepts presented in section 3.8 and especially considering HMI concepts inside the car (since they are the main area of exploration of this thesis and project), important conclusions can be achieved.

- In concepts involving mostly proximity measurements, the better option is to use the ST VL6180X sensing device, mainly for the higher reliability, precision and immunity to external interferences, such as ambient light, temperature, and object color;
- In concepts purely involving gesture detection, sensors like the PixArt PAJ7620U2 and Avago APDS-9960 can lead to better performance. The ams TMG4903 and ST VL6180X sensors are also options, but with worst performance.

It is important to consider the low performance achieved in the In, Out, In-Out and Out-In gestures in the development of the selection of HMI concepts;

The next figure, Figure 5.18, shows the sensing devices mapping to each possible HMI concept previously presented.

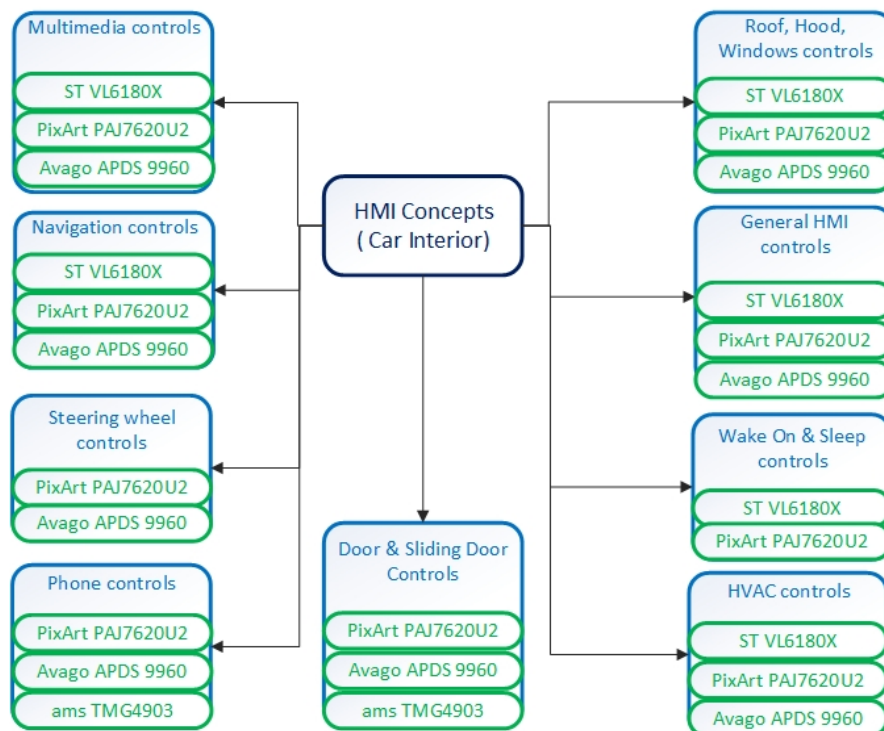


FIGURE 5.18: HMI Concept and possible Sensing devices.

Chapter 6

Conclusions and Future work

In this chapter, a brief reflection of the development of the research project is presented. This reflection covers the conclusions of the project, as well as important proposals for future work. These proposals are important to create a more effective and viable system, which is important considering the possible use cases of this thesis.

6.1 Conclusions

As part of the present thesis dissertation it was conducted a study related to the application of IR sensing devices in the creation of alternative HMI concepts with gesture recognition support.

This study was developed and directed to answer three main research questions covering the selection of the sensing devices to explore, question solved with a brief market study, also covering the set of gestures capable to be recognized with the selection of sensing devices, question solved with the development of gesture detection API's and demos, and finally the question of the selection of automotive HMI concepts capable of being validated with the selection of the sensing devices and gestures. This last question was succinctly solved after the period of tests and validation, documented in the previous chapter.

In order to test and validate the gesture recognition systems, the assembling of the gesture recognition systems and the development and testing of the devices API's was needed. With the execution of these tasks it was possible to answer the need to collect data from the sensors, the need to process that data and the need to communicate the data to a remote system, capable of processing this data and associate the execution of HMI functionalities.

The developed gesture recognition systems are capable of detecting and recognizing gestures with different hand and finger positions, speeds, distances and execution times. Also in terms of capabilities, the systems are capable of granting low calibration levels as well as detecting and recognizing from 6 to 15 gestures.

In terms of performance, the system is capable of processing the gesture data, as soon as the gesture execution ends, and communicate to the central/remote system in a period of time of less than 0.7 seconds, which was under the maximum response time set as requirement (1 second). Also, the system is capable of detecting more than 70% of the gestures, being the actual percentage variable depending of the sensing device and the working conditions and gesture characteristics, percentage that can go from around 86% to 97% of the gestures.

Furthermore, the system could not ensure a percentage of gesture recognition of more than 70% in all the gestures detected, mainly due to the performance of the algorithms for "In", "Out" and sequential "Out-In" and "In-Out" gestures. The percentage

of recognition varies from 38.67% (sequential Out-In gesture) to around 87.33% (Right Gesture) of the gestures, depending on the sensing device, gesture type, working condition, and gesture characteristics.

In terms of contributions the project was important for the development of a base of knowledge, mostly practical, capable of being used in the implementation of new HMI concepts based on gesture recognition, which was the main objective of the thesis.

This HMI concepts can cover the automotive area, as well as other areas of interest for gesture-based HMIs, such as remote contactless controllers, home appliances, virtual showcases, and many others applications.

6.2 Future Work

The study carried out in this dissertation, like many others, always has room for improvements, because several decisions were made and work steps were executed, being clearly possible to improve the work developed.

When talking about the technology used, a important step in the future should be the comparison of this technology with other technologies in terms of real use case scenarios, for example, by creating and testing prototype systems and demos using technologies such as IR, Capacitive and TOF Camera technology.

In terms of sensing devices, this thesis covers four sensing devices, selected under specific criteria. A proposal for future work could be the exploration and comparison of others sensing devices, with the selection of devices used in this thesis. This proposal is important, mainly because it could lead to the analysis of sensing devices more suited for specific purposes, and can also lead to new options of applications for the development of new HMI concepts.

In terms of gesture recognition algorithms and interface codes, the improvements can be very relevant, either by improving the actual algorithms and API's or by adopting new algorithms. These changes in algorithms can lead to better detection and recognition results results and also to a wider range of gestures detected, allowing that way the creation of new concepts of interaction and HMIs. A comparison study between algorithms can be also very relevant, to achieve an optimal approach.

Another important aspect that can be improved is the validation process of this thesis. The work developed was validated on in-Lab conditions, with a limited range of tests. As future work, the validation process should take in consideration the insertion of these systems in a driving simulation mockup, such as Bosch DSM. This insertion will allow the realization of more realistic testing procedures. Here, it is important the help of a Human Factors team, but it is also important the use of a wider group of people and concepts of HMI to test the system.

In case of the system be approved in the tests realized in a driving simulator, a possible option is also the case of a real world testing scenario, in a closed circuit with controlled driving conditions. Here an option to take in consideration is the development of tests in a racing circuit, such as the racing circuit Vasco Sameiro (also known as "Circuito de Braga"). This option is interesting because allows real world tests, with safety to the test subjects and others and without major legal problems or limitations.

Bibliography

- [1] W. H. Organization, “GLOBAL STATUS REPORT ON ROAD,” World Health Organization, Tech. Rep., 2015. [Online]. Available: <http://apps.who.int/iris/bitstream/10665/189242/1/9789241565066{ }eng.pdf?ua=1>
- [2] —, “WHO - Road Traffic Accidents,” World Health Organization, Tech. Rep., 2015. [Online]. Available: <http://www.who.int/violence{ }injury{ }prevention/road{ }safety{ }status/2015/magnitude{ }A4{ }web.pdf?ua=1>
- [3] Centers for Disease Control and Prevention, “CDC Distracted Driving,” Centers for Disease Control and Prevention, Tech. Rep., 2015. [Online]. Available: <http://www.cdc.gov/Features/dsDistractedDriving/>
- [4] P. Green, “Driver Interface / HMI Standards to Minimize Driver Distraction / Overload,” *Convergence 2008*, p. 17, 2008. [Online]. Available: <http://umich.edu/{ }driving/publications/Green2008Convergence.pdf>
- [5] M. R. & M. H. K. Young, “Driver distraction: a review of the literature,” *Monash University Accident Research Centre*, no. 206, p. 66, 2003. [Online]. Available: <http://www.monash.edu.au/miri/research/reports/muarc206.pdf>
- [6] “Image - No Title.” [Online]. Available: <https://i.ytimg.com/vi/H2eqtHj2wK0/maxresdefault.jpg>
- [7] S. Albuquerque, “Books @ Books.Google.Pt,” p. 141, 2003. [Online]. Available: <https://books.google.pt/books?id=VszO{ }kZl4bUC{ }printsec=frontcover{ }hl=pt-PT>
- [8] S. Lenman, L. Bretzner, and B. Thuresson, “Computer vision based hand gesture interfaces for human-computer interaction,” *CID, Stockholm, Sweden*, 2002. [Online]. Available: <http://cid.nada.kth.se/pdf/CID-172.pdf>
- [9] P. Peixoto, “a Natural Hand Gesture Human Computer Interface Using Contour Signatures,” -, 2005. [Online]. Available: <http://home.isr.uc.pt/{ }joaoluis/papers/HCI2005-Final.pdf>
- [10] A. Cienki and C. Müller, “Metaphor , gesture , and thought,” *The Cambridge Handbook of Metaphor and Thought*, 2013. [Online]. Available: <https://www.kuwi.europa-uni.de/de/lehrstuhl/sw/sw0/{ }texte/introgestureanalysis/Cienki{ }M{ }{ }llerMetaphor{ }Gesture.pdf>
- [11] J. Eisenstein and R. Davis, “Visual and Linguistic Information in Gesture Classification,” *International Conference on Multimodal Interfaces (ICMI'04)*, pp. 113–120, 2004. [Online]. Available: <http://www.google.fr/url?sa=t{ }rct=j{ }q={ }esrc=s{ }source=web{ }cd=1{ }cad=rja{ }ved=0CDQQFjAA{ }url=http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.217.6904{ }rep=rep1{ }type=pdf{ }ei=qt6xUdjLNorC7AbLvIGQDQ{ }usg=AFQjCNGntugTuGITA94yTJs3kCmn-SQ>

- [12] S. D. Kelly, S. M. Manning, and S. Rodak, "Gesture gives a hand to language and learning: Perspectives from cognitive neuroscience, developmental psychology and education," *Linguistics and Language Compass*, vol. 2, no. 4, pp. 569–588, 2008. [Online]. Available: <http://faculty.washington.edu/losterho/Compass.pdf>
- [13] R. M. Krauss, R. a. Dushay, Y. Chen, and F. Rauscher, "The Communicative Value of Conversational Hand Gesture," pp. 533–552, 1995. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0022103185710244>
- [14] R. M. Krauss, Y. Chen, and P. Chawla, "Nonverbal Behavior and Nonverbal Communication: What do Conversational Hand Gestures Tell Us?" *Advances in Experimental Social Psychology*, vol. 28, no. C, pp. 389–450, 1996. [Online]. Available: <http://www.columbia.edu/~rmk7/PDF/Adv.pdf>
- [15] R. M. Krauss, "Why Do We Gesture When We Speak?" *Current Directions in Psychological Science*, vol. 7, pp. 54–59, 1998. [Online]. Available: <http://www.columbia.edu/~rmk7/PDF/CD.pdf>
- [16] C. a. Pickering, K. J. Burnham, and M. J. Richardson, "A research study of hand gesture recognition technologies and applications for human vehicle interaction," *3rd Conf. on Automotive . . .*, pp. 1–15, 2007. [Online]. Available: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.135.7688&rep=rep1&type=pdf>
- [17] "0b5f3305fafb90daa8629c212032c558e6787dad.pdf." [Online]. Available: http://www.cogsci.ucsd.edu/~nunez/COGS160/McNeill{}_PS.pdf
- [18] "integrating-gesture-control-inside-the-car @ www.digikey.com." [Online]. Available: <http://www.digikey.com/en/articles/techzone/2012/aug/integrating-gesture-control-inside-the-car>
- [19] "interpret-gesture-recognition-and-hmi-design @ analysis.telematicsupdate.com." [Online]. Available: <http://analysis.telematicsupdate.com/infotainment/interpret-gesture-recognition-and-hmi-design>
- [20] "gesture @ www.media.mit.edu." [Online]. Available: http://www.media.mit.edu/gnl/publications/gesture{}_workshop/gesture.wkshop.html
- [21] "Gesture @ en.wikipedia.org." [Online]. Available: <https://en.wikipedia.org/wiki/Gesture>
- [22] "gesture @ www.justinecassell.com." [Online]. Available: <http://www.justinecassell.com/publications/gesture.wkshop.html>
- [23] "Image - No Title." [Online]. Available: <http://previews.123rf.com/images/stylephotographs/stylephotographs1307/stylephotographs130700106/20936342-Many-hands-showing-different-gestures-with-the-fingers-Stock-Photo.jpg>
- [24] "Image - No Title." [Online]. Available: <http://cdn.designbeep.com/wp-content/uploads/2013/09/8.gesture-icons.jpg>
- [25] "Procedural_memory @ en.wikipedia.org." [Online]. Available: https://en.wikipedia.org/wiki/Procedural{}_memory

- [26] “Implicit_memory @ en.wikipedia.org.” [Online]. Available: https://en.wikipedia.org/wiki/Implicit_memory
- [27] “Image - No Title.” [Online]. Available: <http://brain-basedlearning.weebly.com/uploads/1/2/4/4/12446008/7414212.jpg?557>
- [28] “43595-procedural-memory @ www.livescience.com.” [Online]. Available: <http://www.livescience.com/43595-procedural-memory.html>
- [29] “18646622 @ www.ncbi.nlm.nih.gov.” [Online]. Available: <http://www.ncbi.nlm.nih.gov/pubmed/18646622>
- [30] “what-is-procedural-memory @ www.wisegeek.com.” [Online]. Available: <http://www.wisegeek.com/what-is-procedural-memory.htm>
- [31] “Human_brain @ en.wikipedia.org.” [Online]. Available: https://en.wikipedia.org/wiki/Human_brain
- [32] “brain_parts @ www.human-memory.net.” [Online]. Available: http://www.human-memory.net/brain_parts.html
- [33] “Image - No Title.” [Online]. Available: <https://upload.wikimedia.org/wikipedia/commons/thumb/9/9f/Brainlobes.svg/313px-Brainlobes.svg.png>
- [34] “Are-You-Born-With-All-Your-Brain-Cells-or-Do-You-Grow-New-Ones @ Www.Brainfacts.Org.” [Online]. Available: <http://www.brainfacts.org/about-neuroscience/ask-an-expert/articles/2012/are-you-born-with-all-your-brain-cells-or-do-you-grow-new-ones>
- [35] S. Albuquerque, “Books @ Books.Google.Pt,” p. 141, 2003. [Online]. Available: https://books.google.pt/books?id=VsZO_kZl4bUC&printsec=frontcover&hl=pt-PT
- [36] M. Negulescu, J. Ruiz, Y. Li, and E. Lank, “Tap , Swipe , or Move : Attentional Demands for Distracted Smartphone Input,” *Proceedings of the International Working Conference on Advanced Visual Interfaces - AVI '12*, pp. 173–180, 2012. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2254589>
- [37] M. Bhuiyan and R. Picking, “Gesture-controlled user interfaces , what have we done and what ’ s next ?” *5th Collaborative Research Symposium on Security, E-Learning, Internet and Networking*, pp. 59–60, 2009. [Online]. Available: http://www.glyndwr.ac.uk/Computing/Research/pubs/SEIN_BP.pdf
- [38] S. Albuquerque, “Books @ Books.Google.Pt,” p. 141, 2003. [Online]. Available: https://books.google.pt/books?id=VsZO_kZl4bUC&printsec=frontcover&hl=pt-PT
- [39] S. Gupta, P. Molchanov, X. Yang, K. Kim, S. Tyree, and J. Kautz, “Towards Selecting Robust Hand Gestures for Automotive Interfaces,” 2010. [Online]. Available: https://research.nvidia.com/sites/default/files/publications/paper_3.pdf
- [40] “Image - No Title.” [Online]. Available: <http://www.automotiveworld.com/wp-content/uploads/2014/03/Visteon-Horizon-Cockpit.jpg>

- [41] "Image - No Title." [Online]. Available: [https://d112vpovu2xa8r.cloudfront.net/portal/{_}automotiveitnews{_\]curatasite{_\]com/media/wcBm6GFjRkb7nPP.jpeg](https://d112vpovu2xa8r.cloudfront.net/portal/{_}automotiveitnews{_]curatasite{_]com/media/wcBm6GFjRkb7nPP.jpeg)
- [42] "Image - No Title." [Online]. Available: <http://assets.inhabitat.com/wp-content/blogs.dir/1/files/2013/10/google-gesture-car-controls-patent.jpg>
- [43] "Image - No Title." [Online]. Available: [http://i.telegraph.co.uk/multimedia/archive/02724/ford-gestures{_\]2724017c.jpg](http://i.telegraph.co.uk/multimedia/archive/02724/ford-gestures{_]2724017c.jpg)
- [44] Nielsen-J, *Usability engineering*. Boston: Academic Press, 1993.
- [45] "Image - No Title." [Online]. Available: <http://jupiter.plymouth.edu/{~}wjt/HCI/usability.jpg>
- [46] "Image - No Title." [Online]. Available: [http://mcom.cit.ie/staff/Computing/prothwell/hci/mcsd/Notes/HCID5{_\]files/image017.jpg](http://mcom.cit.ie/staff/Computing/prothwell/hci/mcsd/Notes/HCID5{_]files/image017.jpg)
- [47] "c53e082511ebc549f31e9384165b01de15dc97d2.pdf." [Online]. Available: [http://davidcohen.mit.edu/sites/default/files/documents/1967ScienceV156\(coilMCG\).pdf](http://davidcohen.mit.edu/sites/default/files/documents/1967ScienceV156(coilMCG).pdf)
- [48] "capacitive @ www.sensorwiki.org." [Online]. Available: <http://www.sensorwiki.org/doku.php/sensors/capacitive>
- [49] A. Braun, "Application and validation of capacitive proximity sensing systems in smart environments," -, 2014. [Online]. Available: <http://tuprints.ulb.tu-darmstadt.de/4175/>
- [50] "Image - No Title." [Online]. Available: [https://www.renesas.com/en-us/media/solutions/proposal/touch-key/touch{_\]sensor{_\]system{_\]figure02.jpg](https://www.renesas.com/en-us/media/solutions/proposal/touch-key/touch{_]sensor{_]system{_]figure02.jpg)
- [51] "Image - No Title."
- [52] "Image - No Title." [Online]. Available: [http://www.sensorwiki.org/lib/exe/fetch.php/sensors/capacitance{_\]transmit{_\]mode.png?cache=cache](http://www.sensorwiki.org/lib/exe/fetch.php/sensors/capacitance{_]transmit{_]mode.png?cache=cache)
- [53] "Image - No Title." [Online]. Available: [http://www.kerrywong.com/blog/wp-content/uploads/2011/01/2000px-Sonar{_\]Principle{_\]EN.svg{_\]300x160.png](http://www.kerrywong.com/blog/wp-content/uploads/2011/01/2000px-Sonar{_]Principle{_]EN.svg{_]300x160.png)
- [54] "Image - No Title." [Online]. Available: [http://www.skyradar.com/wp-content/uploads/2014/11/radar{_\]animation{_\]01.gif](http://www.skyradar.com/wp-content/uploads/2014/11/radar{_]animation{_]01.gif)
- [55] "Image - No Title." [Online]. Available: [http://education.rec.ri.cmu.edu/content/electronics/boe/ir{_\]sensor/images/409px-IR{_\]Sensor{_\]Principles.png](http://education.rec.ri.cmu.edu/content/electronics/boe/ir{_]sensor/images/409px-IR{_]Sensor{_]Principles.png)
- [56] "Image - No Title." [Online]. Available: [https://learn.adafruit.com/system/assets/assets/000/000/511/medium800/proximity{_\]pirdiagram.jpg?1396763681](https://learn.adafruit.com/system/assets/assets/000/000/511/medium800/proximity{_]pirdiagram.jpg?1396763681)
- [57] "Image - No Title." [Online]. Available: <http://www.embedded-vision.com/sites/default/files/technical-articles/3DSensors/Figure3.png>
- [58] "Image - No Title." [Online]. Available: <http://www.laserfx.com/Science/s-proj2.gif>

- [59] “Image - No Title.” [Online]. Available: http://www.laserfocusworld.com/content/dam/etc/medialib/new-lib/laser-focus-world/online-articles/2011/01/93355.res/{_}jcr{_}content/renditions/pennwell.web.390.296.gif
- [60] “Image - No Title.” [Online]. Available: <http://mesh.brown.edu/3dpgp-2009/homework/hw2/figures/teaser.jpg>
- [61] “Image - No Title.” [Online]. Available: <http://www.mouser.com/images/microsites/time-of-flight-roboticsfig1.png>
- [62] “Image - No Title.” [Online]. Available: http://tctechcrunch2011.files.wordpress.com/2013/10/2013-10-02{_}13h07{_}34.jpg
- [63] “Image - No Title.” [Online]. Available: <http://www.depthbiomechanics.co.uk/wp-content/uploads/2012/06/stereo-vision-cams.jpg>
- [64] “Image - No Title.” [Online]. Available: <http://www.skytopia.com/gallery/3d/stereo/pyramid.jpg>
- [65] “Image - No Title.” [Online]. Available: <http://cdn.traceparts.com/wp-content/uploads/2014/09/revolutionary-gesture-control-armband-myo.png>
- [66] “Image - No Title.” [Online]. Available: <http://api.sonymobile.com/files/SmartWatch-3-SWR50-gallery-02-1240x840-ee14d83190f2a130bf389af4d9ff159e.jpg>
- [67] “Image - No Title.” [Online]. Available: <http://www.wearable.com/media/images/2015/07/smarty-ring-1436197312-THBt-column-width-inline.jpg>
- [68] “Infrared @ en.wikipedia.org.” [Online]. Available: <https://en.wikipedia.org/wiki/Infrared>
- [69] “07_infraredwaves @ missionscience.nasa.gov.” [Online]. Available: http://missionscience.nasa.gov/ems/07{_}infraredwaves.html
- [70] “50260-infrared-radiation @ www.livescience.com.” [Online]. Available: <http://www.livescience.com/50260-infrared-radiation.html>
- [71] “Image - No Title.” [Online]. Available: <http://fcats-standard-sc8e511.weebly.com/uploads/1/3/3/9/13393140/1639378.jpg?418>
- [72] “Image - No Title.” [Online]. Available: <https://upload.wikimedia.org/wikipedia/commons/thumb/0/05/TOF-camera-principle.jpg/800px-TOF-camera-principle.jpg>
- [73] “Image - No Title.” [Online]. Available: <http://eu.mouser.com/images/microsites/time-of-flight-roboticsfig2.png>
- [74] “continental-integrates-gesture-based-control-into-the-steering-wheel @ www.inautonews.com.” [Online]. Available: <http://www.inautonews.com/continental-integrates-gesture-based-control-into-the-steering-wheel>
- [75] “Pr_2016_05_10_Wheel_Gestures_En @ Www.Continental-Corporation.Com.” [Online]. Available: http://www.continental-corporation.com/www/pressportal{_}com{_}en/themes/press{_}releases/3{_}automotive{_}group/interior/press{_}releases/pr{_}2016{_}05{_}10{_}wheel{_}gestures{_}en.html

- [76] "Threshold_model @ en.wikipedia.org." [Online]. Available: <https://en.wikipedia.org/wiki/Threshold{ }model>
- [77] J. A. B. Adriano, E. A. Aquino, C. J. V. Cabael, B. E. D. Castro, and K. N. S. Jamoralin, "GESTURE-BASED COMPUTER INTERACTION THROUGH INFRARED MOTION SENSING By," 2009.
- [78] H.-t. Cheng, A. M. Chen, A. Razdan, and E. Buller, "Contactless Gesture Recognition for Mobile Devices," -, 2011. [Online]. Available: <http://repository.cmu.edu/cgi/viewcontent.cgi?article=1018{&}context=silicon{ }valley>
- [79] Y. S. Kim and K.-h. Baek, "A motion gesture sensor using photodiodes with limited field-of-view," -, vol. 21, no. 8, pp. 555–560, 2013.
- [80] "DTWAlgorithm." [Online]. Available: <http://www.psb.ugent.be/cbd/papers/gentxwarper/DTWAlgorithm.ppt>
- [81] "Image - No Title." [Online]. Available: <http://www.psb.ugent.be/cbd/papers/gentxwarper/images/dtw{ }algorithm/DTWgrid.gif>
- [82] "Image - No Title." [Online]. Available: <https://www.redbrick.dcu.ie/{~}dwarf/proj/mmodel.gif>
- [83] "Image - No Title." [Online]. Available: <https://intoverflow.files.wordpress.com/2008/04/linear.png>
- [84] "Image - No Title." [Online]. Available: <https://intoverflow.files.wordpress.com/2008/04/leftright.png>
- [85] "Image - No Title." [Online]. Available: <https://intoverflow.files.wordpress.com/2008/04/ergodic.png>
- [86] J. Kim, J. He, K. Lyons, and T. Starner, "The Gesture Watch : A Wireless Contact-free Gesture based Wrist Interface," 2007.
- [87] G. Huang and S. W. Loke, "Gesture-Based Easy Computer Interaction using a Linear Array of Low Cost Distance Sensors," -, 2011.
- [88] S. Kratz and M. Rohs, "HoverFlow," *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '09*, p. 1, 2009. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1613858.1613864>
- [89] H. T. Cheng, A. M. Chen, A. Razdan, and E. Buller, "Contactless gesture recognition system using proximity sensors," *Digest of Technical Papers - IEEE International Conference on Consumer Electronics*, no. 1, pp. 149–150, 2011. [Online]. Available: <https://pdfs.semanticscholar.org/9227/c8548b79e35ccc8f7e9edebc435910e23c7d.pdf>
- [90] S. D. Badgajar, G. Talukdar, and O. Gondhalekar, "Hand Gesture Recognition System," -, vol. 4, no. 2, pp. 1–5, 2014. [Online]. Available: <http://www.ijecs.in/issue/v3-i11/58ijecs.pdf>
- [91] "Waterfall @ Www.Umsl.Edu." [Online]. Available: <http://www.umsel.edu/{~}hugheyd/is6840/waterfall.html>

- [92] “Waterfall_model @ en.wikipedia.org.” [Online]. Available: https://en.wikipedia.org/wiki/Waterfall{__}model
- [93] “03a9356edf5cb9724f45d73128c8feb01251d5b4 @ www.seguetech.com.” [Online]. Available: <http://www.seguetech.com/waterfall-vs-agile-methodology/>
- [94] “d7976e2c0cb7077a9fc83d8e6987f8ffa742ec82 @ testingfreak.com.” [Online]. Available: <http://testingfreak.com/waterfall-model-software-testing-advantages-disadvantages-waterfall-model/>
- [95] “Image - No Title.” [Online]. Available: http://image.nxp.com/v1.49/documents/marcom{__}graphics/OM13063{__}mbed{__}lpc4088{__}0.png
- [96] “Image - No Title.” [Online]. Available: <https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTgNHSE4IcIUm27snnWYMkPITPSiYG7Ct-XfrsE4VBFTs9CDxkE5A>
- [97] “Image - No Title.” [Online]. Available: http://rasbt.github.io/mlxtend/user{__}guide/evaluate/confusion{__}matrix{__}files/confusion{__}matrix{__}1.png
- [98] “Image - No Title.” [Online]. Available: <http://i.stack.imgur.com/yk98D.png>
- [99] “61877b579313c1693215436332458c02808db506 @ www.ni.com.” [Online]. Available: <http://www.ni.com/tutorial/13637/en/>
- [100] “DistanceSensor @ www.robotroom.com.” [Online]. Available: <http://www.robotroom.com/DistanceSensor.html>
- [101] “PNA4602M-Replacement-3 @ www.robotroom.com.” [Online]. Available: <http://www.robotroom.com/PNA4602M-Replacement-3.html>
- [102] “Lux @ en.wikipedia.org.” [Online]. Available: <https://en.wikipedia.org/wiki/Lux>
- [103] “light-level-rooms-d_708 @ www.engineeringtoolbox.com.” [Online]. Available: http://www.engineeringtoolbox.com/light-level-rooms-d{__}708.html