



SAPIENZA
UNIVERSITÀ DI ROMA

Faculty of Engineering

Department of Computer, Control and Management Engineering

Ph.D. thesis in Automatic Engineering and Operations Research

Cycle XXIX

PERSONALIZED ASSISTED HUMAN MACHINE INTERFACE

ADVISOR

Ing. Antonio Pietrabissa

CANDIDATE

Dario Giuseppe Ferriero

Abstract

The objective of this work is the design, development and creation of a system which enables the interaction between disabled people and Home Automation. The actual technologies in the field of Internet of Things, indeed, are a potential resource to be exploited to create aids oriented to subjects with medium-severe disabilities; such aids might help those subjects in overcoming daily struggles as for the interaction with the home environment. The system proposed in this work has as element of innovation the possibility of user-environment interaction in a personalized and assisted way: the system, indeed, behaves in a customized way for each user. This system takes name of Personalized Assisted Human Machine Interface (PAHMI).

The PAHMI system has a graphic interface (Human-Machine Interface, HMI) which shows to the user all possible tasks that can be given to the environment, represented on the HMI as squares oriented in circle. To reach the square corresponding to the desired command (target), the user can move a pointer (PC mouse) using, through the head movement, a smartphone endowed with an accelerometer.

For the univocal functioning of the system, adapted to the specific user, two phases are necessary: a first phase of learning and a second phase of use.

During the learning phase, trajectories given by the user to the pointer are analysed, to automatically learn the specific command features adopted by the user. To estimate user command features during online trajectories analysis, a classifier model named Support Vector Machine (SVM) has been implemented.

During the using phase, the objective is to help the user to exploit the desired trajectory and reach the target; to do so, a PID control system has been used to reach the desired target area of the HMI, having available the user command features (provided by the SVM during the first phase).

Moreover, the system has been physically implemented with the necessary hardware, assembling microcontrollers, WIFI transmitters and embedded systems to be able to actually give commands to the surrounding environment. Finally, a software system for the WIFI communication between HMI, main server and actuators placed in the home environment has been implemented.

The work proposed ended with an experimental phase and the analysis and verification of results.

Riassunto

L'obiettivo del presente lavoro è stato quello di progettare, sviluppare e creare un sistema che permettesse l'interazione tra utenti disabili e Ambiente Domotico. Le attuali tecnologie nell'ambito dell'Internet of Things, infatti, costituiscono una potenziale risorsa da poter sfruttare per la creazione di sussidi rivolti a persone affette da disabilità medio-severe; tali sussidi possono essere d'aiuto ai suddetti utenti per superare le problematiche quotidiane di gestione dell'ambiente domestico. Il sistema proposto con il presente lavoro introduce come elemento di innovazione la possibilità di interazione utente-ambiente in modo personalizzato ed assistito: il sistema, infatti, adotta un comportamento customizzato per ogni utente che lo utilizza. Tale sistema prende il nome di Personalized Assisted Human Machine Interface (PAHMI).

Il sistema è dotato di un'interfaccia grafica (Human-Machine Interface, HMI) attraverso la quale vengono presentati all'utente i possibili comandi impartibili all'ambiente circostante, rappresentati sulla HMI tramite appositi quadrati disposti in cerchio. Per raggiungere il quadrato corrispondente al comando desiderato (target), l'utente ha la possibilità di muovere un puntatore (il mouse del PC) utilizzando uno smartphone dotato di accelerometro e sfruttando il solo movimento della testa.

Per il funzionamento univoco del sistema, adattato al singolo utente, sono necessarie due fasi: una prima fase di esercizio ed una seconda fase di utilizzo.

Durante la fase di esercizio, vengono analizzate le traiettorie che l'utente impartisce al puntatore, con l'obiettivo di apprendere automaticamente le caratteristiche di comando univoche adottate dall'utente stesso. Per poter stimare le caratteristiche di comando dell'utente durante l'analisi online delle traiettorie, è stato implementato un modello classificatore denominato Support Vector Machine (SVM).

Nella fase di utilizzo, l'obiettivo è quello di aiutare l'utente nel suo intento di impartire una traiettoria desiderata e raggiungere il target; per fare ciò è stato applicato un sistema di controllo di tipo PID per il raggiungimento di un'area desiderata dell'HMI, avendo a disposizione le caratteristiche di comando dell'utente (fornite dalle SVM durante la prima fase).

Inoltre, il sistema è stato implementato fisicamente con l'hardware necessario, assemblando microcontrollori, trasmettitori WIFI e sistemi embedded, per permettere di impartire realmente i comandi all'ambiente circostante. Infine, è stato implementato un sistema software per la comunicazione WIFI tra HMI, server principale ed attuatori disposti nell'ambiente domestico.

Il lavoro proposto si è concluso con una fase sperimentale, l'analisi e la verifica dei risultati.

Contents

Abstract	2
Riassunto.....	3
1 Introduction.....	10
1.1 Glossary.....	11
2 The IoT phenomenon	13
2.1 The devices.....	13
2.2 The network.....	14
2.3 The applications	14
3 IoT Potential for persons with disabilities	15
3.1 Disability	15
3.2 Assistive technologies	16
3.3 IoT and Disability.....	17
3.4 Human Machine Interface	18
3.5 Human Factors	19
3.6 Home automation technologies	20
4 The Personalized Assisted HMI system.....	22
4.1 Functioning principle	22
4.2 The HMI of the system.....	23
4.3 The MATLAB mobile app of the system.....	24
4.4 Inclined plan force components	25
4.5 The Butterworth filter.....	27
5 Learning phase	32
5.1 Multi-class classification.....	32
5.2 SVM learning technique.....	33
5.3 Data collection.....	34

5.4	SVM training, validation and confusion matrices.....	35
6	The controlled system	37
6.1	Kinematic model	37
6.2	Control algorithm	38
6.2.1	Moving to point.....	39
6.3	Real time computer assisted control.....	39
7	Hardware and software implementation	45
7.1	The server of the main system.....	45
7.1.1	Qt library.....	47
7.1.2	Configuration	48
7.2	The server of the actuator.....	49
7.2.1	IRremote library.....	50
7.3	The hardware implementation.....	50
7.3.1	WIFI controlled relay using ESP8266	50
7.3.2	WIFI controlled IR led using ESP8266	52
8	Simulations.....	54
9	Conclusions and future works.....	64
	Bibliografy	66

List of images

Figure 1 - Off-line tuning of the multi-class classifier and on-line assisted HMI flow-chart; the dashed arrows indicate that the multi-class classifier is used in real-time by the on-line controller.	11
Figure 2 - Technology roadmap [1]	17
Figure 3 - Human Factors Interfaces [23]	19
Figure 4 - PAHMI data flow	23
Figure 5 - HMI with 8 targets	24
Figure 6 - MATLAB app user interface.	25
Figure 7 - Roll and Pitch angles of the smartphone orientation [26]	26
Figure 8 - Resultant forces of an inclined plane	26
Figure 9 - Magnitude response of an ideal n^{th} -order Butterworth filter [29]	28
Figure 10 - Mutation of the acceleration data through the Butterworth filter.	31
Figure 11 - Coordinates for differential drive wheeled mobile robot [41]	37
Figure 12 - Simulink block for the unicycle kinematic model	38
Figure 13 - Coordinates for moving to a point [43]	39
Figure 14 - Real time assisted control.	40
Figure 15 - Detailed real time assisted control	44
Figure 16 - Sequence diagram	45
Figure 17 - Main PAHMI communications	46
Figure 18 - TCP server/client communications	47
Figure 19 - Electric schematic of an embedded system to switch on/off a device connected to a relay	51
Figure 20 - Electric schematic of an embedded system to switch on/off a device via IR command.	52
Figure 21 - Operational space with 4 targets	54
Figure 22 - Recorded trajectory for a normal user (left) and a disabled user (right) to reach the first target.	55

Figure 23 - Recorded trajectory for a normal user (left) and a disabled user (right) to reach the second target	55
Figure 24 - Recorded trajectory for a normal user (left) and a disabled user (right) to reach the third target.....	56
Figure 25 - Recorded trajectory for a normal user (left) and a disabled user (right) to reach the fourth target.....	56
Figure 26 - Confusion matrices.....	58
Figure 27 - Controlled trajectory for a disabled user (left) and a normal user (right) to reach the first target.....	59
Figure 28 - Controlled trajectory for a disabled user (left) and a normal user (right) to reach the second target	59
Figure 29 - Controlled trajectory for a disabled user (left) and a normal user (right) to reach the third target.....	60
Figure 30 - Controlled trajectory for a disabled user (left) and a normal user (right) to reach the fourth target.....	60
Figure 31 - Improvement results for the disabled user to reach each target	62
Figure 32 - Improvement results for the normal user to reach each target	62

List of tables

Table 1 - Glossary	12
Table 2 - Examples of IoT functionalities for disability types	18
Table 3 - Simulation parameters	58
Table 4 - Resulted for the normal user.....	61
Table 5 - Resulted for the disabled user.....	61

1 Introduction

Objects around us have been connected for decades. Devices like TV, remote controls and garage door openers have been part of the domestic landscape for generations. Industrial applications of these technologies (for example, through remote monitoring and control of production) are also nothing new. In fact, even the phrase “Internet of Things” is not a recent invention, it was coined around twenty years ago. However, recent developments in both networks and devices are enabling a much greater range of connected devices and Internet of Things (IoT) functionalities. Today, the phrase “Internet of Things” refers to the world of smart connected objects and devices. Gone is the remote control, replaced by an intelligent device that will automatically fulfil its task based on its analysis of user behaviour. All of this is made possible by the miniaturization of electronic devices, accompanied by a huge increase in the availability of internet connectivity. The potential applications of this new IoT are virtually limitless and they can greatly improve quality of life [1].

This thesis explores a system that exploits the phenomenon of IoT to help persons with disabilities. This is perhaps most clearly true in home automation, where a person who is blind can monitor and control the lights through an accessible Human Machine Interface, or a person with a mobility-related disability can have his door automatically unlock when he approaches it.

The work is proposed by the Department of Computer, Control and Management Engineering of Sapienza University of Rome in collaboration with the Istituto Superiore di Sanità of Rome.

The objective of this work is to design a procedure to develop a Personalized and Assisted HMI (PAHMI) which actively and dynamically help the user to reach her/his desired target [2]. The proposed procedure is performed in two phases: firstly, a machine-learning based algorithm learns the user characteristics ([3], [4]) while she/he interacts with the (non-assisted) HMI; then, the model built by the algorithm is used by a control algorithm [5], [6] to assist the user in real-time (PAHMI). The thesis also presents an implementation of the proposed procedure, based on a HMI which requires the user to move a cursor toward some targets on the screen, on a Support Vector Machine (SVM) multi-class classifier playing the role of the machine-learning algorithm ([7], [8]), and on a PID controller [9] which is used to add a corrective action, based on the SVM predictions, to the cursor movement controlled by the user. The overall design concept is then validated by implementations and simulations.

The overview of this work is shown in Figure 1:

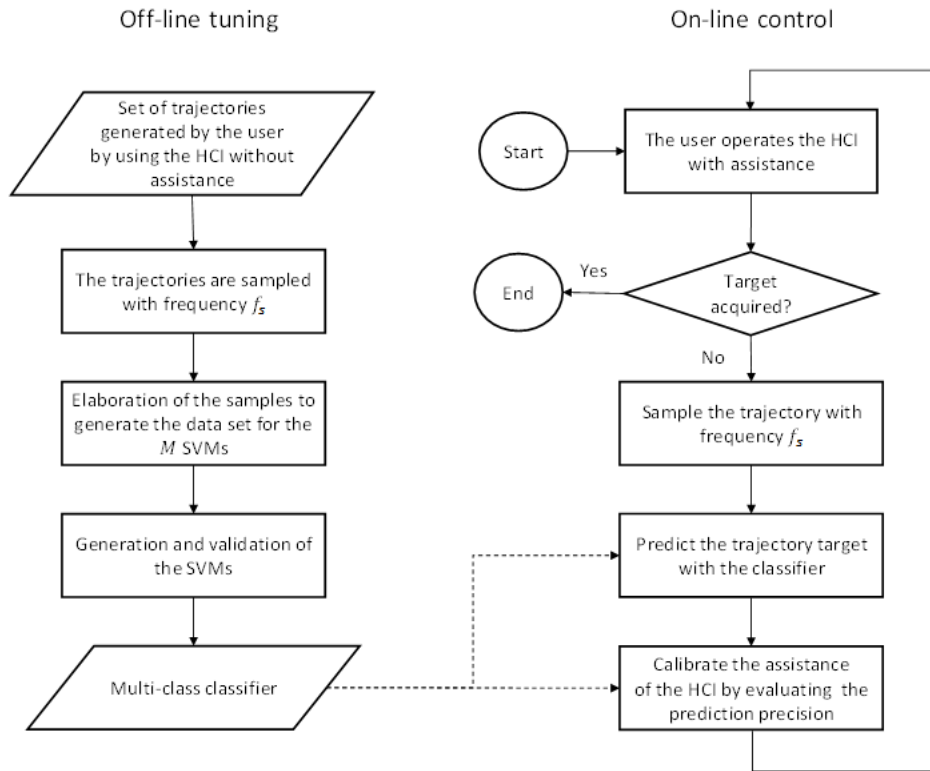


Figure 1 - Off-line tuning of the multi-class classifier and on-line assisted HMI flow-chart; the dashed arrows indicate that the multi-class classifier is used in real-time by the on-line controller

1.1 Glossary

ACRONYM	DESCRIPTION
APP	Application
DDWMR	Differential Drive Wheeled Mobile Robot
GPIO	General Purpose I/O
HCI	Human Computer Interface
HF	Human Factor
HMI	Human Machine Interface
I2C	Inter-Integrated Circuit
ICF	International Classification of Functioning, Disability and Health

I/O	Input/Output
IoT	Internet of Things
IP	Internet Protocol
MEMS	Micro-Electro-Mechanical Systems
MMI	Man Machine Interface
PAHMI	Personalized Assisted Human Machine Interface
PC	Personale Computer
PID	Proportional Integral Derivative
PPV	Positive Predictive Value
QP	Quadratic Problem
RBF	Radial Basis Function
SVM	Support Vector Machine
SW	Software
TCP	Transmission Control Protocol
UI	User Interface

Table 1 - Glossary

2 The IoT phenomenon

The Internet of Things is ushering in a societal revolution [10]: anything and everything can now be connected. While industrial applications like remote management and monitoring have been present in manufacturing for decades (General Electric has over the years developed some standards for industrial uses), the consumer-oriented IoT is a new phenomenon, enabled by the miniaturization of devices, the emergence of the internet as a favoured communications method and the proliferation of powerful and smart mobile devices.

The internet of things is made possible by what Alain Louchez, managing director of Georgia Tech's Centre for the development and Application of Internet of Things Technologies (CDAIT), calls "DNA" [1]:

- Devices: with miniaturization of sensors and captors
- Network: through radio signal, wireless Internet, etc.
- Applications: desktop, embedded and mobile

2.1 The devices

Over the past few decades, researchers and industry players have developed Micro-Electro-Mechanical Systems (MEMS) [11]. These are miniaturized structures, sensors, actuators and microelectronics, the sizes of which range from microns to several millimetres. Micro sensors and micro actuators are categorized as "transducers," which are defined as devices that convert energy from one form to another. In the case of micro sensors, the device typically converts a measured mechanical signal into an electrical signal.

These new microelements are very powerful: for example, researchers have placed small micro-actuators on the leading edge of airfoils of an aircraft and could steer the aircraft using only these devices. As the technology evolves, the merger of MEMS with the integrated circuits that power computing will become more common and will make more complex applications available to the end user.

Ultimately, miniaturization will allow any object to be connected, which could have substantial positive impacts for persons with disabilities as more and more products can be controlled from accessible computing platforms.

The number of connected devices is set to explode from 4.9 billion this year to 25 billion or even 50 billion by 2020.

2.2 The network

As important as these advancements in microelectronics are, the Internet of Things refers not just to devices, but also to the connections between them. For IoT applications to work, the sensors and the actuators must be able to communicate with the devices that inform their action, whether it is a smartphone or something as simple as a remote thermometer [12].

Communication between devices can be established by a variety of different modes, including radio signals, Bluetooth (for devices in proximity) or an Internet connection (Wi-Fi and wireless broadband for a wider range of communication and for video). The potential applications for the Internet of Things will continue to expand.

2.3 The applications

New IoT applications are being introduced to the market every day. While IoT applications are expected to penetrate into many activities—both consumer and industrial—smart home applications seem to be the fastest growing segment. Connected home devices are expected to comprise 25% of all Internet of Things devices shipped this year a market currently valued at \$61 billion and expected to jump to \$490 billion by 2019 [1].

Many of these have potential to improve quality of life for persons with disabilities. Home automation applications and security systems are an obvious example of this. Other examples include self-driving cars that identify traffic and obstacles and remote medical monitoring that makes access to care available to more people in more places [13].

3 IoT Potential for persons with disabilities

The increase of innovation technology, with the IoT expansion applied to assistive technologies, provides the opportunity to improve the possibilities of interaction between disabled patients and the surrounding environment [14]. For this reason, the concept of disability has changed significantly over the last decade [15].

3.1 Disability

Disability concept has deeply changed over time from a scientific, cultural and social point of view. This is evident from the following definition [16]:

People with disabilities are those who have permanent physical, mental, intellectual or sensory impairments which, interacting with barriers of different nature, may be prevented from effectively participate in social life on an equal basis with other people (Convention of United Nations on people with disabilities rights – 30 March 2007).

Apart from specific terminology, today it is used the expression *person with disabilities*, highlighting that problems are not related to health condition or body functioning but to the effects of disabilities on social context.

This implies that the handicap is not related to the individual but to the interaction between disability and environment. Effects of disability may often be reduced through enabler and educational methods, using different aids. Indeed, if we think to a blind person (impairment), he/she is not able to read (disability) but thanks to a speech synthesis (aid), he/she will be able to use the PC to read a document. If in the working environment are used documents which cannot be read by a PC (barrier), the person will not be able to work (handicap).

In 1999 the International Classification of Functioning, Disability and Health (ICF) has been introduced with the aim of building a method of international classification in order to describe, in a consistent way, the capability limits of a person [17]. The sequence Impairment – Disability – Handicap has been replaced by the sequence

Body function and structures – Personal activity – Social participation

which represents a neutral connotation and permits to describe any situation, not only that of disability/handicap. The ICF takes into consideration also physical conditions (disorder or disease) and contextual factors (environmental and personal); for these reasons the ICF is considered a

method which uses a biopsychosocial approach, where health is assessed globally through its biological, individual and social dimensions. The ICF model considers disability as the result of a complex interaction between person's health conditions, personal factors and environmental factors. Even if the impairment is not removable, barriers can be removed and the disadvantages related to disability may be reduced.

Technological progress represents a chance to radically modify the entire project of life for a person with disability, since it gives instruments and supports which can efficiently help with study, learning, communication, independence and integration.

Technology, indeed, plays a role of "facilitator" in the process of inclusion of a person with disability.

The process of inclusion, inspired by a people-centered principle, makes everyone responsible for an early institutional care obligation and permits to program and realize an individualized project suited to personal needs and abilities, able to stimulate attention and commitment.

The first fundamental step for an efficient path of inclusion is the knowledge of good routine procedures, methods, opportunities, instruments and innovative services already available in the home territory.

Technology needs to be a way to further improve quality life and not to create new barriers [18]; in order to let technology be a way of social integration and personal growth it is necessary that all people involved respects shared and participated principles.

3.2 Assistive technologies

In the field of information technologies, the assistive technologies are those realized specifically to make information products (hardware or software) accessible and usable even to person with disabilities [19]. Assistive technologies are often human-machine interfaces developed from an original product conceived for normal users; because of their refined design and production and of their limited diffusion, the assistive technologies tend to be very expensive for the final disabled user. Common examples are screen reader, braille keyboard for blind people, pointer mouse etc.

The possibility of making assistive technologies available for persons with disabilities represents just one of the aspects of the path to independence and inclusion.

Assistive technologies, which can compensate for limitations or deficits, cannot be applied in the same way to each disability, even if similar, but they are bond to a specific person according to multiple individual factors (age, degree of disability, residual potential and effective abilities, assistive technology acceptance etc.), included the context of their application [20].

New technologies, then, may play a fundamental role in supporting social integration of persons with disabilities, being aware that introducing technological aids in people lives may be, in most cases, a delicate and long process which has to be preliminarily evaluated and requires team work, professionalism, time and experience.

3.3 IoT and Disability

Many new IoT applications are comfort and security oriented. These applications enable a user to change his or her thermostat remotely, decrease or increase the intensity of lights, control door locks, activate alarm systems, etc. While these applications certainly add a level of fun and convenience for all users, the applications take on a whole new level of importance when used by persons with disabilities and older adults [21].

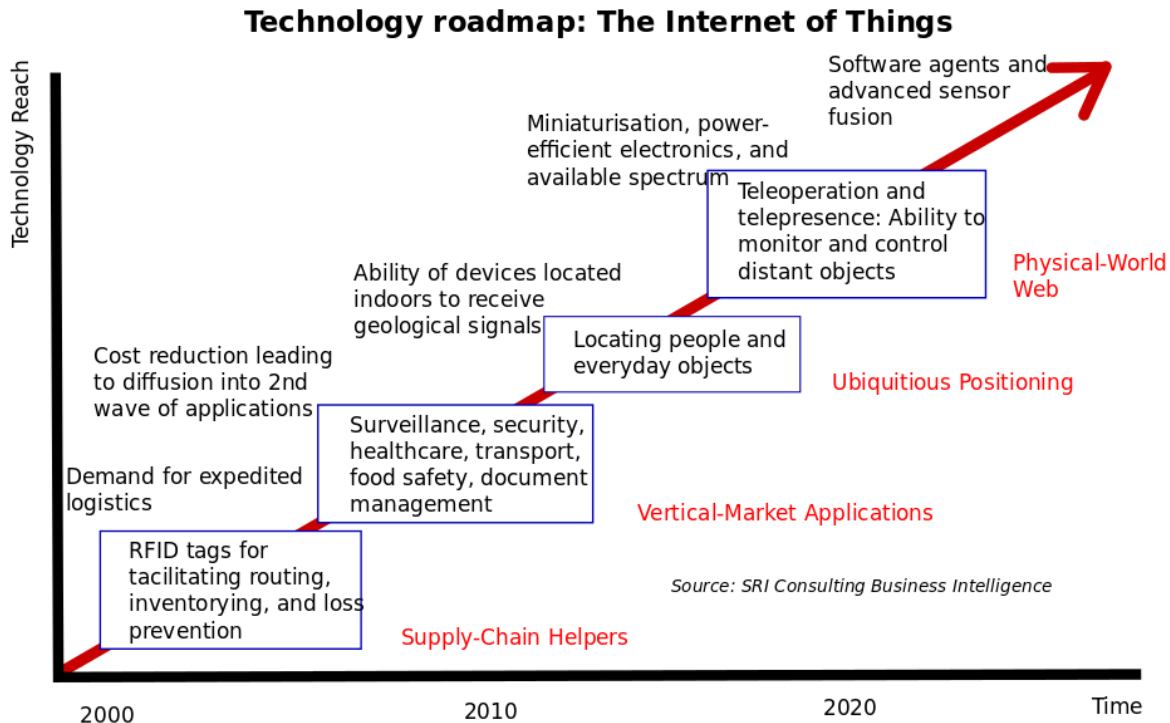


Figure 2 - Technology roadmap [1]

Thanks to improved accessibility features, smartphones can now be used by more persons with disabilities than ever before. Many manufacturers have also developed smartphones with easier and more intuitive interfaces. These interfaces facilitate use by older adults whose usage of smartphones may otherwise be impacted by vision or hearing loss or reduced dexterity and mobility. These accessible smartphones are already becoming the remote controls for smart homes, smart transportation and smart cities, giving persons with disabilities a new and flexible instrument to navigate daily life and participate more fully in the economic and social environment.

Industry experts agree that the Internet of Things has tremendous potential to bring benefits to persons with disabilities. The ability to adjust things such as the temperature and brightness of a room may also be helpful for people with limited mobility. Additionally, the IoT has significant potential to improve employment opportunities, facilitate independent living and improve social integration for persons with disabilities.

Type of Disability	Examples of Useful Functionalities Enabled by IoT
Physical and Dexterity	Remote support and services at home Speech activated devices Automated accessibility functions in public spaces
Visual	Interpretation of user environment for way finding Near field automation Speech activated devices which communicate with speech output
Hearing	Captioning in glasses delivered by beacons Visual cues about status of home devices on mobile device
Cognitive	Localization and orientation Automated reminders Programmable safety processes

Table 2 - Examples of IoT functionalities for disability types

3.4 Human Machine Interface

The User Interface (UI), in the industrial design field of human-machine interaction, is the space where interactions between humans and machines occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' decision-making process. Examples of this broad concept of user interfaces include the interactive aspects of computer operating systems, hand tools, heavy machinery operator controls, and process controls. The design

considerations applicable when creating user interfaces are related to or involve such disciplines as ergonomics and psychology [22].

Generally, the goal of user interface design is to produce a user interface which makes it easy (self-explanatory), efficient, and enjoyable (user-friendly) to operate a machine in the way which produces the desired result. This generally means that the operator needs to provide minimal input to achieve the desired output and also that the machine minimizes undesired outputs to the human.

With the increased use of personal computers and the relative decline in societal awareness of heavy machinery, the term user interface is generally assumed to mean the graphical user interface, while industrial control panel and machinery control design discussions more commonly refer to Human Machine Interfaces.

Other terms for user interface are Man Machine Interface (MMI) and when the machine in question is a computer Human Computer Interface.

3.5 Human Factors

Human Factors (HF) is the scientific discipline concerned with the understanding of interactions among humans and other elements of a system and the application of theories, principles, data and other methods of design in order to optimize human well-being and overall system performance [23]. We take that as our charge, and note that we are especially concerned with the interaction of humans with computer systems, a domain generally known as Human-Machine Interaction (HMI).

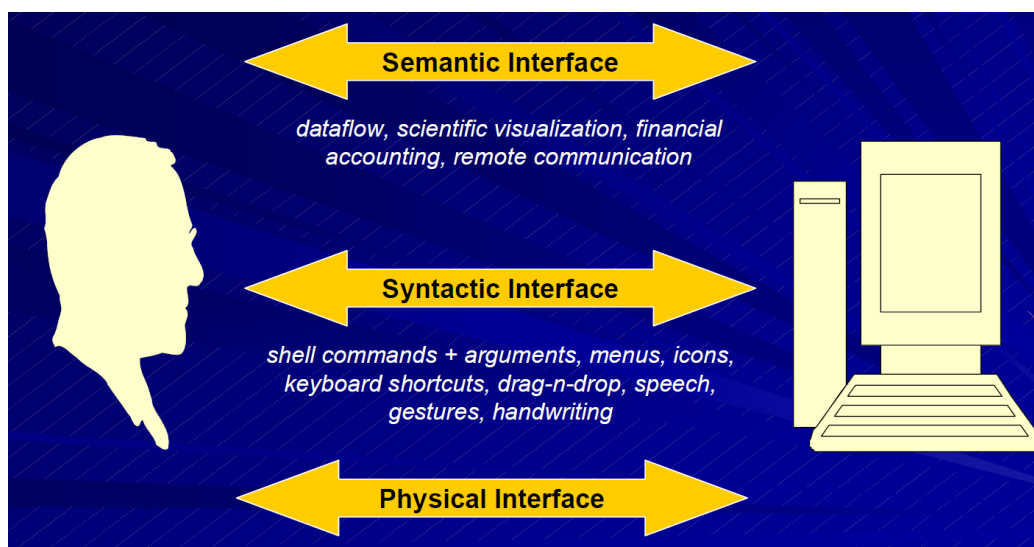


Figure 3 - Human Factors Interfaces [23]

The program of this work focuses on cognitive aspects of interaction (e.g., design of displays, procedures, and information) as opposed to physical aspects (e.g., design of keyboards and work stations).

These are the reflexive considerations:

- Intuition versus learning
 - simplicity VS completeness/efficiency
- Accuracy speed trade-off
 - ROC curve
- User feedback
 - visual, aural, tactile
- Motor-sensory channel separation
 - e.g. is gaze cursor control good interface?
- Ergonomics
 - Minimize physical strain on users

3.6 Home automation technologies

Home automation—or “smart home”—technologies are among the most promising Internet of Things applications that have the potential to improve life for persons with disabilities [24].

Home automation technologies utilize a variety of connected cameras and sensors to relay information to the user’s smart phone, allowing for remote monitoring and control of lights, thermostats, appliances, security systems and more.

Coupled with an accessible smartphone interface, smart home technologies hold enormous promise for persons with disabilities, improving quality of life and facilitating independent living [25]. The examples abound:

- For people who are blind or have low vision, home automation applications allow for easy control of appliances and the home thermostat, all with the touch of a button on a smart phone. It is crucial, however, that these applications be compatible with the smart phone’s screen reader or other accessibility features.
- For people with mobility-related disabilities, smart home technology holds the promise of allowing the user to control things in his or her home that may be physically difficult to reach, such as lights, door locks or security systems.

- For people who are deaf or hard of hearing, one benefit of home automation technology is improved security. A deaf person might not be able to hear a break-in, but a smart security system can help compensate for that by providing alerts through the user's phone in the event of suspicious movement picked up by a sensor outside the home.
- Additionally, smart home technology allows the children and caregivers of older adults to monitor their daily routines, promptly alerting children and caregivers if routine tasks are not performed.

Several providers already have smart home platforms on the market today. One such platform is AT&T's Digital Life, which offers the "classic" features including cameras inside and outside the home, remote temperature monitoring and control, remote home light control, remote water detection and control, coded door locks etc.

4 The Personalized Assisted HMI system

The PAHMI system has been developed to actively and dynamically assist the user to interact with surrounding environment.

The PAHMI system provides a HMI (§4.2) made of targets corresponding to commands which can be given to the domotic environment. The user, in order to interact with the environment, needs to be able to reach the HMI target by moving a cursor through a smartphone accelerometer (§4.3).

The PAHMI system developmental procedure consists of two phases: firstly, an algorithm based on automatic learning studies the user features while he/she interacts with the HMI (not assisted, §4.4); secondly, the model resulting from the learning phase is used by a control algorithm to assist the user in real time (assisted HMI, §6.3).

The learning algorithm, named Support Vector Machine (SVM), has the function of “multi-class” classifier (§5.1), while the controller is a PID (Proportional Integral Derivative) and it provides the corrective angular speed (§6.2), based on SVM calculations (§6.3).

Finally, in order to complete the PAHMI system, actuator systems, necessary to the execution of commands given to the environment, have been developed and implemented (§7.3).

4.1 Functioning principle

The system comprises an HMI (§4.2), presented to the user, made of a set of configurable squares placed in circle (targets) and a cursor (PC mouse); the cursor, properly modelled (§6.1), is subjected to a specific control (§6.2.1) in order to permit a facilitated reach of the target.

By using this system, the user, with the smartphone fixed on his/her head, has the chance to move the cursor on the HMI by only moving his/her head, until the target corresponding to the desired command is reached. In order to move the cursor, the PAHMI system uses the smartphone accelerometer; in this way, when the user moves the head, the smartphone moves and the accelerometer gives to the system data about the angular position of user’s head (§4.3).

The PAHMI system is divided between software application and hardware implementation.

As for the software, the PAHMI requires the use of MATLAB, the smartphone MATLAB application and of a server capable of manage, sort and give commands to the surrounding environment.

As for the hardware, it is implemented based on different communication protocols between server and software on-board of embedded system; those protocols are distinguished in WIFI (§7.3.1) and IR (§7.3.2).

The PAHMI activity flow is shown in the following Figure 4.

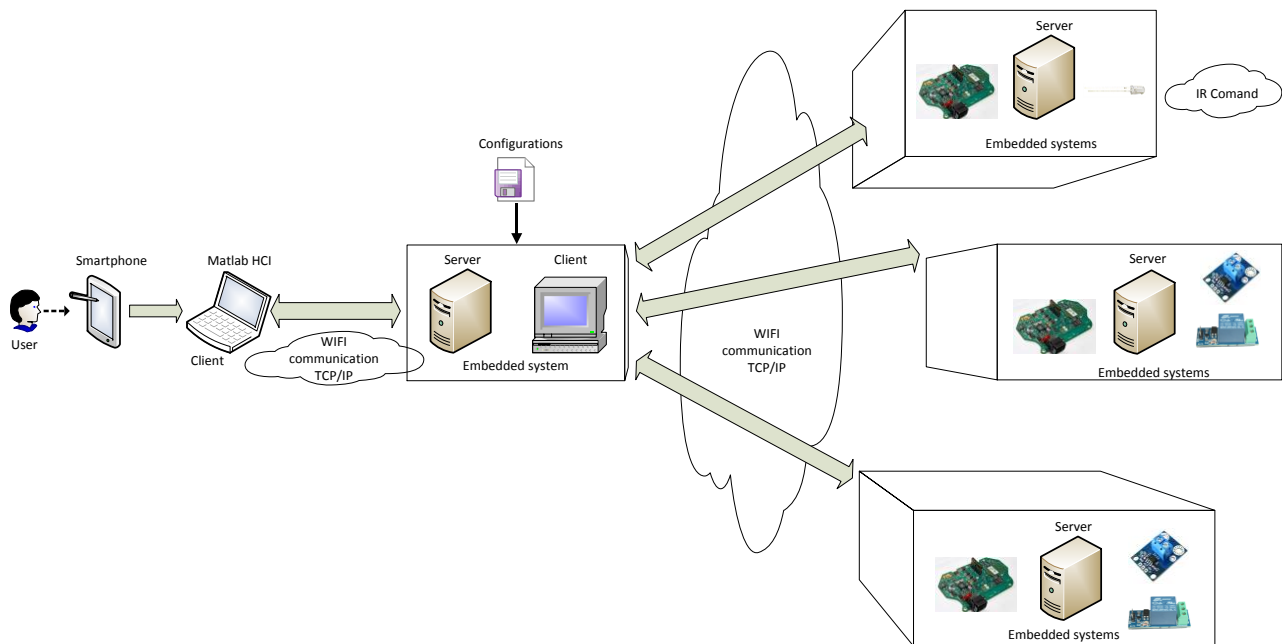


Figure 4 - PAHMI data flow

4.2 The HMI of the system

The main system interface, through which the user interacts with the environment, has been totally developed in MATLAB (© Mathworks); it is an HMI projected and developed with refer to Human Factors studies (§3.5).

The software developed in MATLAB manages a connection with the mobile MATLAB application installed on smartphone (§4.3); when the connection is established, the application sends to the software real time data from the smartphone accelerometer.

The HMI has a number of configurable squares, placed in circle. Those squares are the targets that the user need to achieve by piloting a cursor through the smartphone.

An example of HMI of 1920x1080 pixels resolution is shown in Figure 5.

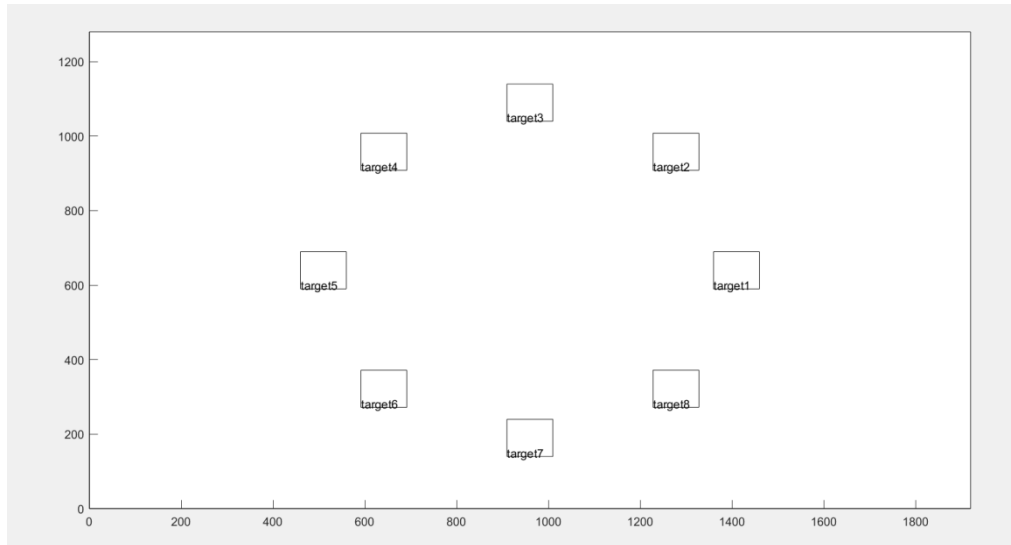


Figure 5 - HMI with 8 targets

In order to reach a target, the cursor has to be piloted until it crosses the target perimeter; to consider the target reached, the cursor has to stay within the square perimeter for a configurable number of seconds (default 2s). The target corresponds to a given command to the surrounding environment, such as switching on/off a light, changing channel on TV etc.

In Figure 8, each target has the label “target i ” with $i = 1, \dots, 8$; in the simulation paragraph (§8), it is possible to read that tags become corresponding to the effective command (switch on, power off etc.) according to the actual state of the surrounding environment.

4.3 The MATLAB mobile app of the system

MATLAB supports the acquisition of data from the built-in sensors on smartphone [26]. With the MATLAB Support Package for Mobile Sensors, it is possible to log data or query the most recent data available from the supported sensors on the smartphone. These include motion sensors like the accelerometer and position sensors like the GPS.

Measurements such as acceleration, magnetic field, latitude, longitude, and altitude can be viewed on the mobile device. It is possible to analyse the data in MATLAB and build applications that make decisions based on the sensor data. To use this package, it is necessary the installation of the MATLAB Mobile™ on the smartphone.

For Apple devices it is necessary to install on MATLAB the *iossensor.mlpkginstall* provided by Mathworks before starting the (Apple) mobile application.

The MATLAB Support Package permits to acquire and to log data for the following measurements:

- Acceleration on 3-axes
- Magnetic field on 3-axes
- Angular velocity on 3-axes
- Azimuth, roll, pitch
- Latitude, longitude, altitude, horizontal accuracy, speed, and course

In the Figure 6 is shown the MATLAB app user interface.

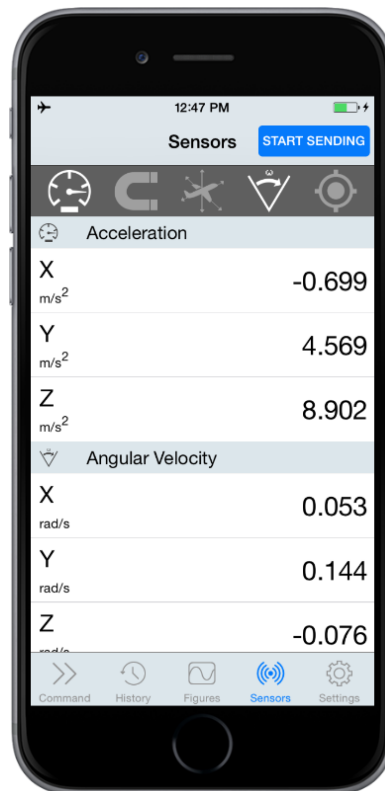


Figure 6 - MATLAB app user interface

4.4 Inclined plan force components

The cursor is controlled through data coming from the smartphone accelerometer, which need to be properly translated in order to be converted into cursor spatial coordinates. Data coming from smartphone orientation are roll and pitch angles (α, β), as shown in Figure 7.

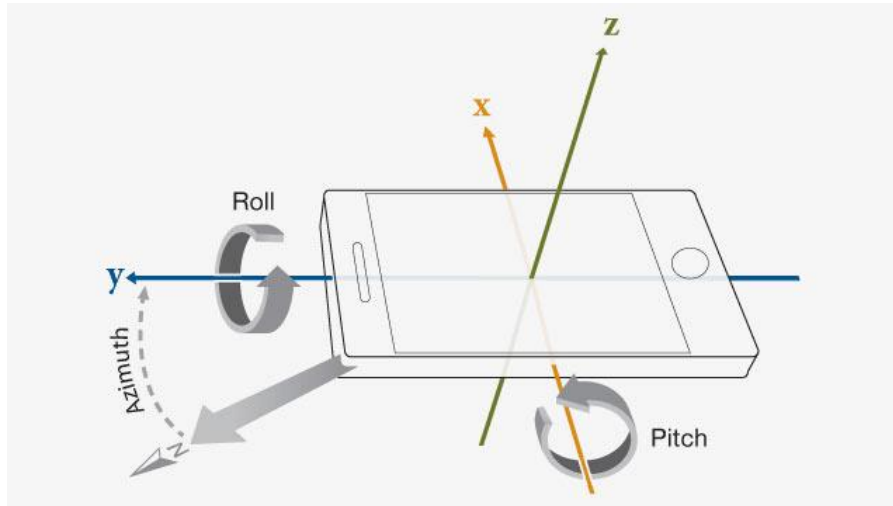


Figure 7 - Roll and Pitch angles of the smartphone orientation [26]

In particular, the cursor is modelled such a virtual ball placed in the center of the smartphone; for the virtual ball, the smartphone acts like an inclined plane [27] with the same dimensions of the HMI (i.e. 1920x1080). Hence, when smartphone roll and pitch angles (α, β) are modified and the virtual ball plane is inclined, the cursor (just like the ball) begins to move.

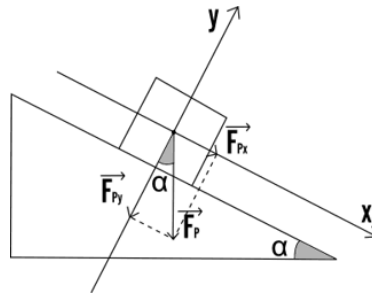


Figure 8 - Resultant forces of an inclined plane

To calculate the virtual ball position s on the inclined plane, the following law of motion [28] can be used:

$$s = s_0 + v_0 t + \frac{1}{2} a t^2 \quad (1)$$

The position and speed initial conditions (s_0, v_0) are the initial conditions given to the law of motion (1), while the acceleration a , considering the unified mass of the virtual ball, is calculated as follows:

$$a = g \cdot \sin(\alpha) \quad (2)$$

where g is gravity and α is the roll angle.

The law of motion is the same both for plane inclined by the α roll angle and for the plane inclined by the β pitch angle, thus values s, s_0, v_0, a have to be considered ad couple of values $(s_x, s_y), (s_{0x}, s_{0y}), (v_{0x}, v_{0y}), (a_x, a_y)$.

In order to avoid sharp movement of the cursor on the HMI, it is important to filter accelerometer data (α, β) before starting with the calculation of the couple of values (s_x, s_y) . The filter chosen for this purpose is the second order Butterworth filter (§4.5) with a cutting frequency $\omega_c = 0.5Hz$.

Afterwards, it is necessary a coordinate transformation (s_x, s_y) in pixel (p_x, p_y) .

This transformation is made through the proportion between monitor resolution (in pixel) and the HMI interface dimension (in coordinates). The result (m_x, m_y) is the final cursor position in pixel:

$$\begin{aligned} m_x &= s_x \cdot \frac{sc_x}{ampRect} \\ m_y &= s_y \cdot \frac{sc_y}{altRect} \end{aligned} \tag{3}$$

Where *ampRect* and *altRect* are, respectively, the amplitude and the height in coordinates of the interface, while (sc_x, sc_y) is the monitor resolution (i.e. 1920x1080).

The calculation of the cursor position has to be made frequently; indeed, cursor position on HMI is updated at each sampling stage. This sampling frequency is configurable, by default 10Hz.

Each target corresponds to a command which can be given to the surrounding environment; hence, by using this method of data transduction, the user has to be able to incline the smartphone in order to move the cursor into the desired target and give the desired command among those proposed.

4.5 The Butterworth filter

To filter data received from the smartphone orientation, it is used a simple and particularly useful class of filters referred to as *Butterworth* filters [29], [30].

Filters in this class are specified by two parameters, the cut-off frequency and the filter order. The frequency response of these filters is monotonic and the sharpness of the transition from the passband to the stopband is dictated by the filter order. For continuous-time Butterworth filters,

the poles associated with the square of the magnitude of the frequency response are equally distributed in angle on a circle in the s-plane, concentric with the origin and having a radius equal to the cut-off frequency. When the cut-off frequency and the filter order have been specified, the poles

characterizing the system function are readily obtained. Once the poles are specified, it is straightforward to obtain the differential equation characterizing the filter.

The design of a discrete-time filter through the use of the impulse-invariant design procedure is applied to a Butterworth filter. The filter specifications are given in terms of the discrete-time frequency variable and then mapped to a corresponding set of specifications for the continuous-time filter. A Butterworth filter meeting these specifications is determined. The resulting continuous-time (CT) system function is then mapped to the desired discrete-time (DT) system function.

A limitation on the use of impulse invariance as a design procedure for discrete-time systems is the requirement that the continuous-time filter be bandlimited to avoid aliasing. An alternative procedure, called the *bilinear transformation*, corresponds to a mapping of the entire imaginary axis in the s-plane to once around the unit circle. Consequently, there is no aliasing introduced by this procedure. However, since the imaginary axis has infinite length and the unit circle has a finite circumference, by necessity there must be a nonlinear distortion in mapping between the two frequency axes. The use of the bilinear transformation is therefore limited to mapping filters that are approximately piecewise constant. For filters of this type, the inherent nonlinear distortion in the frequency axis is easily accommodated by preparing the critical frequencies (e.g., passband edge and stopband edge) prior to carrying out the design of the associated continuous-time system. This procedure is illustrated with the design of a Butterworth filter.

The Butterworth filter is used to remove all high order frequencies from the inputted signal demonstrated in Figure 9.

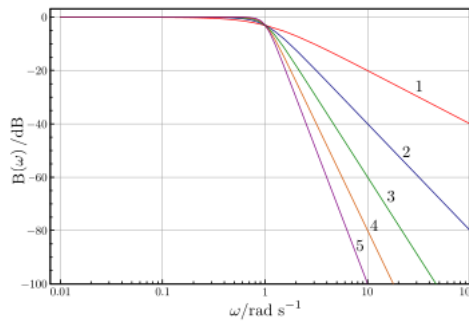


Figure 9 - Magnitude response of an ideal n^{th} -order Butterworth filter [29]

The frequency response for the class of Butterworth filters is:

$$|B(j\omega)|^2 = \frac{1}{1 + (j\omega/j\omega_c)^{2N}} \quad (4)$$

where $\omega = 2\pi f$ is the radian frequency, $\omega_c = 2\pi f_c$ is the constant scaling frequency and N is the order of the filter.

Function $B(s)B(-s)$ has $2N$ poles and doesn't have any finite zeros. It is easy to see that if s_i is a pole of (4), then $-s_i$ is also a pole of (4).

The poles of $B(s)B(-s)$, that have negative real parts, are:

$$s = (-1)^{1/2N} j\omega_c \quad (5)$$

The practical low-pass filter specification is determined by four parameters: $A_p, \omega_p, A_s, \omega_s$, that correspond, respectively, to pass-band attenuation, pass-band radian frequency, stop-band attenuation and stop-band radian frequency. The first step to design a filter with these parameters is to determine the minimum order of the filter that meets this specification. The signal attenuation for the Butterworth approximation can be expressed as follows

$$A(\omega) = -20\log_{10}(|B(j\omega)|) = 10\log_{10}(1 + (j\omega/j\omega_c)^{2N}) \quad (6)$$

Applying (6) to the pass-band and stop-band edges results in the following system of two equations

$$A_p = 10(1 + (\omega_p/\omega_c)^{2N}) \quad (7)$$

$$A_s = 10(1 + (\omega_s/\omega_c)^{2N})$$

These Equations (7) can be rearranged as

$$(\omega_p/\omega_c)^{2N} = 10^{0.1A_p} - 1 \quad (8)$$

$$(\omega_s/\omega_c)^{2N} = 10^{0.1A_s} - 1$$

Variables N and ω_c must be obtained from system (8). The order of the filter N is an integer variable and scaling frequency ω_c is the real variable. Parameters $A_p, \omega_p, A_s, \omega_s$ are real and they are set in the specification. In general, system of Equations (8) doesn't have a precise solution for those kinds of variables but it can be solved if integer variable N is replaced with the real variable X .

Minimum order of the Butterworth filter that meets specification can be found as

$$N = [X] = \left\lceil \frac{\log\left(\frac{10^{0.1A_p} - 1}{10^{0.1A_s} - 1}\right)}{2\log\left(\frac{\omega_p}{\omega_s}\right)} \right\rceil \quad (9)$$

where $[.]$ is the nearest integer operator.

Cut-off frequency for the ideal low-pass filters is determined as a frequency up to which the signals pass well and the signals beyond it are rejected. Due to the transition band in the practical filters, the definition of cut-off frequency is unclear. It is a common practice to characterize practical filters with a natural cut-off frequency, which is determined as a frequency at which the power gain is 1/2 of the power gain at the passband frequencies. Since the power of the signal is proportional to the amplitude, the magnitude response at the natural cut-off frequency is as follows

$$|H(j\omega)| = \frac{1}{\sqrt{2}} \quad (10)$$

Converting gain to decibels results in

$$A_N = -20\log_{10}(|H(j\omega)|) = 10\log_{10}(2) \approx 3dB \quad (11)$$

Therefore, the natural cut off frequency ω_{cut} can be determined as a frequency where the signal loss through the filter is approximately 3dB.

It is easy to see from (4), that for Butterworth filters, the natural cut-off frequency $\omega_{cut} = \omega_c$, and it doesn't depend on the order of the filter.

The minimum required order for the Butterworth filter could be computed using (9). The scaling frequency ω_c cannot be determined to precisely satisfy both of the edge conditions (8). The scaling frequency can be determined in such a way as to satisfy the system of inequalities

$$\begin{aligned} A_p &\geq 10\log_{10}\left(1 + (\omega_p/\omega_c)^{2N}\right) \\ A_s &\leq 10\log_{10}(1 + (\omega_s/\omega_c)^{2N}) \end{aligned} \quad (12)$$

where the order of the filter N is determined from (9). If these inequalities are satisfied, then Butterworth filter meets or exceeds the specification requirements.

For the PAHMI system, the data that have to be filtered are the acceleration data resulted from the transduction (§4.4) of the orientation smartphone data.

Given $A_p = 0.5dB$, $\omega_p = 0.5 \frac{rad}{s}$, $A_s = 1.5dB$, $\omega_s = 1.5rad/s$, the design of the Butterworth filter is a second order filter with the following transfer function:

$$B(s) = \frac{0.7322728}{s^2 + 1.210184s + 0.7322728} \quad (13)$$

In the Figure 10 it is shown a block schema of the acceleration data $a_c(t)$ transformation due to the filtering ($y_c(t)$).

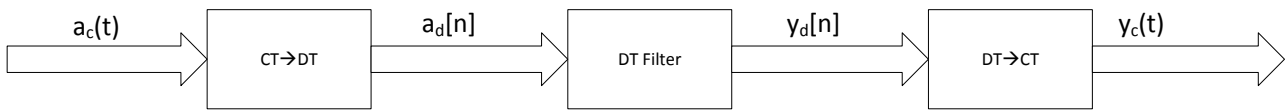


Figure 10 - Mutation of the acceleration data through the Butterworth filter

5 Learning phase

To realize an effective assisted HMI, it has to be tailored to the specific users' characteristics. Therefore, a preliminary tuning phase is needed, during which the user utilizes the non-assisted HMI. The result of this first phase is a collection of trajectories, sampled with a given frequency $f_s = 1/T_s$, where T_s is the sample time. The samples are then organized to build a data set. As detailed in 5.3, each trajectory toward a given target is divided into segments made of a given number of samples and each segment constitutes an input vector for the data set; the trajectory target constitutes the associated output.

As described in 5.4, the collected couples (*input,output*) are then used to train and validate a set of SVMs, each of which is associated to a different target [31]. The SVMs used constitute the multi-class classifier. On the ground of the validation of the SVMs, the so-called confusion matrices are defined, which give indications on the accuracy of the multi-class classifier (§5.1).

5.1 Multi-class classification

Multi-class classification has the objective of building a function $f(\mathbf{x})$ whose result is the classification of instances \mathbf{x} into more than two classes: based on a set of instances (the so-called *training* set), each of which belonging to one among M classes, the classifier must be able to predict which set a new instance belongs to [32].

There are algorithms [33] which natively classify more than two classes, but also binary classification algorithms (i.e., algorithms which classify instances into two classes, positive or negative) can be used by following different strategies. Referring to the case of M classes, the basic strategies are:

- *One vs. all* classification, in which M binary classifiers $f_m(\mathbf{x})$, $m = 1, \dots, M$ are built, one per each class. The m -th classifier considers the instances of the m -th class as positives and the other instances as negatives, i.e., $f_m(\mathbf{x}) = 1$ if \mathbf{x} belongs to the m -th class, $f_m(\mathbf{x}) = 0$ otherwise. The results of the binary classifiers are then interpreted as an M -class classification via heuristics, e.g., by computing $f(\mathbf{x}) = \operatorname{argmax}_m f_m(\mathbf{x})$.
- *All vs. all* classification, in which $M(M - 1)/2$ binary classifiers $f_{ij}(\mathbf{x})$ are built, one per each couple (i, j) , $i = 1, \dots, M, j = i + 1, \dots, M$. A new instance is classified as belonging to class m if it “wins” in the majority of classifiers, i.e., $f(\mathbf{x}) = \operatorname{argmax}_i \sum_{j=i+1, \dots, M} f_{ij}(\mathbf{x})$.

The use of native multi-class classifier, of the *one vs. all* strategy or of the *all vs. all* strategy is dependent on the application and, often, they are equivalent in practice. For the PAHMI system, a *one vs. all* strategy with Support Vector Machine (SVM) binary classifier has been adopted, which proved to be suitable for our application.

The results of the M binary classifier are put together by means of the so-called confusion matrices. Each classifier $f_m(\mathbf{x})$ is fed by a set of instances (the so-called *test-set*), different from the training set. The following data are collected in the confusion matrix of the m -th classifier: the number of *true positives* TP_m (the predicted class was m and the actual class was m), the *false negatives* FN_m (the predicted class was not m and the actual one was m), the *false positives* FP_m (the predicted class was m and the actual one was not m) and the *true negatives* TN_m (the predicted class was not m and the actual one was not m).

The collected data can be interpreted in different ways. Here, it is used the *positive predictive value* (PPV), which is a value between 0 and 1 computed as $PPV_m := TP_m / (TP_m + FP_m)$, representing the precision of the binary classifier. Therefore, given an instance x , each classifier m returns a value p_m equal the PPV_m if $f_m(\mathbf{x}) = 1$, or equal to 0 if $f_m(\mathbf{x}) = 0$: p_m is interpreted as the probability that the instance x belongs to class m .

5.2 SVM learning technique

SVM is a learning technique based on the statistical learning theory introduced by [34]. In the basic formulation, the classification problem is formulated as quadratic programming (QP) problem [35], with quadratic cost function and linear constraints, which solves the problem of classifying two linearly separated sets. If the instances of the data sets are not linearly separable, the SVM transforms the original input space to a higher dimensional one, called ‘feature space’ [36], by means of mapping functions, such as polynomial or radial basis functions. The classification problem is then solved in the feature space, where the optimal separation hyperplane is computed and then the solution is *translated* into the input space. The key element is that the explicit computation of the mappings between input and feature spaces are not required, since the distances between points in the feature space are computed via the *kernel* functions, e.g., the Gaussian RBF (GRBF) or the Exponential RBF (ERBF) kernels.

Several techniques exist for the validation of SVM. For the development of the PAHMI system, it is chosen a standard k -fold cross-validation. The whole input set is randomly partitioned into k subsets with equal cardinality; one out of the k subsets is used as a test set, whereas the union of the

other subsets is used as training set. The described validation process is repeated k times; each time a different subset is used as test set.

5.3 Data collection

To render the assistance effective, the user must preliminarily tune the HMI, with the aim of learning the characteristics of the user's trajectories.

Then, the user has to use the HMI without assistance for a given amount of times. This tuning phase consists in completing n_{task} times the task of controlling the cursor to direct it from the centre of the operational plan to each of the targets (in total, the user has to complete $M \cdot n_{tuning}$ tasks) [37], [38].

Each completed trajectory is sampled with a given sample time T_s . Given a trajectory t , its samples are split into segments of n_s samples. Note that if the number of samples $samples(t)$ of a trajectory t is not divisible by n_s , the appropriate number of initial samples is discarded. In fact, as further clarified in 5.4, the initial samples at the centre of the operational plan are the least significant.

In conclusion, given a trajectory t of length $samples(t)$, the segments obtained as described above constitute input vectors for the multi-class classifier. Let the number of segments obtained from trajectory t be denoted with $n_{seg}(t) = \lfloor samples(t)/n_s \rfloor$, where $\lfloor \cdot \rfloor$ is the lower integer operator. Let the j -th segment s_j be defined as the vector collecting the last n_s samples, up to the j -th one: $s_j := (s(j - n_s + i))_{i=1, \dots, n_s}$, where $s(k)$ is the i -th sample of the segment j , sampled at time $j \cdot T_s$ and $s(k) = (s_x(k), s_y(k))$ is the vector of the coordinates on the operational plan of the k -th sample. The target of the trajectory t , denoted as $target(t) \in [1, M]$, constitutes the output associated to each of the n_t segments obtained from t .

When the HMI tuning is completed, a data base is obtained containing $n_{db} = \sum_{t=1, \dots, (M \cdot n_{task})} n_{seg}(t)$ couples (trajectory segment, trajectory target). Thanks to the segmentation of trajectories, $n_{seg}(t)$ (*input, output*) couples needed to train the multi-class classifier (§5.1) are obtained from a single trajectory t ; this approach is then effective in reducing the number of tasks n_{task} that the user must accomplish to tune the HMI.

5.4 SVM training, validation and confusion matrices

The multi-class classifier is constituted by a set of M SVMs, in such a way that the m -th SVM is associated to the m -th target, $m = 1, \dots, M$. The couples segments/target trajectory identified during the tuning phase are used to train and evaluate the SVMs.

Each SVM serves as a binary classifier: the SVM m has to decide whether the trajectory segment is part of a trajectory directed toward its target m or not.

Firstly, the so-called *features* have to be extracted from the available inputs [39], i.e., parameters which are indicative of the fact that the trajectory segment is directed towards target m .

Let the coordinates of target m on the operational plan be denoted with (m_x, m_y) , and let the distance between the i -th sample to target m be denoted with $dist(s(i), m) = \sqrt{(m_x - s_x(i))^2 + (m_y - s_y(i))^2}$. Here, considering the SVM m , the following features are extracted from the segment s_j of trajectory t :

- the distance from the first sample of segment s_j to the target m :

$$feat_1^m(s_j) = dist(s(j - n_s + 1), m) \quad (14)$$

- the average velocity component towards target m , normalized with respect to $n_s \cdot f_s$:

$$feat_2^m(s) = (dist(s(j), m) - dist(s(j - n_s + 1), m)) \quad (15)$$

- the standard deviation of the velocity component towards target m :

$$feat_3^m(s) = std_dev\left(\left(dist(s(j - n_s + i), m) - dist(s(j - n_s + i - 1), m)\right)_{i=2, \dots, n_s}\right) \quad (16)$$

The first feature aims at differentiating the trajectory segments based on their distance to the target; in fact, as further remarked in the following, closer segments are more significant with respect to farer ones. The second feature aims at identifying the direction of the segment with respect to the target. The third feature aims at identifying oscillating behaviours, which might represent unwanted cursor movements.

If needed, different and/or more features might be extracted (e.g., acceleration towards target m).

In the end, the n_{db} couples (feature vector, trajectory target) contained in the data base are used by the SVM m as follows (we explicitly note that, by the proposed procedure, each one of the M SVM has n_{db} (input, output) couples):

- each feature vector is an input vector of length 3, whose elements are the features defined above;
- the associated output is 1 if the trajectory target is m , 0 otherwise.

Each SVM m performs a standard binary clustering task, since it has to classify the input vectors as trajectory segments directed toward target m or not, by tagging them with 1 or 0, respectively. A standard k -fold cross-validation procedure is followed to build the SVMs [40].

The SVM m is likely to classify the segments of trajectories which are closer to target m with better accuracy with respect to the far segments: e.g., at the beginning of a task, the first trajectory samples are in the centre of the operational plan and thus they give few hints on the target the trajectory is aimed at; on the other hand, the last trajectory samples are close to or even within the area of the target m . Therefore, it is adopted an input stratification procedure, and it is divided the n_{db} (input,output) couples versus the distance between the first sample of the trajectory and the target. With reference to the segment s_j of the SVM m and to a threshold value d_{th} (which is a parameter of the proposed classifier), if $feat_1^m(s_j) \leq d_{th}$ the segment s_j is collected in the so-called *near segment set*; if $feat_1^m(s) > d_{th}$ the segment s_j is collected in the so-called *far segments set*.

Each SVM $m, m = 1, \dots, M$, is then tested against the segments of the two sets and the resulting prediction of the tests are then used to compute *two* confusion matrices, denoted as *far confusion matrix* and *near confusion matrix*, respectively, from which the *near PPV* and *far PPV* of the near and far tests are computed.

The set of SVMs constitutes the multi-class classifier: given a trajectory sample of n_s segments, each SVM m returns its prediction about the target that trajectory is aiming at. Then, the precision of the SVM results can be evaluated by using the PPV values, which are computed for far and near segments, as explained above. Details on how the multi-class classifier is exploited in the HMI are given in 6.3.

6 The controlled system

To help user to reach the desired target and to assist the mouse trajectory, it is studied a navigation control algorithm that enables the mouse, modelled like an ideal unicycle as a Differential Drive Wheeled Mobile Robot (DDWMR), to accomplish its assigned task and to move to the goal [41], [42], [43].

In order to develop this navigation system, a low-level planning is used, based on a simple ideal unicycle model whose input can be calculated using a PID [41], [5], [6], [44].

Finally, as described in §6.3, the multi-class classifier is used to design the assistance to the user while she/he is interacting with the HMI. The control action, i.e., the assistance, is computed on-line, with frequency f_s , and its aim is to let the target predicted by the multi-class classifier “attract” the cursor; the strength of the attraction depends on the accuracy of the prediction, as evaluated on the ground of the confusion matrices.

6.1 Kinematic model

The DDWMR setup is shown in Figure 11. The unicycle is made up of a rigid body and non-deforming wheel and it is assumed that the vehicle moves on a plane without slipping, i.e., there is a pure rolling contact between the wheel and the ground [41], [45].

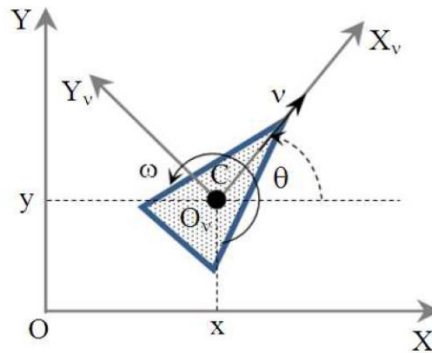


Figure 11 - Coordinates for differential drive wheeled mobile robot [41]

The configuration of the unicycle is represented by the generalized coordinates $q = (x, y, \theta)$, where (x, y) is the position and θ is the orientation (heading) of the axis centre of the wheel, C , with respect to a global inertial frame $\{O, X, Y\}$. Let $\{O_v, X_v, Y_v\}$ be the unicycle frame. The unicycle's velocity is by definition v in the unicycle's x -direction and ω is the heading rate.

The kinematic model of a unicycle is used for the purpose of implementation, which corresponds to a single upright wheel rolling on the plane, with the equation of motion given as:

$$\begin{aligned}\dot{x} &= v \cos \theta \\ \dot{y} &= v \sin \theta \\ \dot{\theta} &= \omega\end{aligned}\tag{17}$$

The inputs of Equation (17) are v and ω .

A Simulink model shown in Figure 12 has been developed to implement the unicycle kinematic model [46].

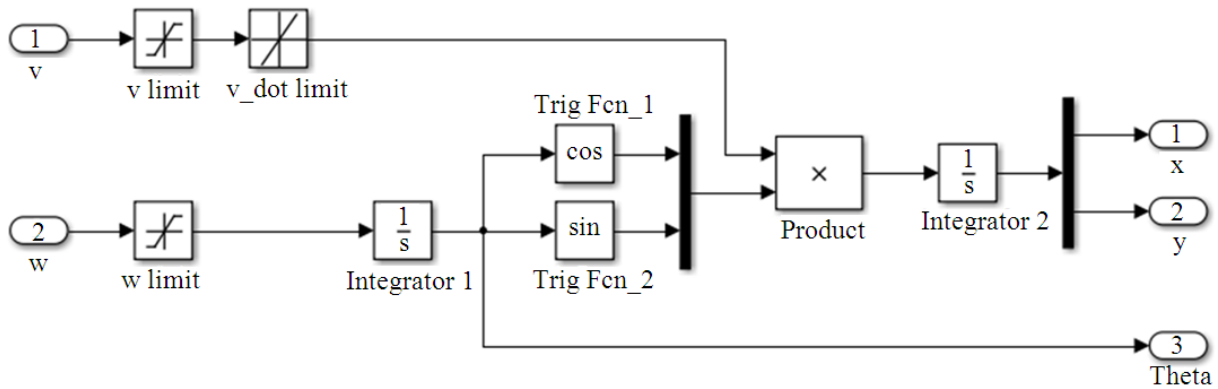


Figure 12 - Simulink block for the unicycle kinematic model

The velocity input has a rate limiter to model finite acceleration and limiters on the velocity and the heading or turn rate.

6.2 Control algorithm

Control of the unicycle model inputs is about selecting the appropriate input, $u = (v, \omega)^T$ and applying the traditional PID-feedback controller [9], [43], given by Equation (18):

$$u(t) = PID(e) = K_p e(t) + K_I \int_0^t e(\tau) d\tau + K_D \frac{de(t)}{dt}\tag{18}$$

where, e is the error between the desired value and the output value, K_p is the proportional gain, K_I is the integrator gain, K_D is the derivative gain and t is time. The control gains used in this research are obtained by tweaking the various values to obtain satisfactory responses. The unicycle is driven at a constant velocity, $v = v_0$ then the control input will only vary with the angular velocity, w , thus:

$$\omega = PID(e) \quad (19)$$

6.2.1 Moving to point

Consider a unicycle moving toward a goal point, (x_g, y_g) , from a current position, (x, y) , in the xy-plane [43], [5], as depicted in Figure 13.

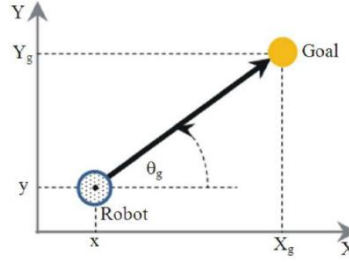


Figure 13 - Coordinates for moving to a point [43]

The desired heading (unicycle's relative angle), θ_g , is determined as:

$$\theta_g = \tan^{-1}\left(\frac{y_g - y}{x_g - x}\right) \quad (20)$$

And the error, e , is defined in the Equation (21)

$$e = \theta_g - \theta \quad (21)$$

To ensure $e \in [-\pi, \pi]$, a corrected error, e' is used instead of e as shown below in the Equation (22):

$$e' = \tan^{-1}(\sin(e), \cos(e)) \in [-\pi, \pi] \quad (22)$$

Thus ω can be controlled using the Equation (18). If the unicycle's velocity is to be controlled, a proportional controller gain, K_V , is applied to the distance from the goal, shown below (Corke, 2011) Equation (23):

$$v = K_V \sqrt{(x_g - x)^2 + (y_g - y)^2} \quad (23)$$

6.3 Real time computer assisted control

The PAHMI system assists the user in real time while he/she tries to impart his/her desired command.

The block diagram of the entire system is depicted in Figure 14.

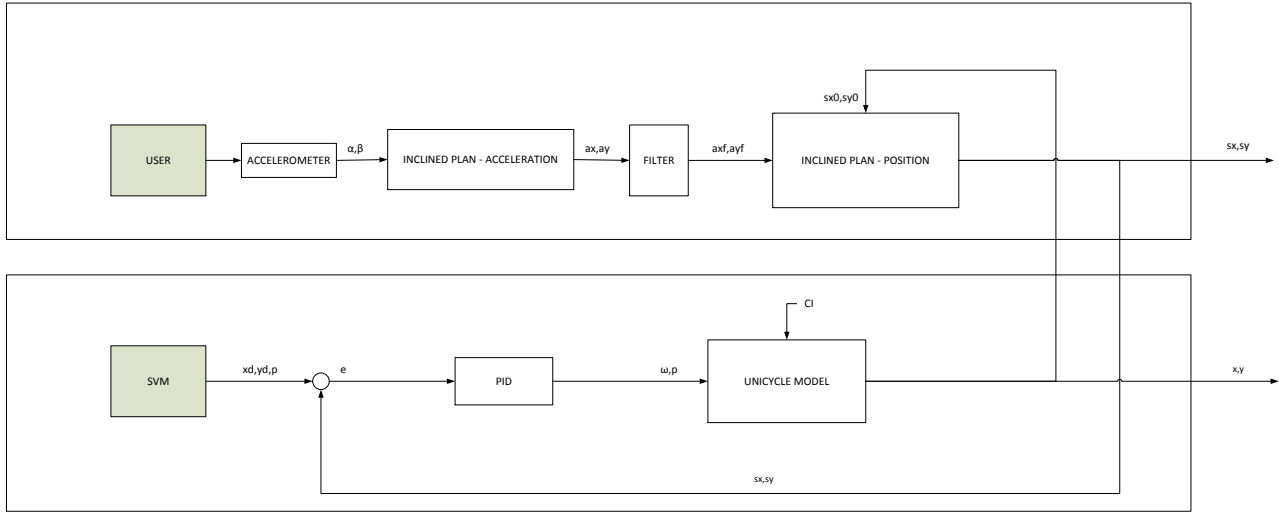


Figure 14 - Real time assisted control

As depicted in the Figure 14, the system is divided into two parts, the first one concerns the transformation of the data coming from the user (smartphone orientation angles) and the second one concerns the control of such data [6], [44], [47].

The PAHMI input data coming from the user are the roll and pitch angles of the smartphone inclination (α, β) , used to calculate the acceleration of the cursor with respect to HMI Cartesian axis (see an example of HMI in Figure 5).

The accelerations along the axis are calculated with the following Equations (24):

$$\begin{aligned} a_y &= g \cdot \sin(-\alpha); \\ a_x &= g \cdot \sin(\beta); \end{aligned} \quad (24)$$

The user starts moving the cursor and the trajectory is sampled with sample time $T_s = 0.1s$. Subsequently, the resulting accelerations ax, ay are suitably filtered using a Butterworth filter with the aim of eliminate abrupt variations of the cursor acceleration. The result is the couple (a_x^f, a_y^f) of acceleration components along the axis of the HMI.

The last block of the discrete system is the transduction of the accelerations in the s_x, s_y coordinates of the cursor calculated with the Equations (25) and (26) of the rectilinear uniform motion law:

$$\begin{aligned} v_x &= (a_x^f) \cdot T_s \\ v_y &= (a_y^f) \cdot T_s \end{aligned} \quad (25)$$

$$\begin{aligned} s_x &= s_{x_0} + v_x \cdot T_s \\ s_y &= s_{y_0} + v_y \cdot T_s \end{aligned} \quad (26)$$

In order to calculate the position is necessary to calculate the velocity (v_x, v_y) of the cursor from the acceleration (25). Knowing the velocity and the initial position, it is possible to calculate the final position at each sampling instant (26).

The (s_{x_0}, s_{y_0}) initial position corresponds to the centre of the HMI only at the first sampling instant, subsequently, it corresponds to the position of the cursor (s_x, s_y) at the previous sampling instant.

The sampled couple of values (s_x, s_y) is compared with the results obtained by the SVM during the learning phase (the multi-class classifier is set-up, §5.1); these SVM results provide the desired target (x_d, y_d) and the probability $0 \leq p \leq 1$ of achieving it [48]. The actual real time control occurs at each sampling instant (§6.2.1).

After the first $n_s - 1$ samples, the assistance starts acting on the cursor. Let j be the last trajectory sample, with $j \geq n_s$, and let s_j be the segment made by the last n_s samples: $s_j = (s(j - n_s + 1), \dots, s(j))$. The HMI computes the (14), (15) and (16) features of the segment and feeds the multi-class classifier with the obtained input vector. The M SVMs return their prediction about the target the user is aiming at (each SVM m returns 1 if the input vector is classified as a segment of trajectory aimed at target m). The accuracy prediction of each SVM m is evaluated thanks to the confusion matrices; the *far confusion matrix* or the *near confusion matrix* is considered if the distance of the j -th trajectory sample to target m (i.e., to $feat_1^m(s_j)$) is larger or lower with respect to the threshold value d_{th} , respectively.

The next position of the cursor is then modified by the Assisted HMI as follows. Considering that the last position of the cursor on the operational plan was (s_{x_0}, s_{y_0}) , let (s_x, s_y) denote the cursor position at time $j \cdot T_s + i \cdot T_i, i \in \{0, 1, \dots, n_i - 1\}$ which the cursor would reach under the sole user control. The actual next positions in the time interval $[j \cdot T_s, j \cdot T_s + (n_i - 1) \cdot T_i]$ are shifted in the position:

$$s_{HCI}(j) = (x, y), j = 0, 1, \dots \quad (27)$$

where $s_{HCI}(j)$ is the position correction given by the assistance, computed at time $j \cdot T_s$.

If the SVMm returns a probability $0 < p \leq 1$, the meaning of the control action is that, during the j -th sampling period, the position of the cursor under the sole user control is modified and the next cursor position is shifted in the direction of the targets which have the best estimates at time $j \cdot T_s$.

The current (s_x, s_y) couple of values of the cursor is compared with the desired target position (x_d, y_d) obtained by SVMm. The resulting error e becomes the input of the PID controller, which controls the unicycle model having $v_{x_0} = v_y = 200px/s$ and $\theta_g = 0$ as initial conditions [49]. The output of the control is the angular velocity ω calculated as described in the following Equations (28), (29) and (30):

$$\begin{aligned} u_y &= y_d - s_y; \\ u_x &= x_d - s_x; \\ \theta_g &= \tan^{-1}(u_y, u_x); \\ e &= \theta_g - \theta_0; \\ e &= \tan^{-1}(\sin(e), \cos(e)); \end{aligned} \tag{28}$$

$$\begin{aligned} \dot{x} &= v_x \cdot p \cdot f \cdot \cos(\theta) \\ \dot{y} &= v_y \cdot p \cdot f \cdot \sin(\theta) \\ \dot{\theta} &= \omega \end{aligned} \tag{29}$$

$$\begin{aligned} integral &= integral + e \cdot dt; \\ derivative &= (e - previous_error)/dt; \\ \omega &= Kp \cdot e + Ki \cdot integral + Kd \cdot derivative; \end{aligned} \tag{30}$$

Where Kp, Ki, Kd are the controller gains and *integral* and *derivative* values are calculated from the error e at the instant dt and from the previous error *previous_error* calculated at the previous sampling instant dt . Clearly, the *previous_error* value at the first sampling is initialized to 0.

Gain values, considered suitable for the control, are the following:

- $Kp = 8000$
- $Ki = 10$
- $Kd = 0.1$

The f parameter value (Equation (29)) is equivalent to a value between 0 and 1. This value assumes the meaning of the damping speed of the cursor (v_x, v_y) so that the control is always a contribution of the cursor guided assistance lower than the desired movement set by the user; in this way, the user has always the possibility to set the cursor in the desired direction and the control is never dominant. The contribution of the control, and then the f value, is inversely proportional to the distance between the cursor and the desired target obtained by the SVM m .

As for the probability value p , it is necessary to consider [50]:

- if the SVM m predicts that the trajectory is not aimed at the target m , the coefficient p is null;
- if the SVM m predicts that the trajectory is aimed at the target m and $feat_1^m(s_j) > d_{th}$, the coefficient p is equal to the *far PPV* value of the m -th SVM;
- if the SVM m predicts that the trajectory is aimed at the target m and $feat_1^m(s_j) \leq d_{th}$, the coefficient p is equal to the *near PPV* value of the m -th SVM.

The value p is the highest value of the probabilities vector of each target achievement returned by the SVM m . Therefore, p is a scalar and is a further contribution to the gain control damping.

Since the near segments are likely to be estimated with better accuracy, the parameter p computed from the *near PPVs* is larger than the one computed from the *far PPVs*, thus increasing the magnitude of the assistance as the cursor gets closer to a target.

For instance, in case of a segment at the instant $j \cdot T_s$ for which i) two SVMs return a positive prediction, ii) $feat_1^m(s_j) \leq d_{th}$, and iii) the m -th *near PPV* returns 0.9 as *true positive* value; the PAHMI shifts the user-controlled position (s_x, s_y) with a $\omega rad/s$ obtained by the control to reach the predicted desired target, scaled by 0.9, obtaining the final position (x, y) at the instant $(j + 1) \cdot T_s$.

The output of the final model is the position (x, y) of the cursor resulted from the user and the control contributions. The couple (x, y) becomes the initial value (s_{x_0}, s_{y_0}) at the immediate next sampling time.

In the Figure 15 is depicted the block diagram showing the inputs, the outputs and the main relationships (in pseudo code) of each sub system.

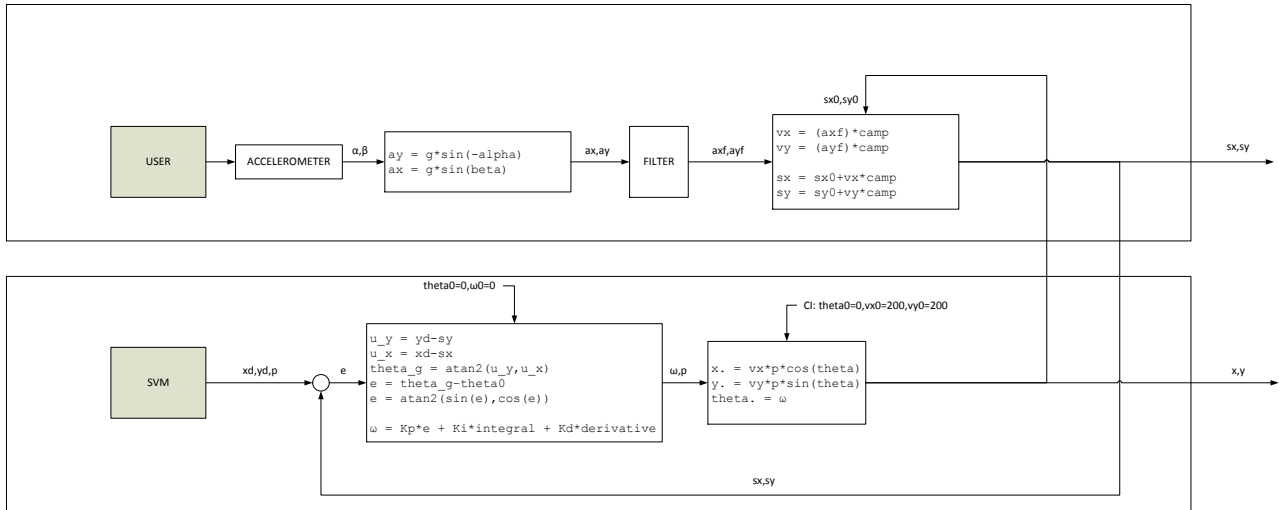


Figure 15 - Detailed real time assisted control

At the generic sampling instant $j \cdot T_s$, if the probability to achieve a target is equal to 0 for all the targets, the control has no impact on the user's decision, therefore, the position of the cursor remains the one imposed only by the user (s_x, s_y) .

7 Hardware and software implementation

The PAHMI main software communicates with the MATLAB software and sorts the commands to the respective actuators.

In the Figure 16 is shown a sequence diagram in which is depicted the requests and the responses from the HMI to the actuator [51].

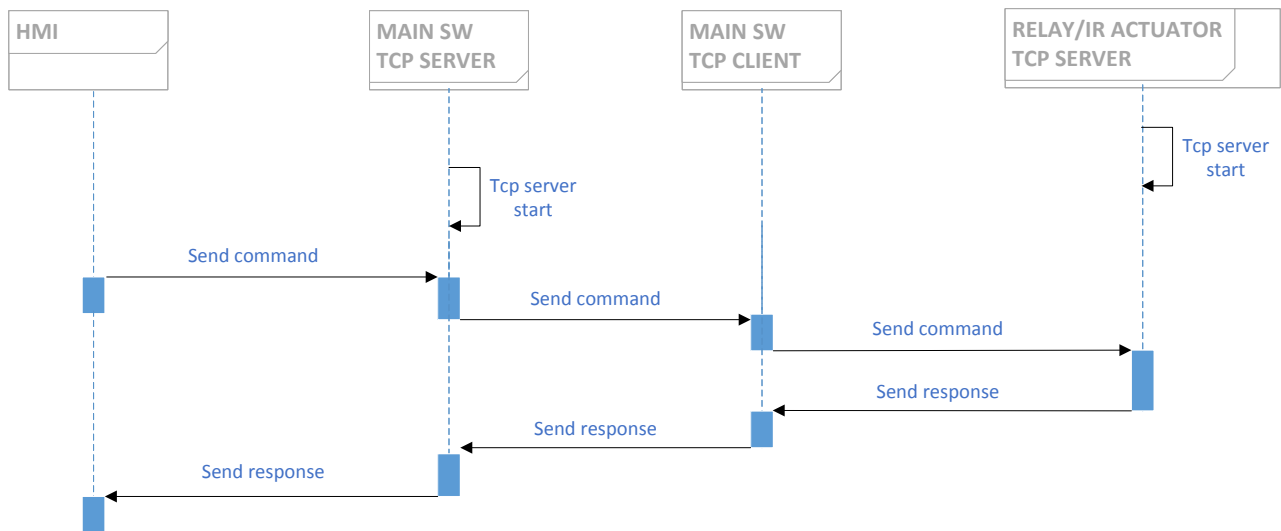


Figure 16 - Sequence diagram

The hardware design is divided into embedded systems and electrical diagrams.

7.1 The server of the main system

The main software of the whole system resides on a Raspberry Pi 2 embedded system [52]; it is written with the C++ programming language using the QT 4 framework (§7.1.1) that is suggested for developing connections and a data exchange by opening special sockets.

The software starts a TCP/IP server that communicates with the HMI MATLAB software indicating all the available commands that can be sent to surrounding environment.

In the Figure 17 are depicted the main connections between the software modules of the PAHMI system.

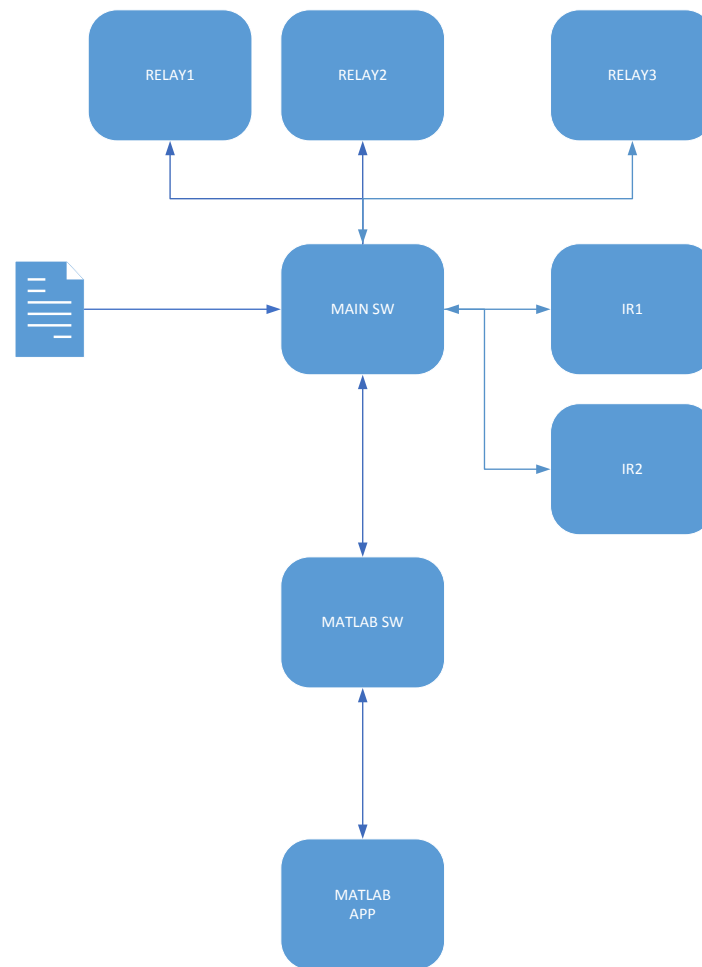


Figure 17 - Main PAHMI communications

In this way, at the startup of PAHMI, the MATLAB software tries to establish the communication with the main software module, knowing its IP and port. If the connection is not established, the HMI cannot present the available commands impartible to the environment, therefore, it waits for connection; otherwise, if the connection is successfully established, there is an exchange of information containing the current state of the domotic environment (§7.2).

In order to know the state of the environment, the main software module must be able to communicate with the actuators available.

The server module has a configuration file as input with the details (*key word, IP, port, timeout*) of the specific actuators. The first check that the server has to perform at its startup, is to check if all the actuators are available, so the software starts a client module that tries to establish a connection, using the TCP/IP protocol, to the server of the actuator on-board software.

In the Figure 18 it is possible to verify the communications divided into TCP server and TCP client connections that can be established.

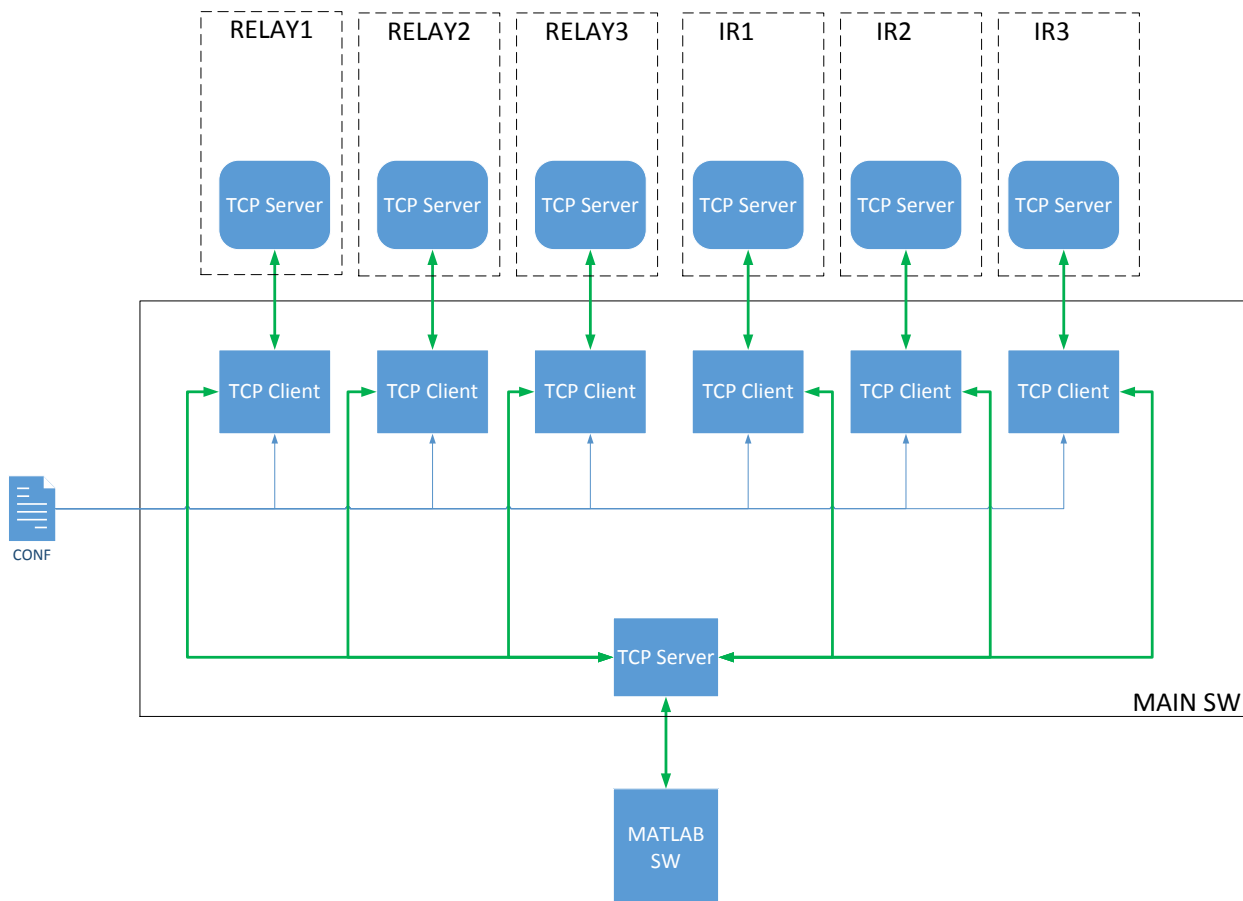


Figure 18 - TCP server/client communications

When the connection with the actuators is established, the server provides the state of the environment to the MATLAB application, using a special encoded string. In this way, the MATLAB application returns to user a dynamic HMI, presenting non-redundant commands, for example, if a device is switched on, the possible command is "switch off" and vice versa.

7.1.1 Qt library

Qt is an independent-platform library for developing programs with graphical interface using *widget* (graphic elements). Qt employs C++ standard language with a massive C preprocessor used for adding language plus.

Qt allows MS Windows, UNIX/Linux, Mac OS X, Embedded Linux, Windows CE/Mobile, Android, iOS, and others language applications developing.

The Qt power stands to the multiplatform code and binary availability, without any changes.

All the Qt library references are available in [53].

The use of this library is mainly due to some functionality in particular; these functionalities include:

- A useful and efficient communication mechanism for the objects, named *Signal and Slot*
- The connection network managing, i.e. TCP or UDP server and client implementation
- The implemented class hierarchy, methods and attributes are immediately usable during the coding phase
- The XML document parsing

7.1.2 Configuration

The configuration file of the central system is a list of values that correspond to the following header:

DOM_MOD; KEY_WORD; IP; PORT; TIMEOUT

The *DOM_MOD* can assume the value 0 or 1: the 0 value indicates that the device to be controlled is a relay; the 1 value, instead, indicates that the device to be controlled is an infrared led.

The *KEY_WORD* for relay device is a keyword unique and it distinguishes all devices that can be controlled; the controlled devices, and their embedded systems, are as many as the lines of configuration file having *DOM_MOD=0*. While, for infrared devices, *KEY_WORD* indicates the name of the device to be controlled and the specific command; the corresponding embedded system software contains all the IR codes that can be sent to the controlled device (such as the channels of a television), therefore, for each of the configuration file line having *DEV_MOD=1* and the same couple of values of *IP* and *PORT*, the embedded system is unique.

The *IP*, *PORT* and *TIMEOUT* values are the data to establish a TCP/IP connection to the embedded system that control actuator. They are respectively the connection's IP and port, and the time within a reply is expected from the embedded system before the device it's considered *OFFLINE*.

This is an example of an extracted configuration file:

```
#DOM_MOD = Domotic Modality is 0 for relay, 1 for IR codes
#KEY_WORD = Key word, unique in the list, refers to mac of the
esp8266/arduino (if the DOM_MOD is 0), otherwise refers to IR
codes (if the DOM_MOD is 1)
#IP = ip of the esp8266/arduino TCP server
#PORT = port of the esp8266/arduino TCP server the same ip/port
couple refers to #a specific esp8266/arduino/embedded system
```

```
#TIMEOUT = max time for receiving response from arduino/esp8266
TCP server (always the same for the same ip/port couple)
```

```
#DOM_MOD;KEY_WORD;IP;PORT;TIMEOUT
```

```
0;RELAY1;192.168.1.208;8888;8000
```

```
0;RELAY2;192.168.1.48;8887;6000
```

```
0;RELAY3;127.0.0.1;8886;6000
```

```
1;TVLG-tvradio;192.168.1.34;8888;6000
```

```
1;TVLG-POWER;192.168.1.34;8888;6000
```

7.2 The server of the actuator

The software built on embedded platform is a server that can communicate with the central software system and it can send the command chosen by the user through the HMI.

The software is written in *Wiring* programming language that is an open-source framework for microcontrollers [54]. Wiring allows writing cross-platform software to control devices attached to a wide range of microcontroller boards to create all kinds of creative coding, interactive objects, spaces or physical experiences.

The software contains a server that accepts TCP/IP connections from the client of the main embedded system. When the connection is established and the server receives a message with an appropriately coded string, the command is sent to the controlled device.

In particular, the coded string depends on the type of the actuator connected to the embedded system:

- if *DEV_MODE*=0 (relay), the string is as follows → *0/KEY_WORD/0/1*
 - 0 is the *DEV_MOD*
 - the *KEY_WORD* is a key word that identifies the controlled device
 - the 0 or 1 value indicates the switching off (0) or switching on (1) of the device, that it translates to an opened or closed circuit relay
- if *DEV_MODE*=1 (IR led), the string is as follows → *0/KEY_WORD*
 - 1 is the *DEV_MOD*
 - The *KEY_WORD* indicates the device and the command to be sent to the device trough IR led

For IR transmitter, the software of the actuator used a library named *IRremote* (§7.2.1).

7.2.1 IRremote library

IRremote is a library that permits to receive or transmit Infrared Remote Control codes [55]. It is possible to do projects controlled by a remote and to control other devices like televisions and stereo components.

The hardware requirement for transmitting is a single Infrared LED and a resistor.

IRsend creates the transmit object. A fixed pin number is always used, depending on which timer the library is utilizing.

IRremote requires a timer for transmitting but it can be edited updating the *IRremoteInt.h* file with a different timer.

7.3 The hardware implementation

The main actuating systems implemented are two: the first one is a relay (switch on/off), the second one is an IR transmitter.

Both actuation systems are controlled by an embedded system belonging to the category of WIFI modules named ESP8266 ([56] and [57]); this kind of embedded system is currently the most widely used for IoT and Home Automation systems.

7.3.1 WIFI controlled relay using ESP8266

In order to remotely turn on/off a device through the voltage regulation, it is necessary that an embedded system sends a command to the actuator connected to it. The actuator is controlled by a relay powered at 5V and it is connected to the embedded system via a digital GPIO [58].

The suitable embedded system for this purpose is the *ESP8266-12E* device, chosen for its ability to connect to a WIFI network, for its small size, low cost and the availability of digital and analog pins.

The Figure 19 below shows the wiring diagram of the system created in order to interrupt the power supply (250V) and therefore turn off (or turn on) a connected device.

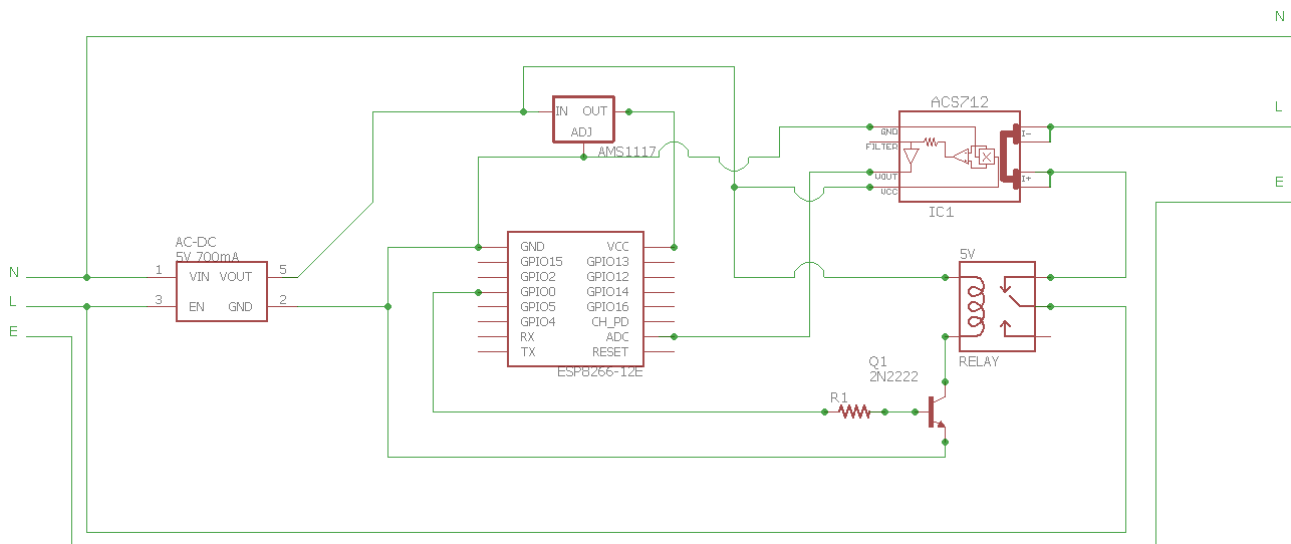


Figure 19 - Electric schematic of an embedded system to switch on/off a device connected to a relay

It is important to always keep the voltage consistent with that required by each device to ensure that there is no loss of functionality from one or more devices. This is crucial because the project involves high power supplies (250V) potentially dangerous for humans if not handled properly.

As depicted in the Figure 19, the power input to the system comes from the 250V power supply of the home automation environment; this power must be transduced to 5V using a special voltage regulator AC-DC 5V 700mA (Power Supply Buck Step Down Converter Module). The 5V voltage obtained must be further transduced as the *ESP8266-12E* WIFI device requires a voltage of 3.3V; this transduction is carried out through the *I2C AMS1117* device.

With the availability of the tensions 5V and 3.3V it is possible to power the WIFI and the relay devices.

The relay chosen is the *Songle SRD 5V DC - 250V AC* and it is controlled by the *ESP8266-12E* device via a digital GPIO, the *GPIO0*. Between the relay signal and the *GPIO0* pin it has been connected a 2N2222 transistor *Q1* and a resistor *R1* equal to $1k\Omega$ in order to adjust the voltage.

Finally, it is important to consider a current sensor to effectively measure if the current circulates in the device that has to be remotely controlled. This is required to provide information on the current status of the device; In fact, it is not necessary to know the exact value of the current, but the purpose is to know if the device is already on or not. By having this information, the HMI provides to the user only the feasible commands that can be sent to the device, rather than having redundant commands.

The current sensor is an *I2C ACS712* powered at *5V* and commanded by *ESP8266-12E* device via the analog ADC pin.

From the Figure 19, it is evident that only the phase *L* of the *250V AC* voltage is controlled, while the neutral *N* and the earth *E* connected directly to the device; it is sufficient to stop the phase *L* to turn off the device.

7.3.2 WIFI controlled IR led using ESP8266

In order to turn on/off a device remotely using infrared commands, it is necessary that an embedded system sends a command to the IR LED connected to it [59].

The actuator that has to be controlled is an IR LED connected to the embedded system via a digital GPIO.

The suitable embedded system for this purpose is the *ESP8266-12E* device, chosen for its ability to connect to a WIFI network, for its small size, low cost and the availability of digital pins.

The Figure 20 below shows the wiring diagram of the system created in order to send an IR command and therefore turn off (or turn on) a remote device.

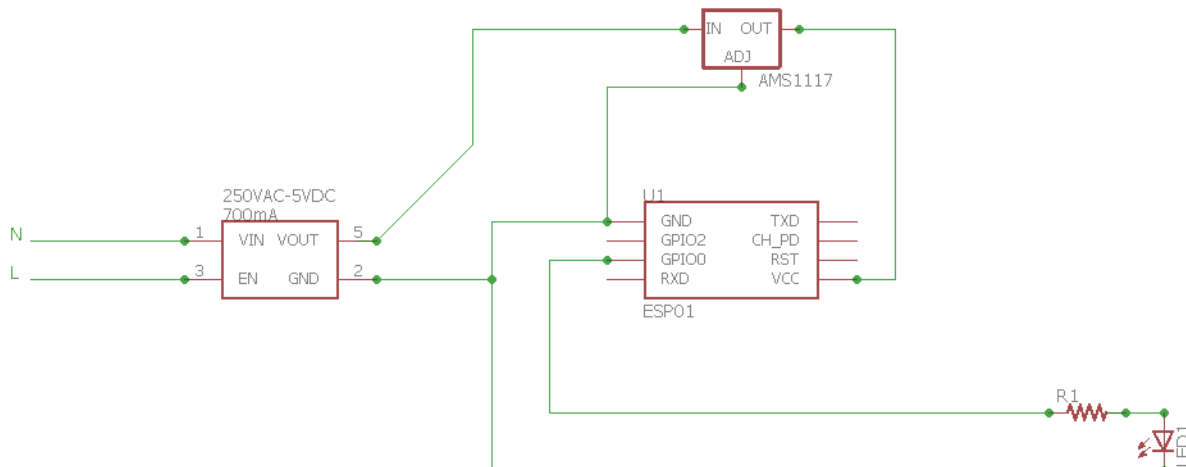


Figure 20 - Electric schematic of an embedded system to switch on/off a device via IR command

As depicted in Figure 20, the power input to the system comes from the *250V* power supply of the home automation environment; this power must be transduced to *5V* using a special voltage regulator *AC-DC 5V 700mA* (Power Supply Buck Step Down Converter Module). The *5V* voltage obtained must be further transduced as the *ESP8266-01 WIFI* device requires a voltage of *3.3V*; this transduction is carried out through the *I2C AMS1117* device.

The IR LED is controlled by *ESP8266-01* device via a digital GPIO, the *GPIO0*.

8 Simulations

A MATLAB 2015b (© Mathworks) environment has been developed to test the proposed PAHMI system.

The smartphone used to perform the experimentations is an iPhone 4s (operative system 7.1.2 iOS).

When the MATLAB software starts, it attempts to establish the connection with the main software through the 192.168.1.207 IP and 1234 port. The main software responses with 4 actuators connected: two of them are IR transmitters and the other two are relays.

The operational space is 1280×720 pixels and M square targets (with dimension 70×70 pixels) can be positioned around a circle as shown in Figure 21, with $M = 4$ (the number of actuators connected).

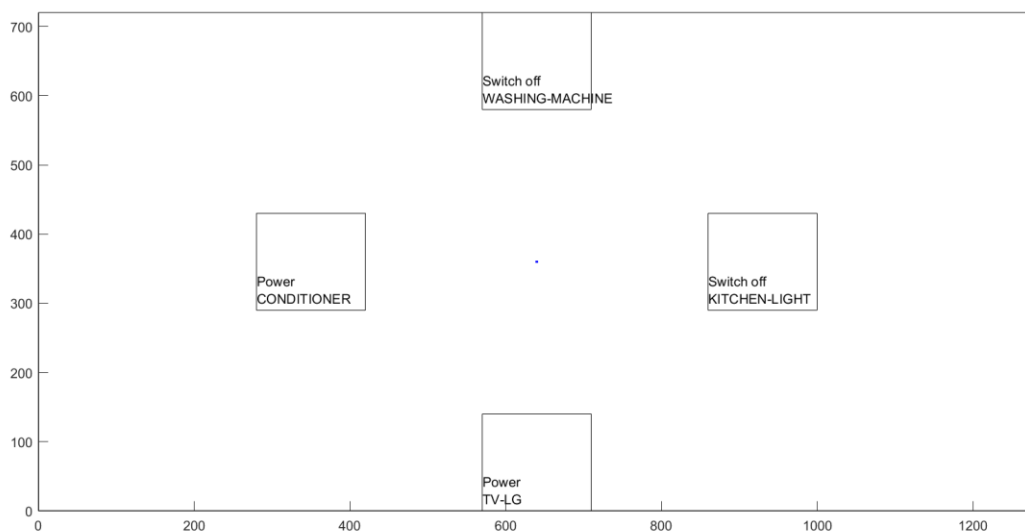


Figure 21 - Operational space with 4 targets

At the beginning of each task, the cursor appears in the centre of the operational space and the task is to direct the cursor over one of the targets. The target is represented by a square area and it is considered as acquired if the cursor remains within the area for 2s. The cursor can be driven by the smartphone via WIFI (§4.3).

At first, the software has to be trained so that it can store the information about the user features; moreover, for each target were 50 tests were performed, therefore 50×4 iterations. This kind of test has been executed by a disabled user and a normal one.

In the following Figure 22, Figure 23, Figure 24 and Figure 25 the trajectories followed to reach each target by a normal user and a disabled user are depicted.

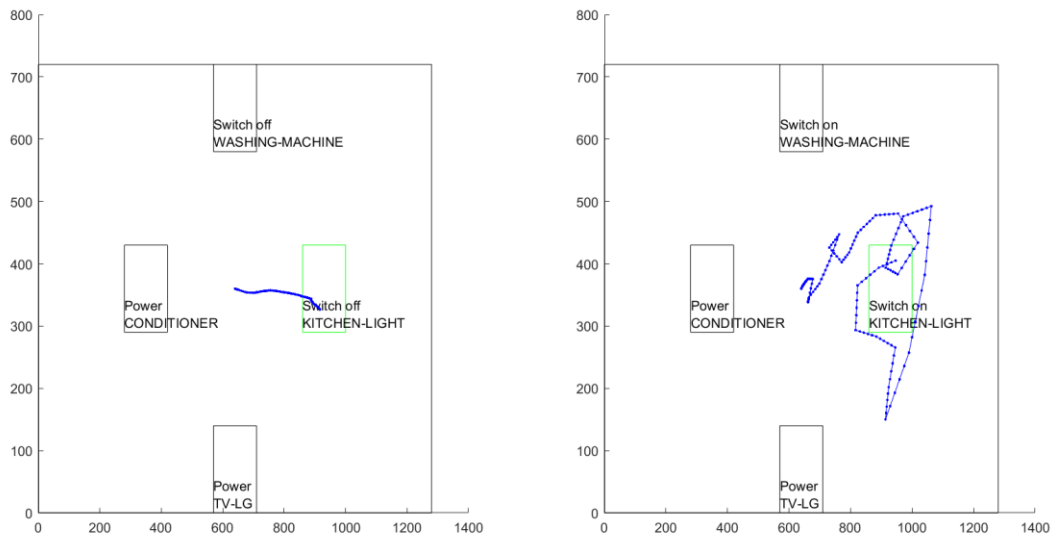


Figure 22 - Recorded trajectory for a normal user (left) and a disabled user (right) to reach the first target

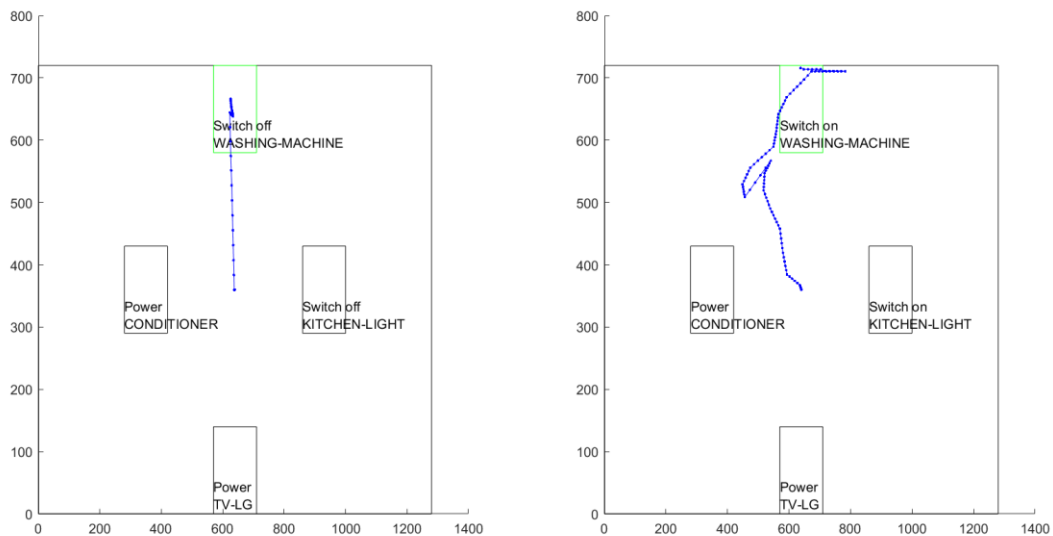


Figure 23 - Recorded trajectory for a normal user (left) and a disabled user (right) to reach the second target

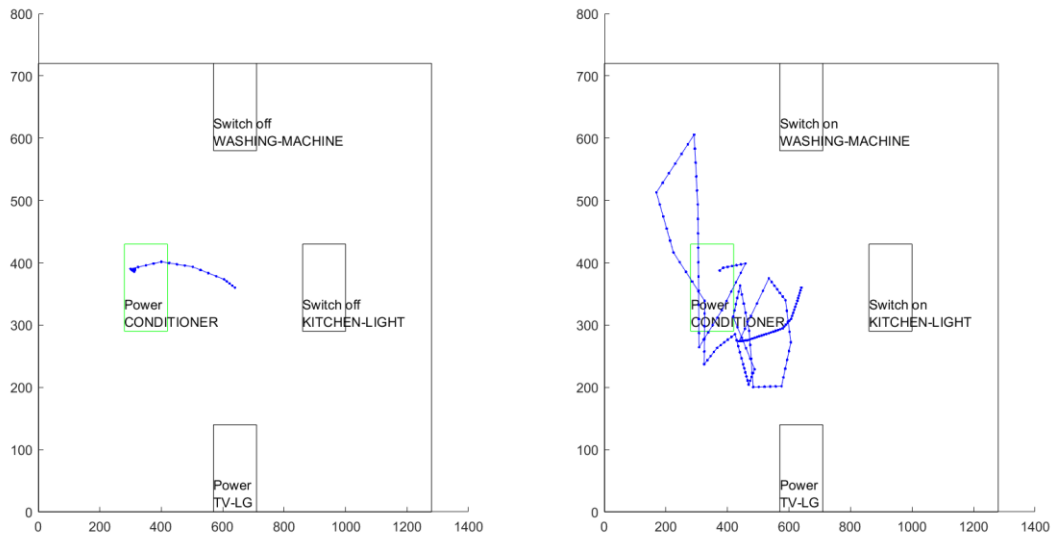


Figure 24 - Recorded trajectory for a normal user (left) and a disabled user (right) to reach the third target

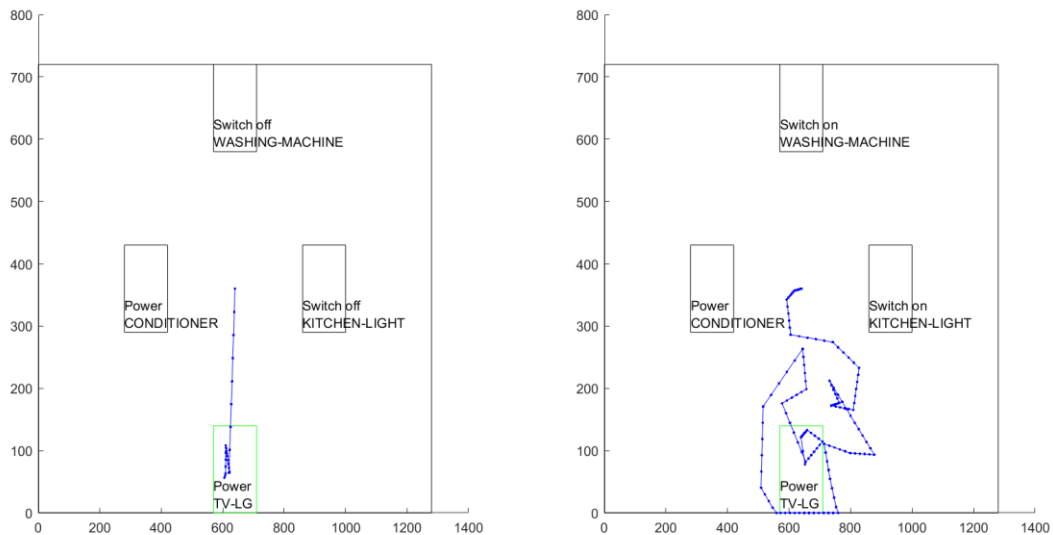


Figure 25 - Recorded trajectory for a normal user (left) and a disabled user (right) to reach the fourth target

The trajectories shown in the figures are chosen randomly among the 50 trajectories that the user used to train the system, while the target to be achieved is highlighted in green (§3.5).

As is clear from the studies of the trajectories, it is possible to understand that the disabled user has more difficulties to reach the desired target, moreover, there are movements not so fluid between the sampling instants. Even the normal user manifests some difficulties, although he is able to maintain the same direction for a longer time.

Certainly, one of the difficulties encountered by both users is to maintain the cursor contained within the target for the time of 2s necessary to obtain the desired command.

The following table shows some of the simulation data and the results obtained.

Parameter	SET 1	SET 2
Operational plan size	$1280 \times 720 \text{ px}^2$	$1280 \times 720 \text{ px}^2$
User	Disabled	Normal
Distance centre-targets	350 px	350 px
Near/far threshold d_{th}	175 px	175 px
Target area	$70 \times 70 \text{ px}^2$	$70 \times 70 \text{ px}^2$
# of targets M	4	4
Time to acquire a target	1.2 s	1.2 s
# traject. per target (train.)	50	50
# traject. per target (contr.)	50	50
Sample time T_s	0.1 s	0.1 s
Control time T_i	0.1 s	0.1 s
Segment samples n_s	12	12
Avg. time to reach target 1	9.85 s	3.15 s
Avg. time to reach target 2	14.15 s	5.15 s
Avg. time to reach target 3	14.4 s	4.4 s

Avg. time to reach target 4	14.85 s	4.85 s
-----------------------------	---------	--------

Table 3 - Simulation parameters

The SVM were generated using the *LIBSVM* library with Gaussian kernel, [39], [60].

After the SVM validation phase, the far and near confusion tables were computed, as explained in §5.4.

The following matrices shown in Figure 26:

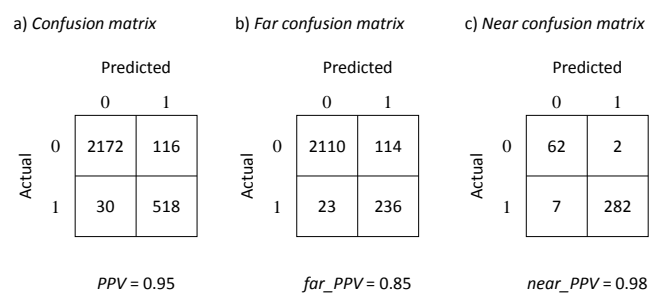


Figure 26 - Confusion matrices

are an example of couple of full (left), far (centre) and near (right) confusion matrices obtained for a target of a run of simulation set 1.

Now that the PAHMI system is trained and it has the information about the user's features, provided by SVM (§5.4), it is possible to check how much the control applied in real time is efficient.

In the Figure 27, Figure 28, Figure 29 and Figure 30 below is shown how much the guided control of the cursor movement towards the desired target has improved the targets achievement time for both users.

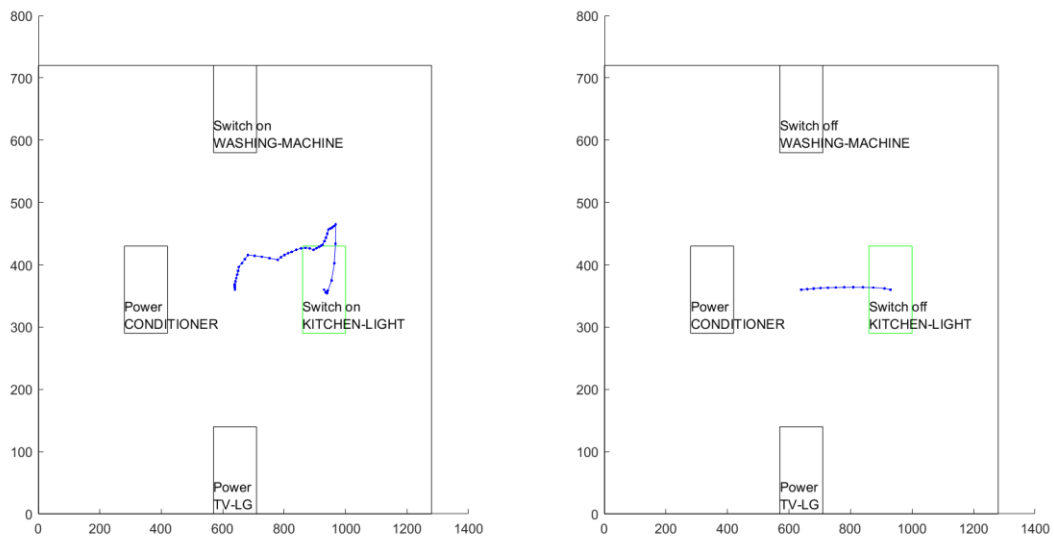


Figure 27 - Controlled trajectory for a disabled user (left) and a normal user (right) to reach the first target

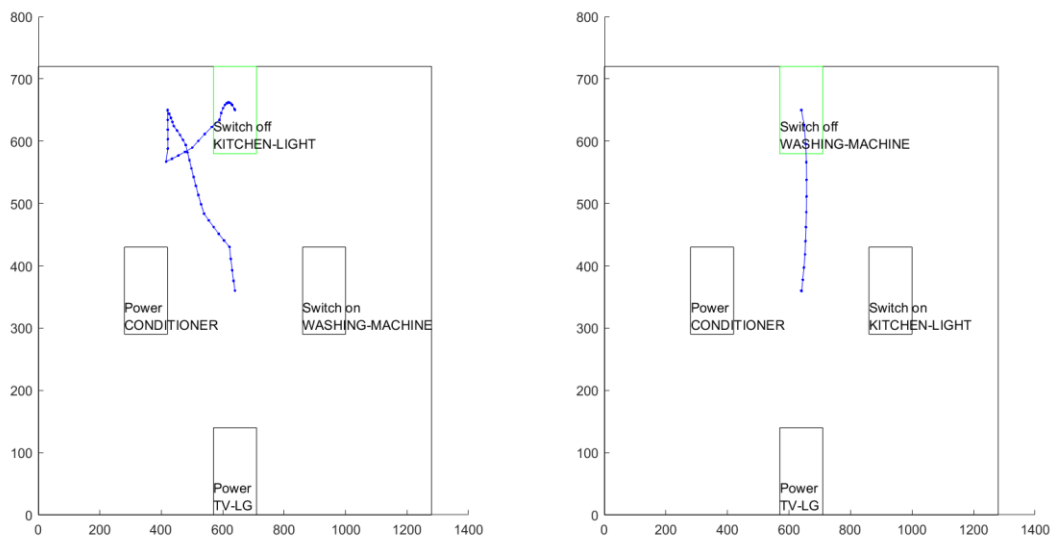


Figure 28 - Controlled trajectory for a disabled user (left) and a normal user (right) to reach the second target

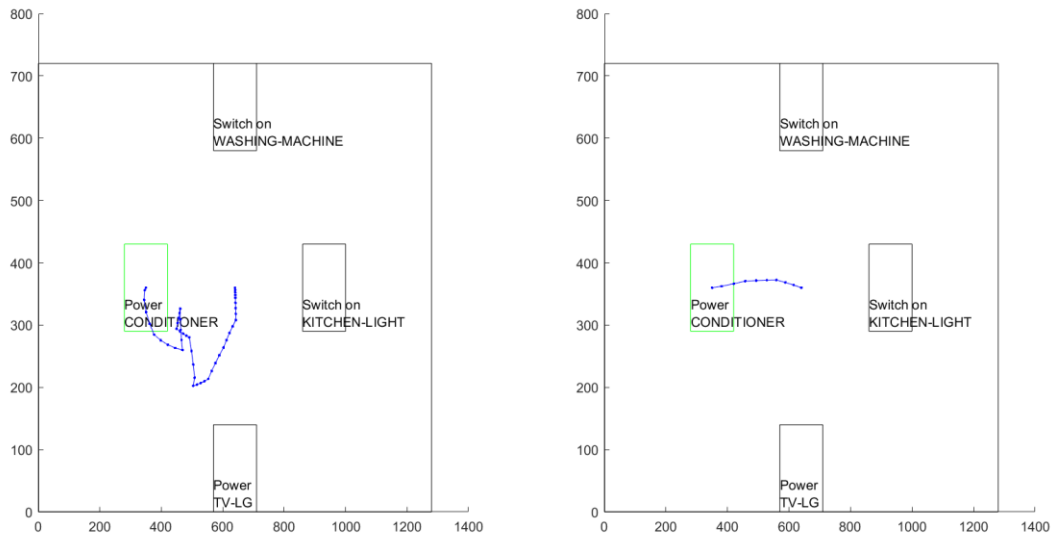


Figure 29 - Controlled trajectory for a disabled user (left) and a normal user (right) to reach the third target

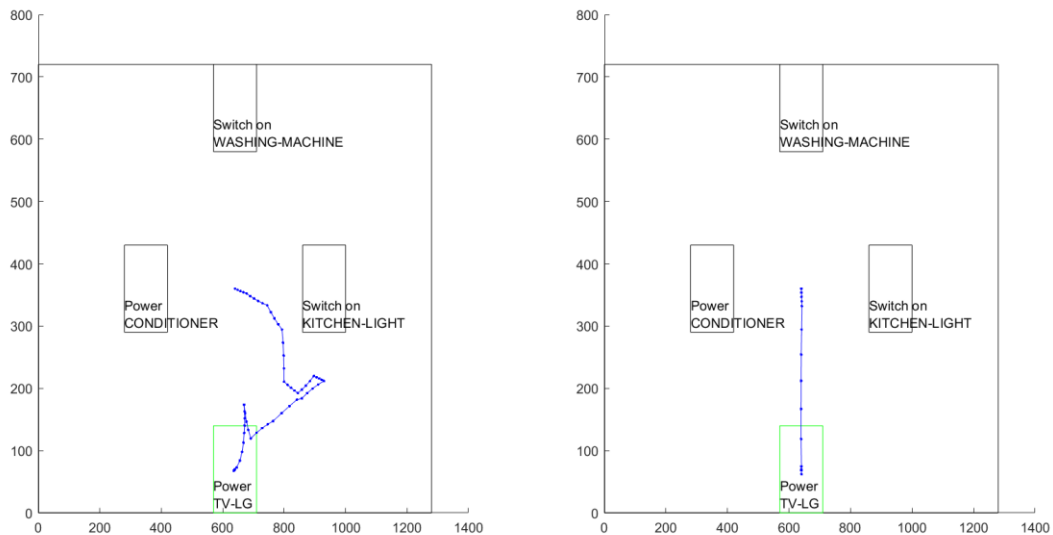


Figure 30 - Controlled trajectory for a disabled user (left) and a normal user (right) to reach the fourth target

The Table 4 and the Table 5 collect, for the normal and disabled user respectively, the average of the task completion times with the HMI and with the PAHMI, as well as the standard deviation; the tables also show the obtained near and far PPVs of the SVMs. The high values of PPV do not represent the capacity of the multi-class classifier to predict the target the trajectory segment is directed to: the PPV is the precision of the prediction that the destination target is, e.g. target 1, subject to the fact that the classifier predicts that the destination target is target 1. For instance, there might be a segment, at time k , for which no SVM returns 1 (i.e., each SVM predicts that the trajectory is not directed to its associated target): in those cases, no assistance is given to the user.

Target	Avg. HMI	Std. dev. HMI	Avg. PAHMI	Std. dev. PAHMI	Far PPV	Near PPV	Improvement
1	3.15 s	3.93 s	2.2 s	1.70 s	0.98	0.99	30%
2	5.15 s	3.41 s	2.7 s	2.74 s	0.89	1	51.2%
3	4.4 s	2.85 s	2.1 s	1.91 s	0.94	0.99	50%
4	4.85 s	4.87 s	2.4 s	1.82 s	0.99	0.99	52.7%

Table 4 - Resulted for the normal user

Target	Avg. HMI	Std. dev. HMI	Avg. PAHMI	Std. dev. PAHMI	Far PPV	Near PPV	Improvement
1	9.85 s	8.88 s	5.2 s	4.53 s	0.91	0.98	47.2%
2	14.15 s	10.75 s	8.3 s	4.50 s	0.98	0.99	41.3%
3	14.4 s	10.05 s	7.1 s	5.13 s	0.93	0.96	50.7%
4	14.85 s	11.16 s	5.1 s	2.58 s	0.85	0.95	65.7%

Table 5 - Resulted for the disabled user

The Figure 31 and Figure 32 depict, in bar diagrams, the improvement results of the disabled and of the normal user to reach each target.

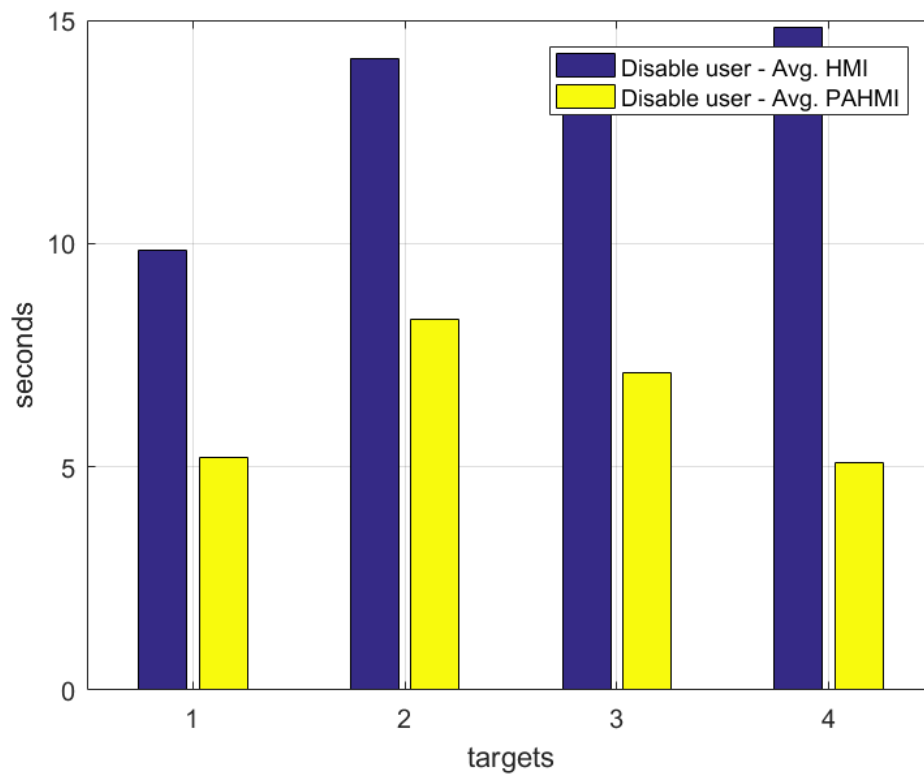


Figure 31 - Improvement results for the disabled user to reach each target

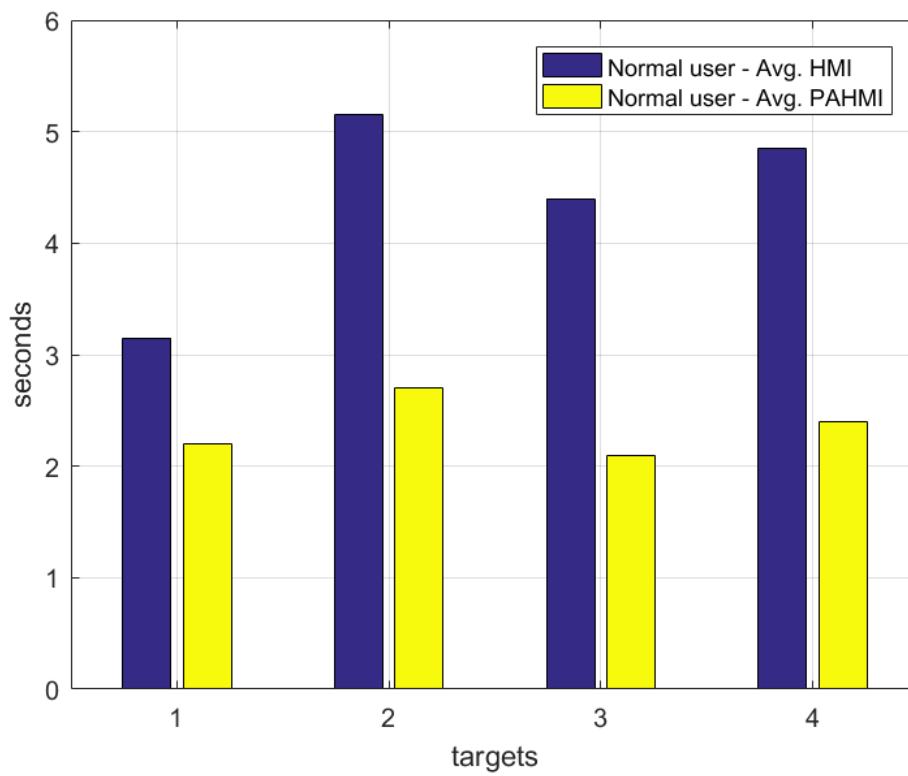


Figure 32 - Improvement results for the normal user to reach each target

Finally, when the target is reached, the result is that the corresponding actuator executes the command desired by the user as explained in the previous paragraphs (§7.3), so the light goes on, or the TV powers and so on.

9 Conclusions and future works

This thesis work has permitted the development of an entire system to establish an interaction between disabled users and surrounding environment. It has been developed both the software and hardware implementation.

The idea originates from the particular need to create a system in the world of the Internet of Things, that is continuously expanding, and to use the auxiliary technologies to help users with severe physical difficulties to interact with the domotic environment.

The PAHMI system provides a user interface endowed with some targets, that corresponds to commands that can be sent to the surrounding environment; to reach a target, the user has to rotate, along two directions, a smartphone. The user has to imagine to have a ball at the centre of the smartphone plane, corresponding to the HMI plan; the ball corresponds to the cursor of the PC mouse, starting from the centre of the HMI. Through the rotation of the smartphone, the cursor starts to move in the direction set by the user. Once the cursor reach the target, and then the user requests its desired command, the corresponding actuator executes the order.

The PAHMI system is divided into software and hardware implementations.

The software implementation is divided into MATLAB mobile application for smartphone, MATLAB software, C++ software on board of a Raspberry Pi system and Wiring software on board of embedded systems. In particular, the MATLAB software provides the HMI created following Human Factor studies, and it accepts a connection coming from the MATLAB mobile application to exchange data of the smartphone accelerometer. Furthermore, the MATLAB software communicates through TCP/IP connection with the C++ software built on the Raspberry Pi to know and to request the desired and executable commands by the user. The C++ software sorts the received commands from the MATLAB software and send them to the Wiring software built on specific embedded systems; this connection exploits again the TCP/IP protocol. All the connection parameters are in a configuration file in input to the C++ software. The software built on the embedded systems are demanded to execute the final command to the controlled device.

The hardware implementation is of two types, the first one uses a relay to interrupt or not the power to the controlled device; the second one is a IR transmitter, that sends infrared signals to devices like TVs or air conditioners.

The main embedded system used for the PAHMI system is the ESP8266 element endowed with a WIFI antenna; it is the most suitable system to reach the goal, given its extremely small size.

The transduction from the smartphone accelerometer to the movement of the cursor occurs using the equation of the inclined plan motion combined into two rotation axis and, subsequently, using the ideal unicycle model.

The PAHMI system, compared to other systems that permit the interaction between the user and the domotic environment, uses a learning phase in which it is exercised by the user. In this phase, the PAHMI learns some user features like, e.g., its predisposition to go in a particular direction or its velocity or its ability to maintain a direction. Furthermore, this learning phase has to be performed only one time for each user. All the information gained from the learning phase are analysed by the Support Vector Machines that supervised learning models with associated learning algorithms for classification and regression analysis.

The real using phase of the system requires a Butterworth filter that has to be applied to the acceleration coming from the smartphone accelerometer. The filter is useful to avoid abrupt changes of the cursor speed. Then, a PID controller is applied to assist the cursor movement to the desired command of the user. The control uses the results provided by the SVM, built during the learning phase, and it is applied real time.

The PAHMI system is completed and it has been tested by several users; the results of all of them helped the system improvement for the calibration of the used defines.

For the future, in order to improve the PAHMI system's potential different kind of technologies can be integrated to meet the needs of the IoT, like, e.g., radio frequencies (Zigbee) or Bluetooth/Bluetooth Low Energy. Furthermore, embedded systems can be used to optimize the actuator's dimensions to integrate them into the electric system. Then, another embedded system endowed with an accelerometer can substitute the smartphone, in this way, a smaller device can be used more easily by the user [61].

Finally, with regard to the software, a tuner of the controller parameters can be developed to achieve an optimized result.

Bibliografy

- [1] C. Forget-Leblois, “Internet of Things: New Promises for Persons with Disabilities,” no. July, 2015.
- [2] F. Martelli, “FILTRAGGIO DI TRAIETTORIE DI PUNTATORI A SENSORI INERZIALI PER UTENTI CON PARALISI CEREBRALE,” 2014.
- [3] W. Zhou, L. Zhang, and L. Jiao, “Linear programming support vector machines,” *Pattern Recognit.*, vol. 35, no. 12, pp. 2927–2936, 2002.
- [4] S.-W. Lin, Z.-J. Lee, S.-C. Chen, and T.-Y. Tseng, “Parameter determination of support vector machine and feature selection using simulated annealing approach,” *Appl. Soft Comput.*, vol. 8, no. 4, pp. 1505–1512, 2008.
- [5] M. Egerstedt, “Control of Mobile Robots - Module 4: Control Design,” 2014.
- [6] M. Egerstedt, “Control of Mobile Robots - Module 5: Hybrid Systems,” 2014.
- [7] K. Polat and S. Güneş, “Breast cancer diagnosis using least square support vector machine,” *Digit. Signal Process.*, vol. 17, no. 4, pp. 694–701, 2007.
- [8] M. F. Akay, “Support vector machines combined with feature selection for breast cancer diagnosis,” *Expert Syst. Appl.*, vol. 36, no. 2 PART 2, pp. 3240–3247, 2009.
- [9] A. K. Johan, “PID Control,” pp. 216–251, 2002.
- [10] X. Feng, Y. Laurence, W. Lizhe, and V. Alexey, “Internet of Things,” *Int. J. Commun. Syst.*, vol. 23, no. 5, pp. 1101–1102, 2012.
- [11] O. Vermesa and F. Peter, *Internet of Things - Converging Technologies for Smart Environments and Integrated Ecosystems*, vol. 291, no. 4. 2013.
- [12] S. Yinbiao *et al.*, “Internet of Things: Wireless Sensor Networks,” *Int. Electron. Commision*, no. December, pp. 1–78, 2014.
- [13] F. Peter, *Internet of Things - From Research and Innovation to Market Deployment*, vol. 6, no. 1. 2014.
- [14] M. C. Domingo, “An overview of the Internet of Things for people with disabilities,” *J. Netw. Comput. Appl.*, vol. 35, no. 2, pp. 584–596, 2012.

- [15] L. Coetzee and G. Olivrin, "Inclusion Through the Internet of Things," *Assist. Technol.*, 2012.
- [16] S. Katrina and W. Shirli, "Changing attitudes to learning disability," 2015.
- [17] World Health Organization, *The International Classification of Functioning, Disability and Health*. 2001.
- [18] D. Theresia, "Definition of Disability," no. August, 2004.
- [19] C. on T. and Disability, "Assistive Technology," 2013.
- [20] N. Scholz, "Assistive technologies to support the needs of people with disabilities," no. June, pp. 1–8, 2015.
- [21] Gagan, "IOT based system for person with physical disability," *Ijireeice*, vol. 4, no. 2, pp. 157–160, 2016.
- [22] L. M. Muñoz, A. Casals, L. M. Muñoz, and A. Casals, "Human – Computer Interaction Improving the Performance of Input Interfaces Through Scaling and Human Motor Models Improving the Performance of Input Interfaces Through Scaling and Human Motor Models," vol. 24, no. June, 2016.
- [23] T. Jen, "Human Factors in HCI Design Pervasive Computing: Technology - Centric View," 2010.
- [24] A. J. D. Rathnayaka, V. M. Potdar, and S. J. Kuruppu, "Evaluation of wireless home automation technologies," *IEEE Int. Conf. Digit. Ecosyst. Technol.*, no. January, pp. 76–81, 2011.
- [25] Moor Insights & Strateg, "Wireless Technologies for Home Automation," pp. 1–11, 2014.
- [26] M. M. Mathwroks, "MATLAB Support Package for Apple iOS Sensors," 2015. .
- [27] U. Physics, "FRICTION, WORK, AND THE INCLINED PLANE," pp. 39–47, 2015.
- [28] Studyphysics©, "Gravity on Inclined Planes," pp. 27–29, 2014.
- [29] V. Kim, "How to Design 10 kHz filter," 2005.
- [30] A. V. Oppenheim, "Butterworth Filters," pp. 422–428, 2005.
- [31] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," vol. 43,

pp. 1–43, 1997.

- [32] A. C. Berg, “SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category,” 2006.
- [33] Chih-Wei Hsu and Chih-Jen Lin, “A comparison of methods for multiclass support vector machines,” *IEEE Trans. Neural Networks*, vol. 13, no. 2, pp. 415–425, Mar. 2002.
- [34] V. Vapnik, S. E. Golowich, M. Ave, and M. Hill, “Support Vector Method for Function Approximation, Regression Estimation and Signal Processing,” 1997.
- [35] M. Sciandrone, “Support Vector Machines Cenni sui concetti riguardanti la teoria statistica dell ’ apprendimento,” 2005.
- [36] B. Caputo, C. Schuldt, and I. Laptev, “Recognizing Human Actions: A Local SVM Approach,” pp. 3–7, 2004.
- [37] R. Burbidge and B. Buxton, “An introduction to support vector machines for data mining,” *Keynote Pap. young OR12*, pp. 2–14, 2001.
- [38] S. Tong and D. Koller, “Support Vector Machine Active Learning with Applications to Text Classification,” *J. Mach. Learn. Res.*, pp. 45–66, 2001.
- [39] D. Meyer, “Support vector machines: the interface to libsvm in package e1071,” ... *Syst. their ...*, vol. 1, pp. 1–8, 2014.
- [40] A. Solberg, “SVM classification,” 2013.
- [41] S. Armah, S. Yi, and T. Abu-lebdeh, “IMPLEMENTATION OF AUTONOMOUS NAVIGATION ALGORITHMS ON TWO-WHEELED GROUND MOBILE ROBOT,” vol. 7, no. 1, pp. 149–164, 2014.
- [42] M. Egerstedt, “Control of Mobile Robots - Module 1: Introduction to Controls,” 2014.
- [43] M. Egerstedt, “Control of Mobile Robots - Module 2: Mobile Robots,” 2014.
- [44] M. Egerstedt, “Control of Mobile Robots - Module 6 The Navigation Problem.” 2014.
- [45] M. Egerstedt, “Control of Mobile Robots - Module 3: Linear Systems,” 2014.
- [46] M. Abu-khalaf, R. Chen, and A. Turevskiy, “PID Control Design Made Easy,” pp. 2–5, 2010.

- [47] M. Egerstedt, “Control of Mobile Robots - Module 7: Putting It All Together.” 2014.
- [48] J. Weston, “Support Vector Machine (and Statistical Learning Theory),” 2008.
- [49] M. Egerstedt and X. Hu, “A hybrid control approach to action coordination for mobile robots,” *Automatica*, vol. 38, no. 1, pp. 125–130, 2002.
- [50] M. H. Nguyen and F. de la Torre, “Optimal feature selection for support vector machines,” *Pattern Recognit.*, vol. 43, no. 3, pp. 584–591, 2010.
- [51] D. G. Ferriero, “DATA RECORDER,” pp. 1–30, 2016.
- [52] D. Whale, “The Raspberry Pi projects book,” 2014.
- [53] G. Rudy, “Cross-platform C++ development using Qt,” 2005.
- [54] A. G. Smith, *Introduction to Arduino*. 2011.
- [55] Arduino, “<https://github.com/z3t0/Arduino-IRremote>,” 2016. .
- [56] E. Systems, “ESP8266EX Datasheet,” 2015.
- [57] E. Systems, “ESP8266 WiFi Module Quick Start Guide,” 2015.
- [58] Omron, “Relay User’s Guide.” 1990.
- [59] Everlight, “Technical Data Sheet 5mm Infrared LED, T-1 3/4 IR333-A,” pp. 1–7, 2005.
- [60] C. Chang and C. Lin, “LIBSVM: A Library for Support Vector Machines,” pp. 1–39, 2013.
- [61] STMicroelectronics, “SensorTile: The IoT design Lab on the tip of a Pencil SensorTile Sensing, processing & Bluetooth Low Energy connectivity,” 2016.