Analysis and Recognition of Social Gestures for Human-Robot Interaction with Wearable Sensors

Matteo Dicenzi

DIBRIS - Department of Computer Science, Bioengineering, Robotics and System Engineering

University of Genova

In collaboration with Istituto Italiano di Tecnologia

Supervisors

Fulvio Mastrogiovanni (UNIGE), Francesco Rea (IIT), Alessandra Sciutti (IIT)

Co-supervisors

Alessandro Carfí (UNIGE), Linda Lastrico (IIT)

In partial fulfillment of the requirements for the degree of

Robotics Engineering

October, 2021

Acknowledgements

I would like to thank my supervisors, Fulvio Mastrogiovanni, Alessandra Sciutti, Francesco Rea, Linda Lastrico, Alessandro Carfì. They welcomed me with extreme kindness, professionalism and sympathy, and their contribution was essential for the development of my thesis. I would like to thank all my colleagues at the Istituto Italiano di Tecnologia and Emarolab, with whom I shared many enjoyable moments. In particular, I would like to thank those that shared with me the fantastic experience of I-RIM.

A special thanks to my family and Laura, for always supporting, appreciating, and motivating me to do my best.

Last but not least, I would like to thank all my friends, especially Marco, Emanuele and Giulia, who shared unforgettable moments with me during our studies in France.

Abstract

Communication is a fundamental aspect of our lives. It has progressed along with the evolution of human beings and uses many different modalities. Among them, non-verbal communication plays a central role. To create a more spontaneous interaction between humans and robots, social robots should be able to understand all the information we convey during interaction, including gestures. However, the gestures analyzed in the literature are often unnatural, synthetic and without social relevance.
To this regard, (i) we collected a dataset with 2884 examples of twelve common Italian hand gestures using a custom-made inertial glove, through experiments organized as human-robot interactions. The collection took place in two successive phases, in which participants reproduced gestures before and after watching a short illustrative video. The robot guided the acquisition and provided a brief description of the social context in which each gesture could be used.
We propose (ii) an analysis of the collected gestures, aimed at identifying their most informative features, which are 27% of the total, and investigating common behaviours adopted by participants during the experiments, such as the similarity in the way two specific classes are performed. We propose (iii) an offline gesture recognition model based on Long-Short Term Memory (LSTM) Recurrent Neural Network (RNN), showing that the performances evaluated on the data collected in the final session are better than the ones evaluated on the initial session. Moreover, we discuss the benefits of reducing the number of considered features, such as a 7.5% increase in the model accuracy.

Contents

List of Figures

List of Tables

Abbreviations

DOF Degree of Freedom

DTW Dynamic Time Warping

FNN Feedforward Neural Network

HRI Human-Robot Interaction

IMU Inertial Measurement Unit

kFCV k-Fold Cross Validation

LSTM Long Short-Term Memory

PaCMAP Pairwise Controlled Manifold Approximation Projection

RFE Recursive Feature Elimination

RFECV Recursive Feature Elimination and Cross-Validation

RNN Recurrent Neural Network

ROS Robot Operating System

YARP Yet Another Robot Platform

Chapter 1

Introduction

1.1 Motivation

In the past few years, research has shown that social robotics may bring major benefits to the lives of people. Robots could be used in public spaces, education and personal care (17). In order to be truly effective in the interaction, robots should *understand* human communication signals.
Communication plays a fundamental role in the course of our lives. From the moment we are born, we unconsciously participate in the process of acquiring the rules of communication. This occurs slowly and includes multiple forms of communication existing in our daily lives. They are often referred to as verbal, when they relate to the information content of the message itself (i.e., what we vocally transmit); paraverbal, when referring to the way in which we convey the message (i.e., the tone of voice); non-verbal, when referring to facial expressions and gestures (18).
Without loss of generality, human communication can also be categorized according to the intentionality of the interlocutor: it can be explicit, if two or more people intend to exchange information (19), or implicit, if a person communicates to others unintentionally, e.g., through eye gaze or body posture (20).

Unlike what it may seem, non-verbal communication plays a central role in human interactions. The importance of non-verbal communication has been underlined by studies carried out by Mehrabian, a psychologist who has shown that the communicative message inferred from non-verbal language corresponds to more than half of the total information content (21).
Consequently, to achieve a successful level of Human-Robot Interaction (HRI), robots should not only be able to process verbal information, but should rather understand all the information content derived from interactions, with great interest in what is communicated in the gestural channel. In this regard, to recognize gestures, it is necessary to develop models which, on a probabilistic basis, allow

to distinguish between different classes of gestures.
The literature contains many examples that address gesture recognition. As we will see in the bibliography review, some of them are "image-base" and work using cameras, while others are "sensor-based" and involve the employment of inertial sensors, as the case for this study. However, the gestures considered in the literature are usually artificial and synthetic (22) - (23). Instead, to pursue the feasibility of more natural and spontaneous human-robot interactions, we would like to focus our attention on gestures closely related to non-verbal communication, with a strong social connotation and clear real-world applications. Hence, we will consider the Italian Hand gestures.

1.2 Goal

The purpose of this thesis is to carry out a pioneering study on Italian gestures, a form of explicit, culture-oriented, spontaneous and yet very specific non-verbal communication. To perform this analysis, we will collect a novel dataset, containing 12 of the most popular examples of gestures in the Italian culture.
The dataset will be collected through experiments organized as human-robot interactions, mediated by the first prototype of inertial glove that, during the experiments, will be worn in the participant's right hand.
In these experiments, the humanoid robot iCub will be in charge of the data collection, allowing the human operator to be completely excluded from the scene. The goal of iCub will be to guide participants in the data collection by providing imaginary situations, identical for all, meant to help immerse themselves in the social context in which they perform the corresponding Italian gesture.
Besides the data collection, the contribution of this study is to provide an in-depth analysis of the Italian gestures. First, we will investigate the presence of similarities in the way participants perform specific gestures, with the goal of identifying the presence of similar patterns. In addition, we will explore whether fewer inertial sensors can be used than the 11 on the glove employed during the experiments.
Once these analyses are completed, we will move on to the topic of gesture recognition, the last contribution of this work. In the literature, there are many studies addressing gesture recognition through wearable sensors (9) - (8) - (24). However, as mentioned above, most of them focus on artificial and simplified gestures. Instead, the Italian gestures, which are natural and spontaneous, evolved along with other human communication channels, and this makes them conceptually different from the aforementioned synthetic gestures. On the other hand, the spontaneity and personal style by which people perform Italian gestures implies a high variability in their execution, which could lead to difficulties in recognition.

Therefore, we will address the topic of gesture recognition through an already established approach available in the literature (10), based on neural networks able to learn temporal dynamic information.
Then, we will implement a probabilistic classifier that, once trained, will recognize the gestures in the dataset. The model will be trained and tested offline, according to standard approaches based on cross-validation. According to the findings discovered during the features analysis, we will investigate the possibility of training a model based on a subset of features, establishing if it performs better than the model trained with the whole set of the features.

1.3 Thesis structure

The structure of this thesis is organized as follows. Chapter 2 is devoted to the literature review, where we discuss, in a general sense, human communication and its implications in the field of robotics. Hence, we briefly describe Italian gestures and the model of the human hand on which the inertial glove is based. More importantly, we review the state of the art of gesture recognition. The review is comprehensive and covers both the initial stages of data collection, with a focus on the technologies used for this purpose and their implications, and the implementation of the recognition model. The latter is analyzed at a general level, examining both model-based and data-driven approaches and providing their respective advantages and drawbacks.
Chapter 3 is devoted to the experiment description, where we describe the experimental protocol and the most important components, e.g., the humanoid robot iCub and the custom-made inertial glove. We explain in detail the role of the participant during the experiment, describing precisely his/her tasks. Furthermore, we specify details such as the duration of the experiment and the definition of the gesture dictionary considered in the study.
In Chapter 4 we explain the software implementation needed to carry out experiments, providing details on the overall architecture. According to their purpose, the explanation is divided into three modules. The first one is ROS-based and deals with low-level tasks, e.g., the inertial data acquisition driven by the glove. The second is YARP-based and handles the interaction with the robot.
The last module, which is the main one, communicates with the other two and manages different aspects of the experiment, e.g., robot behavior, data collection timings and inertial data storage.
Chapter 5 covers the topic of data analysis, where we reduce the number of features through data-driven techniques and investigate common behaviours adopted by participants during the experiments. Moreover, we identify the most informative features through data-driven selection algorithms.

In Chapter 6 we address the topic of gesture recognition. At first, we describe data pre-processing, i.e., normalization, automatic segmentation, padding, and then we describe the model selected for gesture recognition. More specifically, we describe the approach used to test its performance, i.e., cross-validation. Lastly, given the subset of the most informative features (identified in Chapter 5), we train a new model with that subset, compare its performance with those of the original one and evaluate the preferred solution.
Finally, in chapter 7 we summarise the work that has been carried out during this thesis and the contribution it brings. We also point out some limitations of the current state of the study, providing some ideas that could overcome them and make the study more interesting.

Chapter 2

Literature review

2.1 Human communication

Generally speaking, human communication is carried out in different modalities. Verbal communication is the common one to think about: it is the easiest way humans communicate, since it allows those who talk to directly tell something to the interlocutor. However, it is not always possible to speak: there are some situations where other channels must be used. This may be the case of two people talking in different languages, which will not be able to understand each other through words; or it may be the case of a very young child, who is not able to talk yet. In both the previous examples, the intention of the individual will be expressed through different channels. In particular, the child will use his facial expression (i.e., smiling) and his body (fast arm/hand movements) in order to communicate with the parents his needs.

Communication that does not involve words is referred to as non-verbal. One example of such communication can be found in the orchestral context: the conductor provides real-time information to the orchestral members, by moving the wands in his hands in specific ways. This kind of communication is also referred to as explicit. On the other hand, there exists implicit non-verbal communication as well. As described in (25), it is not consciously performed by the person, like the direction of the gaze that may occur whenever an action needs to be performed. For example, a person looking for the exit label in the walls of a public office may be the sign of the start of a walking motion; or more simply, a child looking at a toy may be the signal of the child's intention to grasp that object.

Sensorimotor communication is a very specific type of communication, which is useful whenever two or more people need to carry out a task that requires coordination. The concept of sensorimotor communication can be explained with the following example: suppose that two people drive two carts, one each, in series. If, at a certain point, the person leading the other one sees a sharp turn

headed, he/she may change the cart trajectory so that the person in the back understands the danger in advance.
This example evidences a common characteristic of sensorimotor communication: it is carried out in the same channel of the action (26).
According to Pezzulo *et al.* (26),"sensorimotor communication emerges as part of a strategy that enhances coordination and the success of joint action". By slightly changing the action a person is performing, it is possible to communicate information to another person within the action itself. As a consequence, the change in the action may be considered as a cost. In fact, coming back to the previous example, the person approaching the sharp turn had to change his cart trajectory, thus increasing the distance covered. However, this cost was repaid by notifying the other person of the danger to act accordingly.

Candidi *et al.* carried out an experiment (1), involving a couple of people: one is the person who leads the action (that consists of grasping a given object); the other follows and imitates the action performed by the leading person. Besides, the imitation process is symmetric: another object, equal to the original one, is grasped in a symmetric position, as shown in Figure 2.1 (right). The goal of the experiment was to understand the sensorimotor communication adopted by the leader and the follower when a reciprocal simultaneous motion is performed: when the leader grasps the object from a given point (maximum height, as depicted in Figure 2.1 - right), the follower has to grasp another equal object from the opposite direction (minimum height).
With their study, the authors noted the tendency of the leader to change his kinematic motion in order to implicitly communicate with another person. In practice, this means that when the leader has to grasp the object from its highest point, he tends to increase the height of the wrist trajectory. On the contrary, when he has to grasp the object from its lower part, the wrist trajectory will be complementary to the previous case.
Finally, they noticed that the level of synchronization increases as the implicit communication increases: the more the trajectory is different from the original one, the more the two people understand each other.

In the previous experiment it was proved the possibility to signal spatial information. However, there are other object properties like weight, fragility and temperature. These are referred to as hidden properties and are characteristics of objects that are not directly perceivable until a person has the possibility to interact with them.
In this context, Schmitz *et al.* investigated whether it was possible or not to use sensorimotor communication to signal hidden object properties (27). They discovered that one person can non-verbally communicate the weight of an object to another through sensorimotor communication. The person managing the object, who knows in advance the weight of what is grasping, will change his action such

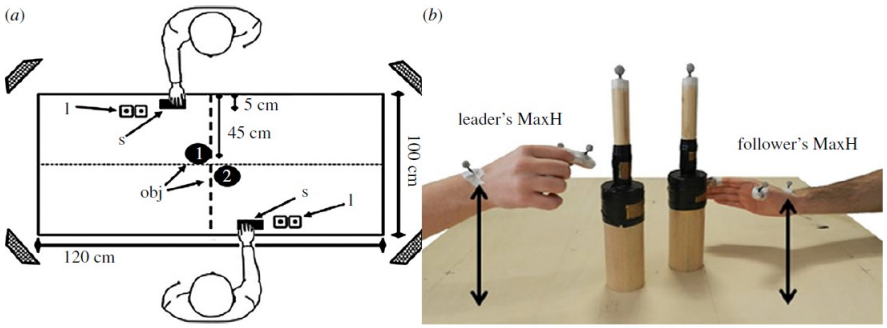

Figure 2.1: Experimental set-up (left), example of experiment (right) (1)

that the other person can estimate the weight of the object. More specifically they evidenced that, when an object is light, people tend to grasp it from the top; on the contrary, when an object is heavy, they grasp it from the bottom.

2.2 Communication in HRI

It has been shown what sensorimotor communication is, how it is used for human-human interaction and why it is important. Starting from these premises, it is possible to extend the principles to robots so that a sensorimotor communication between humans and robots holds, and the overall performances of the HRI are increased.

Sciutti *et al.* showed that humans can non-verbally communicate hidden object properties (e.g., the weight) (2). Depending on the vertical velocity peak and on the duration of the movement, it is possible to distinguish light objects from heavier ones: their experiment shows that, when a person grasps heavier objects, the time duration of the movement is longer and the velocity peak is smaller.
These considerations have been taken into account when designing the HRI. Through their experiment, which consisted in a person watching iCub (humanoid robot described in (28)) grasping objects of various weights, they concluded that humans can implicitly understand the weight of an object even though they know nothing in advance about it. However, the weight estimation is possible only if the robot motion respects the previous criteria of the human-human interaction: different peak velocity and different time duration for objects of different weights. This is shown in Figure 2.2; the three pictures describe the velocity of the motion of a human (top image) and iCub (middle and bottom images). The top figure proves that the velocity of the operation carried out by the human depends on the object's weight. The middle image depicts iCub performing a motion without taking into account the constraint on the weight-velocity. The bottom image rep-

resents the case where iCub follows the weight-velocity constraint. In a realistic

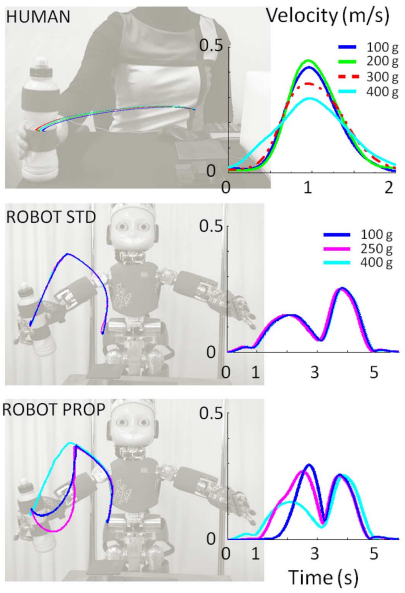

Figure 2.2: Velocities and trajectories of: human operator (top); iCub without respecting the motion criteria (middle); iCub respecting the motion criteria (bottom) (2)

HRI scenario, the robot should be able to understand:

- biological motions
- object fragility

Understanding biological human motions is very important in a real HRI context, where humans and robots work together in coordination, as it could happen in facilities.
However, to make it possible, it is necessary that the safety of the human operator is assured. This concept is very popular among people and it is known as the first Asimov law of robotics. To achieve this, the robot must understand whether or not a movement is being performed by a human being (29), in order to preserve the safety of the human operator.

Object fragility is a hidden property that may define the efficiency of the HRI. Indeed, in a smart home scenario, the robot should understand if the object grasped by the person is fragile or not, and then act as a consequence. The fragility property can be extracted by looking at how a person approaches an object: if he is careful the action will have different characteristics with respect to the ones typical of a careless action.

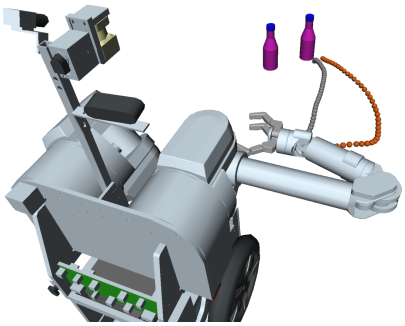

Figure 2.3: Predictable trajectory (gray) and legible trajectory (orange) (3)

These characteristics are non-verbally communicated among people through sensorimotor communication: it has been proved that a robot is able to understand if a person that grasps an object is careful or not (7).

With the goal of having an efficient HRI experience, it is not sufficient for the robot to understand verbal and non-verbal communication. In fact, even if the robot could respect these characteristics, the interaction could fall and be unpleasant. To avoid this, the robot's movement should be predictable and legible. According to the formalization performed in (3), predictability is a characteristic of movement that provides information on how similar the movement performed by a robot is to what a person expects. Indeed this concept depends on personal experience, which for example is different between children and adults.
On the other hand, legibility is referred to as the property of a robot movement to be understandable: the person can predict in real-time the motion goal, without waiting for the execution of the entire movement. Figure 2.3 shows two trajectories performed by the HERB robot (described in detail in (30)); from the figure is visible the difference between the predictable trajectory (depicted in gray) and the legible trajectory (orange).

As a last characteristic of the robot movement, it is worth mentioning the Uncanny Valley theory. It is a common hypothesis in robotics, formulated by Masahiro Mori (4), which relates the human likeness of a robot to the emotions felt by a human being. The concept is illustrated in Figure 2.4 (still line), where the x-axis and y-axis represent respectively the human likeness and the affinity felt by the person. From the plot, it can be noted how the human's affinity increases as the human likeness increases. However, in the proximity of what Mori called Uncanny Valley, the affinity felt by the human drastically decreases. In this region, the robot fails to be similar enough to humans, and the person feels unpleasant feelings.
According to Mori, if the robot moves then the Uncanny Valley is amplified

(dot line in the Figure) and the HRI quality is even worse. In conclusion, when designing an HRI is important to take into account these considerations.

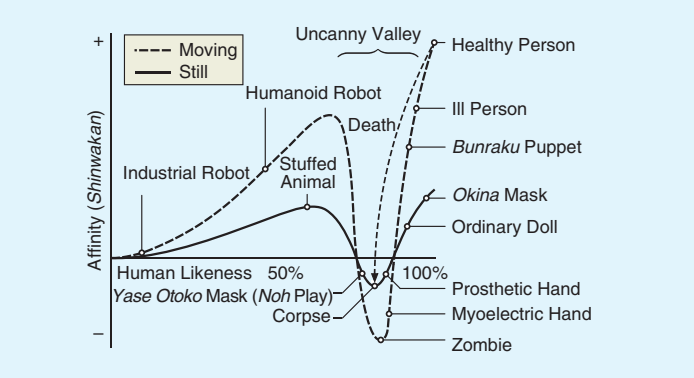

Figure 2.4: Uncanny Valley (4)

2.3 Gestures

The definition of gestures used in this scope is the one provided by Carfí *et al.* (31): "Gestures are trajectories $\tau(t_s, t_e)$ that humans intentionally perform to affect the behavior of an intelligent system".
This definition is focused on two aspects:

- the trajectory $\tau(t_s, t_e)$, where t_s is the start instant and t_e the end instant
- the willingness to perform a gesture

The trajectory is defined as a set of joint configurations, velocities and accelerations which allows to describe the human body with skeleton models.
The second characteristic concerns the fact that the human operator wants to communicate explicitly (in other words, he is willing to perform a gesture). As described in (31), a gesture may be characterized by several attributes such as:

- temporal duration
- level of instruction
- context of execution
- body part affected
- spatial influence

Among these, the most interesting ones are the temporal duration and the level of instruction. The first one allows distinguishing static gestures from dynamic

gestures. The word static is used to refer to movements in which the hand is still, fixed to a given joint configuration $\boldsymbol{q}$. On the contrary, a dynamic gesture is characterized by the fact that the hand configuration changes over time. Moreover, as happens in (7), this kind of gestures are usually characterized by three different temporal phases:

- a preliminary phase, in which the human operator brings the hand to a given initial configuration
- a central phase, where the execution of the gesture is carried out
- a final phase, which allows to bring the hand back to the initial configuration (or to bring it to another arbitrary configuration that may be useful for a next gesture)

Level of instruction refers to the information that a person needs to properly carry out a given gesture. According to Vuletic *et al.* (32), gestures can be classified as:

- prescribed
- free-form gesture

The first category indicates all the gestures that are part of a predefined dictionary. On one hand, these gestures have the drawback that the user has to adapt himself to the fixed dictionary. This clearly reduces the quality of the HRI, since the user cannot easily customize the interaction according to his habits. Moreover, in order to adapt himself to the dictionary, he has to focus and learn the available gestures, whose operation is more difficult with older people. On the other hand, prescribed gestures have the advantage of allowing the user to express symbolic meanings.
Free-form gestures have an opposite meaning with respect to the previous category: the user does not need to learn a new vocabulary and the robot will directly understand what gesture he or she is performing. As suggested by the authors of the article, even though this kind of gesture seems to be more useful rather than the previous one, they have the drawback of not allowing the user to communicate symbolic meanings.

Keeping these considerations in mind, to carry out an efficient HRI it is necessary that a proper analysis is carried out in the design stage.
The aim of such analysis is to understand whether prescribed gestures are more suitable than free-form gestures (or the contrary), so that the system is developed focusing on one kind of gesture rather than the other. Moreover, in case prescribed gestures are considered, a trade-off between the easiness of learning

the dictionary and the number of possible gestures must be reached, so that the mental stress of the user is minimized. In addition, the age of the target people for whom the gesture system is developed may change the weights in the trade-off: if the system is designed for young people, then the trade-off will mostly benefit the complexity of the dictionary.

Besides the previous classification, gestures can also be divided according to the effect they cause in the robot (31). This concept can be explained with the following example: consider a simple gesture like the rotation of the wrist, which is encoded by the robot as a non-verbal way to ask what is the current time. Now consider another hand gesture, which involves two fingers (for instance the thumb and the middle finger). Moving the fingers closer is interpreted by the robot as a command to get closer to the user. On the contrary, if the user moves the fingers away the robot will move away.
The first is an example of a discrete gesture: it is like a sample, which can only be interpreted by the robot once it is entirely carried out. Analogously, the second gesture is continuous: it is like a sequence of samples, in which at each time instant the robot associate a meaning to the current movement.
From this example, it is clear that discrete gestures are easily associated with prescribed gestures. Indeed, having a dictionary of movements allows to directly and easily associate a meaning to every action. In this perspective, the design stage previously described should also take into account the distinction between discrete and continuous, and then chose properly depending on the application considered.

Italian hand gestures represent a common way of communicating with our body. These gestures, which are used daily by Italians and have a discrete reputation throughout the world, are a form of explicit, culture-oriented form of communication, characterized by a gesture-meaning pair that is well-defined and shared among people. According to Poggi (14), Italian gestures are “coverbal” if they are strictly related to dialogue and enforce its meaning, or “autonomous” if they do not necessarily support the speech. Also, they can be described as “biological” or “cultural”. Biological gestures are closely related to certain situations, in which a person reacts psychologically to an external event. For example, a gesture like this may occur after a soccer player has scored a goal: for the moment of sudden happiness, he shakes his hands to the sky. On the other hand, cultural gestures are common gestures within a culture.

2.4 Hand model

In order to correctly understand how to recognize and categorize gestures, it is useful to know how to represent the human hand, especially when the classification is carried out through gloves, as in this study. In these cases, to recognize every finger movement, it is necessary to understand the key points of the movement so that the glove allows these points to be used as reference positions for the data acquisition.
The hand is a complex structure composed by the following kind of tissues:

- bones
- muscles
- tendons
- soft tissues

The bones are 27, and can be divided in carpals, metacarpals and phalanges (33). The muscles make the movement possible thanks to the tendons, which connect muscles to bones. Soft tissues include the skin and the adipose tissue.

In the literature, there are different skeleton models that allow to represent the human hand, and they differ in terms of model complexity. Usually, hand models (like the one provided in (33)), identity five type of joints in the human hand, that are named as:

- metacarpophalangeal (MCP)
- proximal Interphalangeal (PIP)
- interphalangeal (IP)
- distal interphalangeal (DIP)
- trapeziometacarpal (TM)

Another common aspect in hand models literature is to constraint the human finger configurations. Doing so, the hand is more realistic to the real anatomy. In fact, not all the finger configurations are possible, thus constraints are necessary in the representation. A complete description of the hand constraints is carried out in the model provided by Lee *et al.*, where four types of constraints are taken into account (34). As the authors suggest, the first constraint category regards geometric characteristics like the limits on the angle and the movement type. Then there are joint constraints on the interphalangeal and metacarpophalangeal movements (flexion). As an example of interphalangeal constraints, the authors refer to the difficulty to move separately the DIP and the PIP of the four fingers

(note that the thumb has a different structure). This difficulty is mathematically modelled through the relation:

$$q_{DIP} = \frac{2}{3} q_{PIP}$$

Speaking about the constraints on the MCP joints, the author noticed that the motion range is approximately 90 degrees for each finger, even though the range is a bit larger for the ring and the pinky fingers.
In (34) the model is characterized by 27 degrees of freedoms (DOF). A simpler kinematic structure is utilized in the gesture recognition carried out in (35), in which the skeleton is composed by 20 joints. In (36) the structure is even simpler and consists of only 23 DOF. This last structure is shown in Figure 2.5: the MCP of each finger has two DOF; the other joints of the fingers have one DOF at each; the joint at the carpus has 3 DOF. In addition, in Figure 2.6 is provided an x-ray image of the left hand, which can be used by the reader to understand the real joint positions.

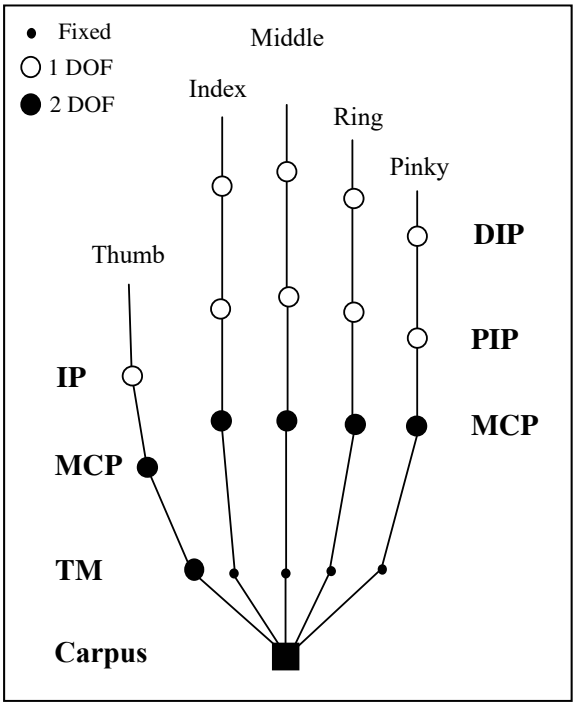

Figure 2.5: Kinematic structure of human hand (5)

Figure 2.6: X-ray image of right human hand (5)

2.5 Sensor technologies

As we will discuss in Section 2.7, the objective of gesture recognition is to establish to which class an input gesture belongs, analyzing, in probabilistic terms, how similar the input is to each of the gestures recognizable by the classifier. In the whole gesture recognition problem, the first operation that must be carried out regards the data collection. To do so, it is essential to use sensors. A sensor is a device that allows acquiring real-time information. In the context of gesture recognition, many sensors are adopted and each of them brings its own advantages and drawbacks. The most common devices used to acquire data are the following:

- marker
- single camera
- stereo camera
- depth sensor
- glove
- band

Markers are used in Motion Capture systems in order to reconstruct the 3D position of the point they are attached to. A single camera is a device that allows to

capture an image of the scene, while a stereo camera is a term used to refer to a couple of cameras that are positioned at a given distance. Depth sensors consist of a RGB camera, which captures colored images, and a depth sensor, that acquires depth images. It is composed of two units: the infrared projector, which is a laser, and an infrared camera. An example of a depth-image of 640×480 pixels is shown in Figure 2.7: the distance of the camera from each object in the scene is encoded with the gray color (darker gray pixels correspond to closer object), while the black color is used to indicate that no depth information are available for that specific pixel. Gloves and bands are devices that allow acquiring data such as: linear accelerations, orientation, angular velocity and magnetic fields. They rely on accelerometers, gyroscopes and magnetometers.
Among these devices, it is possible to make a distinction and categorize them as image-based and non-image-based (11). Image-based (marker, depth sensor, stereo camera, single camera) have a great advantage: the human operator does not need to wear any type of additional hardware (apart from the marker case). On the other hand, they bring some drawbacks: they involve a high computational complexity; they can have occlusion problems due to the anatomy of the fingers, which causes a not satisfying estimation of the hand pose (36); their performances may depend on environmental factors (like the lighting condition). Besides, vision-based sensors may not be the most appropriate choice for those who are particularly interested in protecting their privacy. On the contrary, non-image-based devices (gloves and band) do not have the occlusion problem; they principally rely on accelerometers, gyrometers and magnetometers, and allow to efficiently reconstruct the hand movement. Nevertheless, the user has to wear a specific hardware, which limits the simplicity of the interaction with the environment (36).
In Table 2.1 are provided in detail the most important advantages and drawbacks of some of the sensors used in gesture recognition.

Figure 2.7: Depth image of a person performing a gesture with the hand (6)

Table 2.1: Advantages and drawbacks of different sensors (11)

Sensor	Advantages	Disadvantages
Marker	Low computational workload	Markers on user body
Single camera	Easy setup	Low robustness
Stereo camera	Robust	Calibration difficulties
Microsoft Kinect	Fast emerging	Cannot be used for recognition over 2 m
Glove	Fast response, precise tracking	Cumbersome device with a load of cables
Band sensor	Fast response, large sensing area	Band needs to contact with human body

2.6 Classification methods

The classification methods can be grouped into two categories: those based on computer vision and those based on time-dependent sequences of accelerations. Template matching is a technique that works on visual features (e.g., skin color (11)) and belongs in the computer vision category. Usually, these algorithms perform the classification considering the Euclidean distance between the visual features of the input gesture and those of the template: if the distance is low enough then the gesture is considered classified as the template considered (37). The second classification category includes different learning algorithms such as: Dynamic Time Warping (DTW), Support Vector Machine (SVM), Feedforward Neural Network (FNN) Long Short-Term Recurrent Neural Network (LSTM-RNN).
DTW is an algorithm that measures how much two time series are alike. Through a cost function, it initially computes a cost matrix and then finds as the optimal warping path the one with the smaller cost (38).

SVM is a linear binary classifier that split into two classes the data contained in a given data set. The split is carried out with a hyperplane that is computed by maximizing the margin, that is the distance between the line separating the two classes in the data set and the data set entry most close to such line (39). The algorithm itself will not be very useful in HRI unless the gestures available are only two. Kernels can be used in order to allow the SVM classifier to classify more than two gestures.
FNN is a network whose elementary component is the neuron. The structure is

always characterized by at least three layers: the input layer, the output layer and the hidden layer in the middle (more than one hidden layer is permitted). Inside FNN, every neuron of a layer is connected to every neuron of the successive layer, and the connection is characterized by the *synaptic weight*. The values of the synaptic weights are computed during the training of the model through the back-propagation algorithm, which iteratively adjusts the weight values so that a given cost function is minimized. The algorithm is applied in the training phase, where the network takes as input a training set. After training, the network will have a certain generalization capability that will allow it to classify new unknown inputs. During the training, care must be taken to the overfitting problem, a situation where the generalization capability of the NN is lost. (40)

Recurrent Neural Network (RNN) is an architecture that is particularly useful when the data to be classified consists of a sequence of elements, as it happens with the speech. In RNN, the hidden layers contain information about the past of the current sequence in input, and this allows to model the time dependencies and predict the next input (41). The potential issue of RNNs is that they have difficulties modeling long temporal dependencies. For this reason, Long Short-Term Memory (LSTM) is considered. This kind of architecture can memorize longer temporal information through hidden units called memory cells (41). In Table 2.2 are summed up the most important advantages and drawbacks of the proposed classification algorithms.

Table 2.2: Advantages and drawbacks of classification approaches

Approach	Advantages	Drawbacks
DTW	temporal variability	complexity is $O(n^2)$, with n templates in the dictionary
SVM	high generalization capability, also with a small data set	Number of support vectors, binary classification method
HMM	allow to model temporal variable time series (42)	complexity is $O(n^2)$, with n number of states (42)
NN	user dependent, user independent, fast prediction (8)	training phase
LSTM-RNN	very efficient for sequential data	retrain if the dictionary changes, high number of weights (43)

2.7 Gesture recognition

According to Carfí *et al.* (10), gesture recognition involves the following steps:

1. Perception
2. Detection
3. Classification

The first sub-problem that occurs when dealing with gesture recognition is referred to as perception, where the goal is to acquire real-time data, which will then be used in the following phases. The data acquisition is carried out through sensors; as shown previously in Section 2.5, there are various technologies that are usually divided into image-based and non-image-based. In general, no sensor is better than the others: indeed the choice will be determinant for the implementation of the algorithm, since different sensors involve different features (i.e., accelerometers return accelerations, RGB cameras return colored images). The choice depends on the application for which that particular sensor will be more appropriate than the others.
Once the perception is performed, the problem is to extrapolate meaningful data from the one obtained by the sensor (detection). This problem occurs because it is not true, in general, that each recorded data corresponds to a time instant where a gesture is being performed. Consequently, it is necessary to identify, within a data sequence, where the gesture starts and ends.
Finally, the classification problem consists of establishing which gesture corresponds to the one whose data has been analyzed. As will be shown, there are different ways to solve the problem.

2.7.1 Perception

As explained before, the problem of perception refers to the acquisition of new data. Nowadays, there are multiple technologies that allow collecting data. Three-axis accelerometers make it possible to measure the linear accelerations along the x, y, z coordinates of a body to which the sensor is attached.
The advantages of accelerometers are essentially two: they are usually built-in smartphones and smartwatches and are accessible to everyone interested (44); they are quite cheap, allowing who is curious to develop new solutions, like gloves with built-in accelerometers (36).

Among image-based sensors, many researchers have focused their attention on RGB-D cameras like the Microsoft Kinetic (45). As explained before, this kind of sensor allows obtaining a color image and a depth image, with the advantage of not having problems due to lighting sensitivity or incorrect calibration, typical

of RGB camera. These sensors have been used to build frameworks that allow to recognize and interpret hand gestures of an operator (46), (47).

Elbert *et al.* have utilized a Kinect camera in a medical context, where the surgeon is able to control images relevant for the surgery without any classical input device (6). They implemented a kinect-based gesture recognition framework that allows the surgeon to interact with the medical images only with his or her gestures, with the advantage of not contaminating the patient during the operation.

Once data are acquired, whatever the sensor utilized for the perception, it is a common practice to pre-process the measured data in order to reduce complexity, which is due to the high number of input data, and noise introduced in the measurement.

Liu *et al.* proposed an efficient pre-processing algorithm that reduces both the noise introduced by the IMU and the computational complexity (9). Their algorithm is based on the quantization of raw accelerations, which works as follows: at first, the data in input is compressed through an averaging window of 50 ms (which is updated every 30 ms); then, a non-linear quantization is carried out, by giving more importance to the accelerations in the range $[-g, +g]$, where g is the accelerations of gravity. According to the authors, the choice of this specific interval is based on statistical observations of the accelerations values obtained experimentally: generally, such values are inside the well-defined range. For this reason, it is given more importance to the input data within the range. The output of this process is a shorter time series that, thanks to quantization, contains 33 possible values, as shown in Table 2.3.

Table 2.3: Acceleration quantization (9)

Acceleration data (a)	Converted value
$a > 2g$	16
$g < a < 2g$	[11, 15]
$0 < a < g$	[1, 10]
$a = 0$	0
$-g < a < 0$	$[-1, -10]$
$-2g < a < -g$	$[-11, -15]$
$a < -2g$	-16

In (48) the pre-processing algorithm consists of two steps: in the first one, a complementary filter is applied to the linear accelerations in input (49); the goal of the filter is to obtain accelerations that are only due to the hand motion (gravity is removed). In the second step, the filtered components of the linear accelerations are used to calculate the Euclidean norm.

Xie *et al.* in the pre-processing phase of their gesture recognition framework (8) use a different strategy with respect to the previous ones: in order to limit as much as possible the noise introduced by unwanted hand movements, the authors implemented a button strategy which consists of pressing a button just before performing the motion and releasing it right after its complete execution. Another characteristic of their pre-processing algorithm consist in removing the gravity acceleration from the input data a, obtaining the acceleration a_s that is only due to the hand motion. This is carried out by removing the mean acceleration value from each row data.
In accordance with the literature, the authors consider a filtering in order to remove noise, which is most evident in high frequencies. The filter considered is the symmetric Moving Average, a low pass filter that allows to cut-off high frequencies from $a_s[n]$, returning the filtered signal $a[n]$. It is defined as:

$$a[n] = \frac{1}{2M+1} \sum_{m=-M}^{M} a_s[n+m]$$

where $M = 5$ is based on empirical tests and comports a window of size of dimension $2M + 1 = 11$.

In (50), Bruno *et al.* filter the linear acceleration coming from the three-axis accelerometer with a Median filter (51), a non-linear filter again used to remove the noise in high frequencies. They also remove the gravity acceleration a_g from the row data a, by applying a low pass filter in order to isolate a_g, and then obtaining the acceleration due only to the hand movement as $a_s = a - a_g$.

2.7.2 Detection

As shown previously there are various sensors that allow to collect data, and they are mainly divided into image-based and non-image-based. The choice of the sensor will have an impact on the gesture recognition algorithm, since the two categories provide different data. However, the goal of the detection problem will be the same: to extract some meaningful features and prepare them for the gesture classification.
In computer vision, there are different ways to select some specific features like motion, color and geometric shape (52). The methods that allow obtaining such features are referred to as feature-extractors. Hasanuzzaman *et al.* developed a gesture recognition method based on Template Matching and skin-like regions (53). The features are extracted through a color segmentation algorithm that takes as input images expressed through the YIQ color space, where Y-channel is used to represent the illuminance and I, Q for the chrominance. Assuming that the only visible skin is in the hand, the gesture detection is carried out by

applying a threshold to the Y-channel, the one that allows to identify the skin from the background.

The detection problem arises also in the case of non-image-based sensor, since the raw data coming from the accelerometer does not give any additional information about the type of gesture the data refers to. In this context, it is very important to understand the time instants where a gesture starts and ends, and this is possible thanks to segmentation techniques.
If a person performs two given gestures, it is very likely that some random movements will occur between them. As consequence, the algorithm must be able to understand that two different gestures have been performed and what is between them does not correspond to any meaningful gesture. In the literature, different approaches that take into account this problem. They can be differentiated because some of them rely on manual segmentation, while others are automatically performed by the algorithm itself.

One example of manual segmentation can be seen in (48), where the procedure works as follows: right before the actor starts performing a gesture, an external operator presses a button and records that time instant t_s; when the gesture is entirely performed, the operator can release the button and the final time instant t_f will be recorded. A similar approach is considered in (54), with the difference that there is no external operator available to press and release the button. This task has to be carried out by the actor himself.
As pointed out by Luzhnica *et al.* the main drawback of manual segmentation approach regards the arbitrary of the process: the operator responsible to press the button, as careful as he may be, will never be able to capture the exact initial and final time instants of the gesture (48). This is the reason it is better in general to consider automatic segmentation algorithm.

There are various examples in the research community that involve automatic segmentation. One of them is in (7), where Lastrico *et al.* have proved that, when a person has to grasp an object that may or may not require carefulness, the hand velocity profile is always similar to the one provided in Figure 2.8. From such figure, it is possible to identify three segmented regions that are typical of this kind of movement. Note that the hand velocity profiles plotted in the figure have been computed in two different ways: one involving a Motion Capture system and another with the Optical Flow.

Another useful automatic segmentation algorithm is presented in (44), where accelerations are measured from an IMU sensor belonging to a smartphone. At first, the algorithm compute the distance D between each component of the current accelerations (time instant k) and the previous one (time instant $k-1$), as stated below:

$$D = \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2 + (z_k - z_{k-1})^2}$$

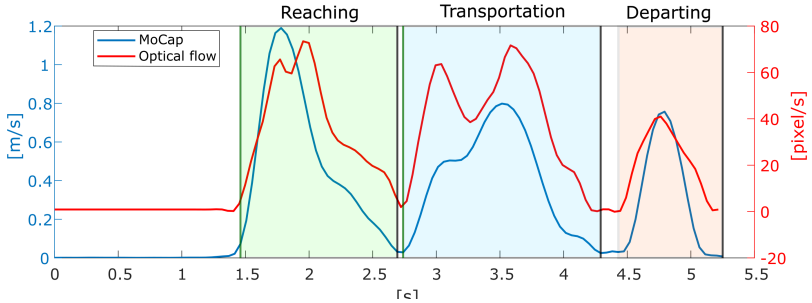

Figure 2.8: Reaching, Transportation and Departing phases of gesture segmentation (7)

then it checks if D is within a given range (chosen empirically). If $D > 0.3$ then the linear acceleration (x_k, y_k, z_k) is associated to the start of a gesture $(t_s = t_k)$; if $D < 0.1$ and its previous values were bigger than 0.3 then (x_k, y_k, z_k) is the accelerations corresponding to the final time instant $(t_f = t_k)$.
As the authors precise, the algorithm is efficient if the gesture has a temporal duration within the interval [0.6, 2] seconds. For this reason, the algorithm presents some lack of generalization capability, since in an online non-constrained scenario it may be the case that a gesture requires more than 2 seconds.

Algorithm 1: segmentation algorithm (44)

Result: t_s, t_f
while input data (x_k, y_k, z_k) **do**
  compute D_k;
  if $D_k > 0.3$ **then**
    $t_s = t_k$;
  end
  if $D_k < 0.1$ and $D_{k-1} > 0.3$ **then**
    $t_f = t_k$;
  else
    idle gesture
  end
end

A similar approach is carried out in (8).
Given $a[n] = (a_x[n], a_y[n], a_z[n])$, at first is computed the Euclidean distance between the current acceleration $a[n]$ and the previous one $a[n-1]$, where $a[n] = \{a[1], a[2], \cdots, a[n]\}$. Then a threshold $(0, 24525)$ chosen empirically is applied to select the initial and final time instants t_s, t_f. This algorithm holds since

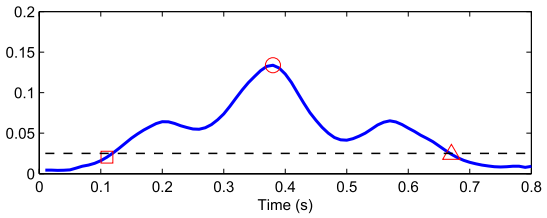

Figure 2.9: Euclidean distance evolution (8)

the authors made the assumption that the Euclidean distance is higher than the threshold only in case of meaningful gestures. In Figure (8) it is shown a graphical interpretation of the Euclidean distance evolution represented in the y-axis: the first square symbol highlights the time instant in which the gesture starts its execution; the red circle in the middle is in correspondence of the peak, and the final red triangle records the final time instant.

2.7.3 Classification

Gesture classification is a problem that arises when the goal is to understand the movement performed by a person. As seen before, it is typically preceded by signal pre-processing and gesture segmentation. Different techniques allow to solve the problem, by correctly recognizing and categorizing every gesture performed by an actor. These techniques involve models that may work on features directly extracted from the input data (both image-based and accelerometer-base method). From this it is evident that the first step in the gesture classification consists of the model building. As pointed out in (31), this phase does not follow a standard procedure. Usually, models work on some features extracted from the input, as it happens in (55) where the SVM model considers fast Fourier transform features (56), while the one in (57) considers Haar coefficients (58).
A common step in the model building consists in the dictionary definition. In FNN and LSTM-RNN this definition determines the training phase, where the classifier learns how to represent the model (generalization capability). Typically this is a long procedure (59) with a high computational time. On the other hand, in DTW algorithms the training phase is absent since they require only one gesture for each element they want to represent in the dictionary (9).
When building a model it is important to keep into account how the gesture classification will be used. If the system is designed for an individual, then it will be enough to develop a user-dependent algorithm. If more than one person is considered then the algorithm should be user-independent. The latter denotes algorithms where the accuracy is high even though the online classification is performed by a person that did not contribute to the model building.

In (9) Liu *et al.* implemented a DTW algorithm that performs classifications by minimizing the cost function between the input and each gesture in the dictionary. The algorithm is very efficient, as the high accuracy (98.6%) points out. The cost function is the Euclidean distance D between input $\mathcal{G}$ and dictionary template $\mathcal{T}_j$ computed for each acceleration $\mathbf{x}$ as:

$$D = \sqrt{(\mathcal{G}[x] - \mathcal{T}_j[x])^2 + (\mathcal{G}[y] - \mathcal{T}_j[y])^2 + (\mathcal{G}[z] - \mathcal{T}_j[z]^2)}$$

The dictionary is composed of 8 simple gestures and is shown in Figure 2.10.
The proposed algorithm works as follows: once a gesture $\mathcal{G}$ is detected, it is computed the Euclidean distance between $\mathcal{G}$ and $\mathcal{T}_j$ ($j \in [1, 8]$). At the end input $\mathcal{G}$ is classified as $\mathcal{T}_j$ if they present the highest similarity (thus smaller distance D). Figure 2.11 shows a graphical representation of the algorithm, focusing on three possible gesture distances.
As pointed out by the authors themselves, high accuracy is one of the most important advantages and points out a confident algorithm. On the other hand, the computational complexity grows exponentially as the dimension of the dictionary increases. This causes slower classifications, which may potentially compromise the quality of the HRI. In addition, due to the way the dictionary is updated, the classifications are user-dependent (57). Starting from the same gesture dictionary it is possible to increase the classification accuracy up to 99%, if a SVM classifier with a Gaussian kernel is implemented (57).

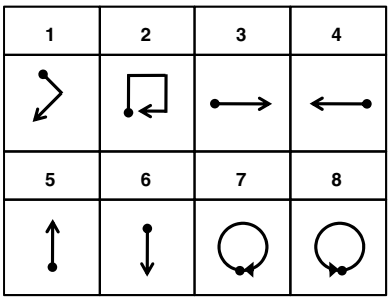

Figure 2.10: Gesture dictionary (9)

In (8), Xie *et al.* developed a gesture recognition system based on a device directly fabricated by the authors themselves. As usual, the device contains a three-axis accelerometer that allows to acquire real-time linear accelerations. The classifier used is a FNN, which consists of one input layer, one hidden layer and one output layer. The dictionary considered by the authors is composed of eight basic gestures and sixteen complex gestures and is represented in Figure 2.12. Note that the complex gestures are obtained as a combination of the simpler ones. On

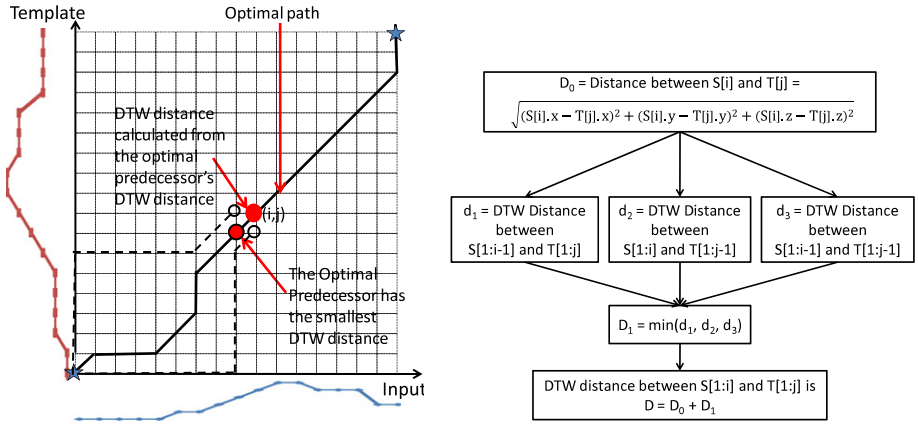

Figure 2.11: Graphical DTW algorithm (9)

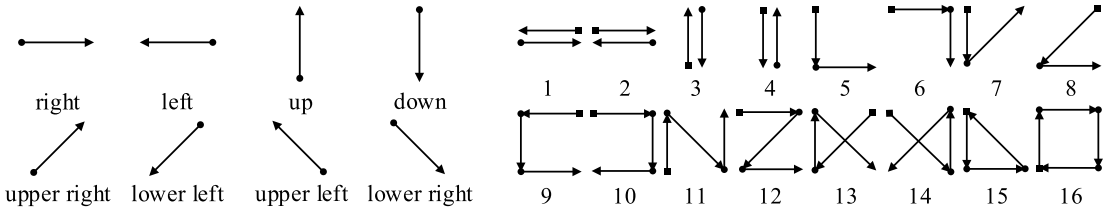

Figure 2.12: Gesture dictionary: basic gestures (left) and complex gestures (right) (8)

the assumption that two consecutive gestures are always at the minimum distance of 0.3 seconds, the authors tested the system accuracy for both user-dependent and user-independent cases. In the first case, the overall accuracy is 99.88% for both simple and complex gestures; in the last case, the accuracy goes from 98.88% (simple gestures) to 98% (complex gestures).

One example of continuous gesture classification carried out through RNN is developed in (10), where the network takes as input three-axis linear accelerations coming from an IMU sensor in the wrist. The input is then processed by the LSTM hidden layer, and finally a softmax output layer returns the probabilities associated with the input gesture. This architecture allows to continuously classify inputs with a dictionary of six predefined gestures (Figure 2.13). Of course the RNN module can classify input gestures only after the training phase, where gestures (that compose the dictionary) are performed multiple times by different people and then are fed to the classifier. In this specific case, a training set of 540 samples was considered.
During online classification, the accelerations of gesture $\mathcal{G}$ are given as input to

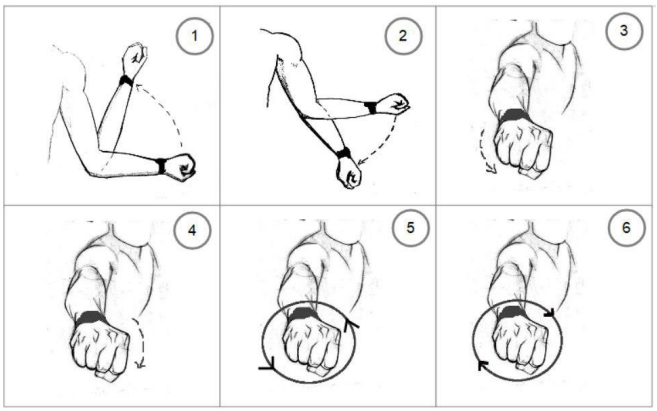

Figure 2.13: Gesture dictionary (10)

the classifier, which returns the probability of $\mathcal{G}$ being classified as one of the gestures in the dictionary $\mathcal{T}_j$.
In the testing phase, the authors obtained an overall accuracy of 99%, proving that in this context LSTM-RNN is a reliable probabilistic classifier. However, they also evidenced that such classifier must be retrained every time a new gesture is added into the dictionary. As a consequence, this limits the possibilities of the HRI, since training the classifier is a long and complex operation (59) that limits the generalization capability and the user customization.

Table 2.4: Summary of gesture classification approaches

Approach	Used in	user independent	user dependent	gestures in dictionary	Accuracy (%)
DTW	(9)	no	yes	8	98.6
SVM	(57)	yes	yes	8	99
FNN	(8)	yes	yes	25	98
LSTM-RNN	(31)	yes	yes	6	99

The provided algorithms are summed up in Table 2.2, where particular importance is given to characteristics such as user-dependency, user-independency, number of gestures in the dictionary and accuracy of the online classification. All models present a high accuracy, thus the gestures will be correctly classified most of the time. Indeed every model works with its own assumptions; for example, the choice of the dictionary will be determinant in the whole algorithm performances, since it has consequences on the model-building and determines characteristics like time delay and computational complexity.

Apart from DTW, all the classifiers are user-independent and allow to recognize gestures performed by other people than the ones needed to build the dictionary. This is particularly evident in FNN and LSTM-RNN methods as a consequence of the training phase. Even though this is a step that requires time, it allows the system to be user-independent because the dictionary is composed of gestures performed by different people.

Chapter 3

Experiment

In this section, we review the main aspects of the experiment, such as the description of the experimental protocol and the description of the Italian gestures considered. Furthermore, we will provide details about the custom-made inertial glove, as well as the criticalities introduced by it.

3.1 Inertial glove

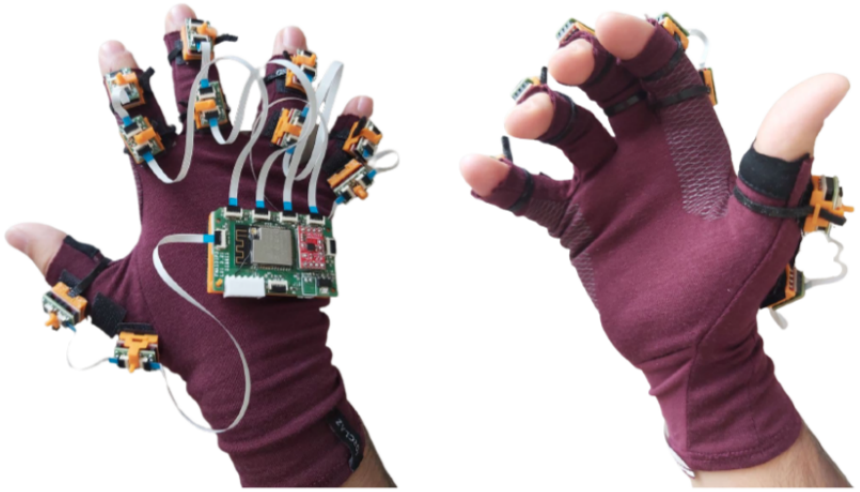

Figure 3.1: Custom-made inertial glove

Figure 3.1 shows the custom made inertial glove from both sides. As you can see, the glove has two Inertial Measurement Units (IMUs) for each finger. In the thumb, they are close to the metacarpal and intermediate phalanges. In all the others, the IMUs are always put on the proximal and intermediate phalanges.

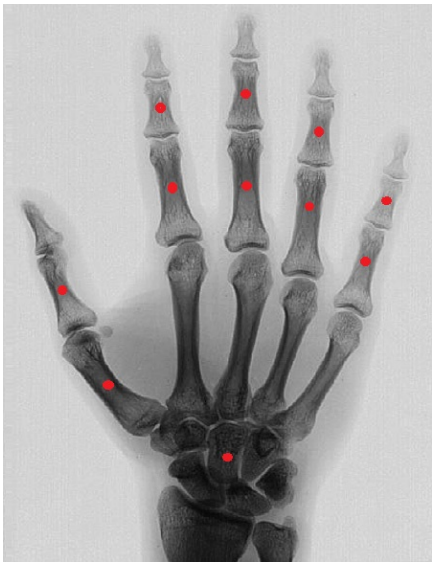

Figure 3.2: Positioning of IMUs in the hand phalanges

Moreover, an additional IMU is placed on the back of the hand (on the metacarpal bones). To get a better idea of where the IMUs are located, the reader can refer to Figure 3.2.

The acquisition of inertial data is carried out with the InvenSense MPU-9250, a nine-axes (gyro + accelerometer + compass) MEMS Motion Processing Unit, which allows to record the linear acceleration, angular velocity and orientation of the body where it is attached to. The MPU-9250 provides a user-programmable gyro full-scale range of ± 250, ± 500, ± 1000, ± 2000 deg / sec (dps) and a user-programmable accelerometer full-scale range of $\pm 2g$, $\pm 4g$, $\pm 8g$, $\pm 16g$, where g is the gravity acceleration. In total, the glove has 11 MPU-9250, located on the phalanges as described above.
The processing of inertial data is performed by the Esp32 located on the back of the hand. It is a microcontroller that interfaces with the MPUs to receive the inertial data and then transmits such data to a storage computer. The transmission is carried out through the Wi-Fi module integrated in the Esp32. Details on how it works will be provided in the following chapters.

Each IMU records ten time series with a frequency of 28 Hz: the triaxial linear accelerations, the triaxial angular velocities and the four orientations expressed as unit quaternions. Having 11 IMUs, we can collect a total of (i) $11 \cdot 3$ linear acceleration components, (ii) $11 \cdot 3$ angular velocity components, (iii) $11 \cdot 4$ orientation components. Because every phalanx generates different data, each of the 110 features is associated with the name of the phalanx to which it belongs. This leads to the 110 features, all different from each other, described in Table 5.1.

Table 3.1: Dataset features

Base	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Thumb proximal	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Thumb intermediate	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Index proximal	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Index intermediate	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Middle proximal	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Middle intermediate	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Ring proximal	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Ring intermediate	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Pinkie proximal	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Pinkie intermediate	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w

The table groups each acceleration a, angular velocity ω, orientation θ components according to the phalanx (on the left) it belongs to, i.e., *base*, *proximal*, *intermediate*.

3.2 Gestures dictionary

In this study, we consider twelve of the most popular Italian hand gestures, as shown in Figure 3.3 along with the gesture ID. Note that the names of the classes are omitted for simplicity, but can be observed in the video at the following link [1], where we show how to perform the gestures. The images in Figure 3.3 are the same used during the experiments. As stated in the literature, each of them has a specific social meaning (12) - (15), well-encoded in the Italian culture:

- *Let's go away*, a gesture to be used when one wants to communicate the intention to leave a place or a situation; it may express one's wishes, be a request to a friend or an unpleasant invitation to an acquaintance.
- *A drink*, gesture that indicates to those present that the person making the gesture is thirsty and would like to drink.
- *Very good*, used to describe to others that what you are eating, or have eaten, is to your taste; it is often used talking with children.
- *Bye*, a very common gesture, used to greet someone.
- *What do you want*, a very popular Italian hand gesture; literally, it means "What are you doing"; however, it may have an ironic reading, i.e., "What

[1] Web: https://youtu.be/PFiZEmKKo-Y

are you doing?! Don't do that!"; depending on the circumstances and the degree of impatience expressed, the hand may be held still or be shaken more or less violently up and down.

- *What a bore*, a lesser known gesture used to express boredom and tiredness with something or someone; it usually has a negative connotation.
- *It's not possible*, gesture that means "nothing to do" and expresses the impossibility of accomplishing an objective and/or carrying out an operation.
- *Fear*, a gesture very similar to "What do you want", both in terms of the hand movements and the literal/ironic interpretations; the literal meaning indicates that the subject is in a state of fear; the ironic interpretation is used to laugh at someone's unwarranted fear.
- *Silence*, a gesture used to explicitly invite someone to be quieter; it usually has a negative connotation.
- *Come here*, gesture indicating to the person you are interacting with to come closer.
- *Quotation marks*, a gesture indicating to other people that what the subject is saying is a quotation, from which they takes distance; it can also be used in sarcastic circumstances.
- *Victory*, a gesture used in sporting activities to express euphoria at a success.

Below, Table 3.2 provides a concise description of how to perform each of the classes of gestures considered. In addition, the table presents each gesture name in both Italian and English, and an ID column.

Table 3.2: Gesture dictionary description (12) - (13) - (14) - (15)

ID	English	Italian	Descprition
0	Let's go away	Andiamo via	The palm is turned inwards, the other fingers are flattened and the hand is waved up and down several times
1	A drink	Bere	The fingers are curved, like the shape of a glass, while the thumb is outstretched and suggests the flow of a liquid
2	Very good	Che buono	The index is extended and touches the cheek, the other fingers are closed and the hand rotates on itself
3	Bye	Ciao	The palm, with extended fingers, swings between left and right
4	What do you want	Cosa vuoi	The tips of the fingers are brought sharply together to form an upward-pointing cone
5	What a bore	Noioso	The hand, with outstretched fingers, beats repeatedly at the stomach level
6	It's not possible	Non possibile	The hand rotates around the index finger, keeping it extended together with the thumb, while the other fingers are closed
7	Fear	Paura	The fingertips open and close quickly and repeatedly
8	Silence	Silenzio	The index finger is laid across the lips, as if to keep them close
9	Come here	Vieni qua	With the palm facing upwards, the index fingers stretches and closes repeatedly
10	Quotation marks	Virgolette	The index and middle fingers move up and down in parallel, with both hands raised and palms forward
11	Victory	Vittoria	The hand is raised with extended index and middle fingers

Figure 3.3: Gestures in the dictionary along with IDs

3.3 Analytic gesture definition

Each Italian gesture can be described as a time-series evolution of 110 unique features. The length of the time-series, i.e., the number of *samples*, is generally different for every gesture, since it depends on the temporal duration employed by the participant to carry out such gesture.
As the IMUs have the same data acquisition frequency (that is 28 Hz), a given example is characterized by a fixed number of samples, equal for every feature.
The dimensionality of one single gesture in the dataset is:

$$\mathfrak{G} = [m, n]$$

with $n = 110$ the number of features and m the number of samples. Hence, the dimensionality of the dataset is:

$$\mathfrak{G} = [k \cdot m, n]$$

where k is the total number of examples carried out by all the participants and m is the number of samples. Figure 3.4 provides the time-series evolution of the features measured on the index intermediate phalanx, during the "What do you want" hand gesture.

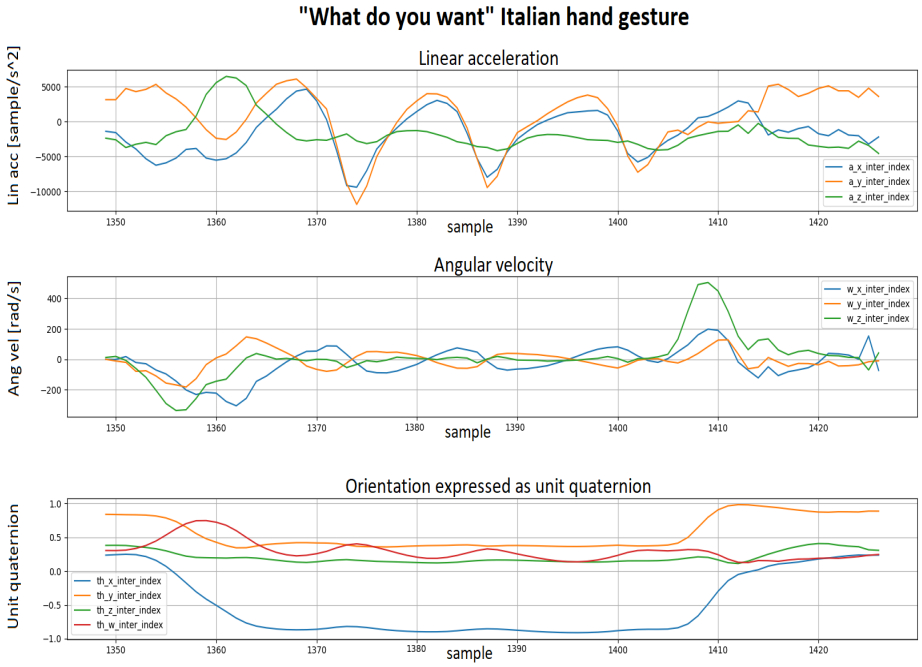

Figure 3.4: Index intermediate phalanx profiles of “What do you want” gesture

3.4 Room description

The room contains three key elements: a desk, where the participant is asked to put their hands on, a chair, which allows them to sit comfortably in front of the desk and iCub, a humanoid robot that guides the participant during the data collection. The position of the chair is adjusted to ensure that the angle between the arm and forearm is approximately 90 degrees. Note that the participant can put their right-hand in every area of the desk.
The human operator is not part of the scene, as a black veil completely separates them from the data collection station, and makes them invisible to the participant’s eyes.
A monitor is placed on the desk, on the right of the participant. It is intended for a visualization task, later discussed. On the other side of the desk, in addition to iCub, there is also a webcam that records the scene. The reader may refer to Figure 3.5 to have a better idea of the key elements in the room.

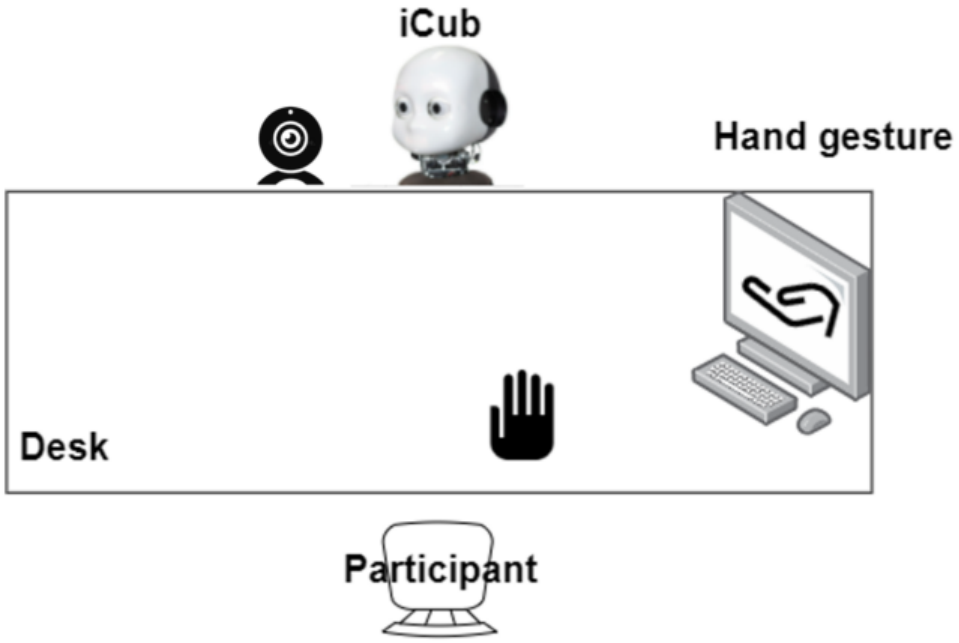

Figure 3.5: Room description

3.5 Experimental protocol

The volunteer enters the room accompanied by the operator responsible for the data collection. At first, they ask the volunteer's personal information (name, surname, date of birth, dominant hand); then they briefly describe the key elements in the room and explain to the participant that they will have to perform twelve gestures selected from the Italian culture. In addition, the operator explains that the management of the experiment will be totally entrusted to iCub, which guides the participant in the gesture collection. More specifically, due to the strict correlation between the Italian hand gestures and their social meaning, iCub will always introduce each new class by vocally providing a social context where the gesture is potentially useful; this to preserve at maximum at possible its social meaning. Each gesture will be recorded once at a time, i.e., gesture by gesture. To record the hand gestures data, the participant needs to wear a custom-made inertial glove. Then, the operator helps the participant to wear it correctly, on their right hand, and ask them to put their hands on the surface of the desk. The operator checks whether the wireless connection between the glove and the computer (where data is collected and stored) works correctly and finally he leaves the data collection station.

Each class of gestures will be recorded in two consecutive phases of about 20 minutes each, where we collect, for each phase, 4 examples for every class. In the first session, the participant has to perform the gesture without being socially influenced. iCub explains which gesture should be executed simply by showing a picture of it on the monitor (as those shown in Figure 3.3). Moreover, the robot provides a context for the gesture by explaining, in a short speech, a circumstance where the gesture in the picture could be performed. We will refer to this first session as "no-priming", since no complete gesture execution is shown to the volunteer. To clarify how the first session is carried out, here follows an example: after recording four gesture performances of the first class, the collection is temporarily stopped and iCub shows the volunteer the next gesture to be recorded (again randomly chosen and always by providing first an illustrative image and a brief social context by explaining a possible scene where to use the gesture). Afterwards, the volunteer performs four repetitions of the proposed gesture, the beginning of each one signaled by a sound. This procedure is repeated until all the 12 classes are collected, with four examples each.
The second session includes a *priming phase*, where the volunteer is explained how to *correctly* (i.e., in a more standard way) perform the gesture. This is achieved by showing, on the monitor, a video recording of a person performing the gesture, thus implicitly giving relevant information (i.e., velocity, acceleration, angular velocity, temporal duration, number of repetitions). The advantage of providing images and videos of the gesture is that all participants have exactly the same information needed to perform the gestures.
Similarly to the previous scenario, all the videos are introduced by iCub, which explicitly asks to repeat every gesture. Note that the videos provided for the participants are always the same and each gesture in the video is performed by the same person.

3.6 Further details

The experiments were conducted within 18 days at the "Istituto Italiano di Tecnologia" (Genoa, Italy), by involving one to three people per day. As previously mentioned, each experiment lasted around 55 minutes, where the first 15 ones were devoted to the description of the experiment and the informed consent, while the rest to the actual experiment.
The collection process involved thirty-one Italian volunteers (19 males, 12 females, age: 29 ± 5 years). Each participant, which had a well-known knowledge about the Italian hand gestures, experienced the same human-robot interaction and performed every gesture eight times.

Among the participants, 23 worked at the "Istituto Italiano di Tecnologia and had already experienced interactions with the humanoid robot iCub. The 8 remaining participants had never experienced any interaction with iCub nor, more generally, with other humanoid robots.
The glove was always worn in the right hand. Occasionally, some gestures allowed to use both their hands repeating the same movements, but only the data coming from the right hand were collected. Among the participants, 29 were right-handed and only 2 were left-handed.

3.7 Error sources and management

Some examples were removed from the data set due to errors that occurred during the experiment. These errors can be classified according to the causes from which they originated. The first one is related to the glove wiring connections that, if not coupled correctly, lead to a drastic reduction in the frequency of the input data (i.e., from 28 Hz to 10 Hz). This occurred during an experiment in which the participant used to make very pronounced movements, causing the glove to collide vividly with the body and thus altering the wiring connections.
For the participants, a very common mistake was to start performing the gesture outside the predetermined time interval. As explained previously, this interval was signaled *acoustically* by playing a specific sound. However, it happened that some volunteers started to perform the gesture before the sound was fully played, thus causing a permanent loss of inertial data.
Regarding the *no-priming session*, we removed 73 examples that corresponds to 5% of the total. Figure 3.6 shows the number of examples for each participant. The x-axis represents the participants IDs (anonymously depicted in codes) that took part to the data collection; the y-axis depicts the number of examples collected for each participant. Figure 3.9 shows the number of examples removed for each participant (x-label) and for each class (colored legend).
Regarding the *priming dataset*, we removed 19 examples (1.27% of the total) and, From Figure 3.7, we can see the number of examples for every participant, while from Figure 3.9 the number of examples removed for each participant (x-label) and for each class (colored legend).
As pointed out from the previous figures, we can conclude that removing examples from the dataset does not compromise the balance of its classes.

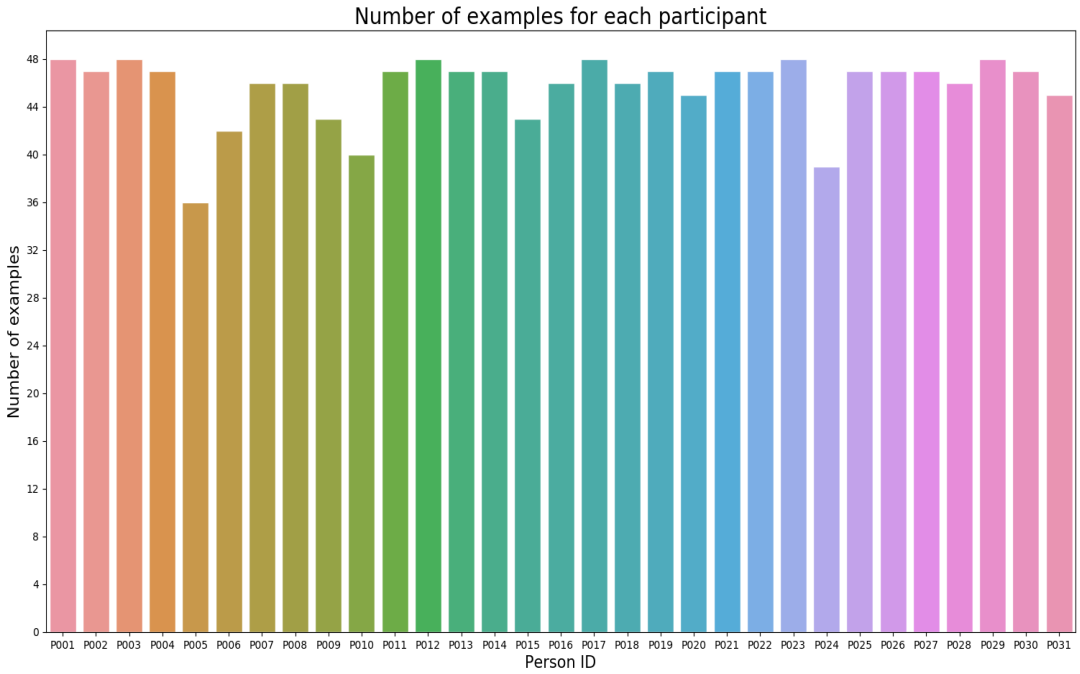

Figure 3.6: Number of examples (first session) generated by 31 participants

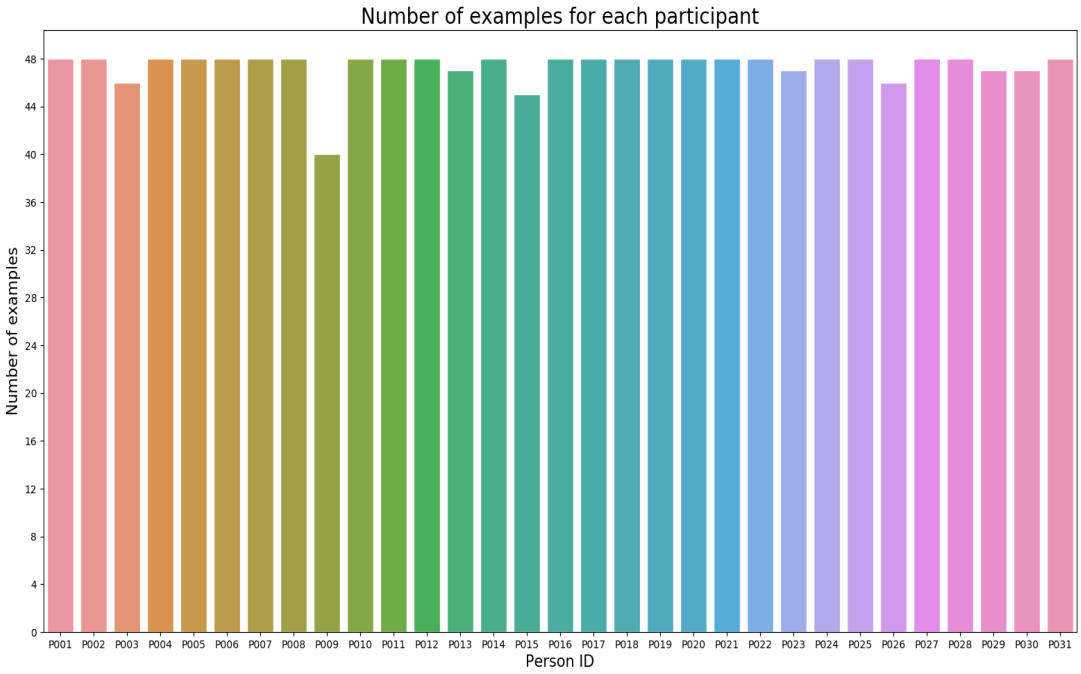

Figure 3.7: Number of examples (second session) generated by 31 participants

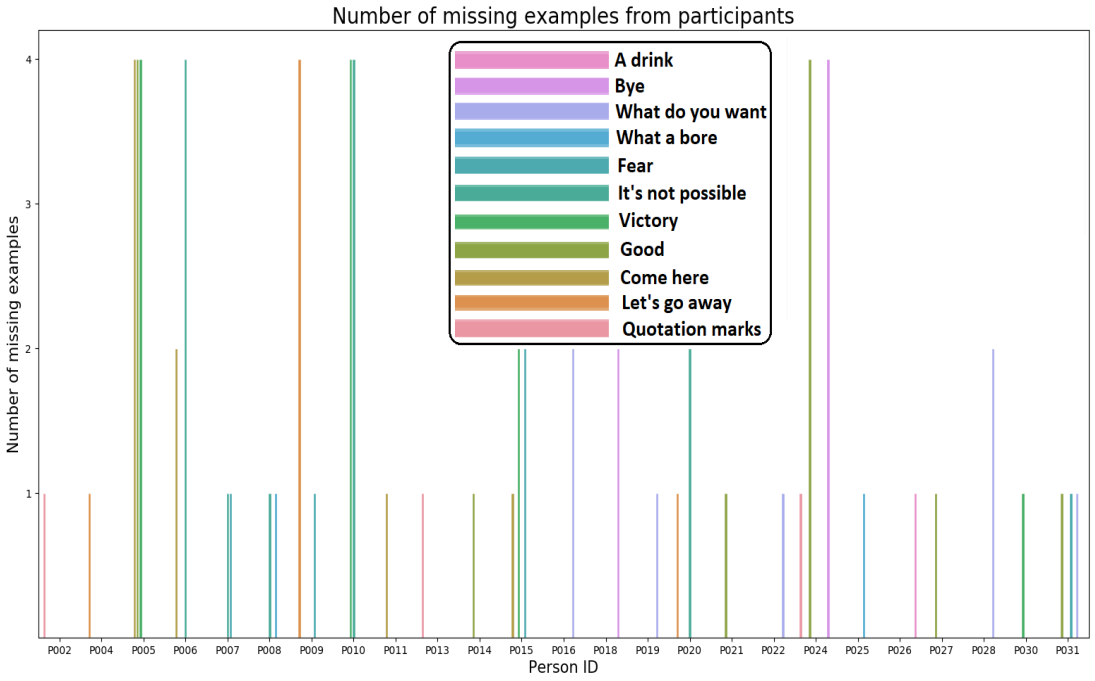

Figure 3.8: Number of removed examples (first session)

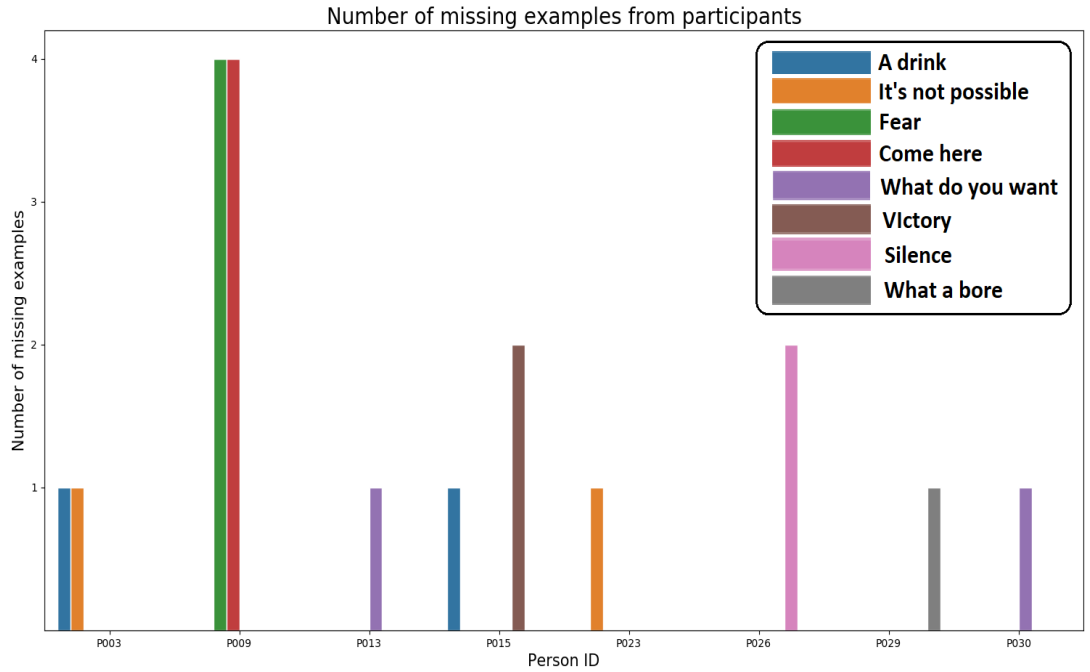

Figure 3.9: Number of removed examples (second session)

Chapter 4

Data acquisition architecture

This chapter explains the methods and tools used to perform the experiments, with particular attention to the underlying software architecture.
We developed the experiments to collect a dataset of Italian hand gestures. To achieve this goal, we used several tools, such as a custom-made inertial glove and iCub (60), a humanoid robot that replaced the role of the human operator responsible for data collection. In addition, we used two middlewares: Robot Operating System (ROS) (61) and Yet-Another Robot Platform (YARP) (62).
The overall system architecture can be divided into three macro blocks. Figure 4.1 shows the three modules that compose the system. On the bottom of the figure, it is depicted the first module, referred to as "ROS Module". As it will be explained later on this chapter, this module:

- Measures the inertial data during the execution of Italian hand gestures
- Transmits inertial information to the Main Module following a Publisher-Subscriber design pattern

The "YARP Module" is depicted on the top of Figure 4.1; this is an intermediate module that allows the Main module to properly control iCub during the experiment. Its main tasks are:

- Time synchronisation of the Main Module with the iCub architecture
- Management of the iCub behaviour (i.e., motion, facial expressions, voice)

The "Main Module", depicted in the left of the Figure, has the following main tasks:

- Management of the input messages
- Management of images, videos and sounds

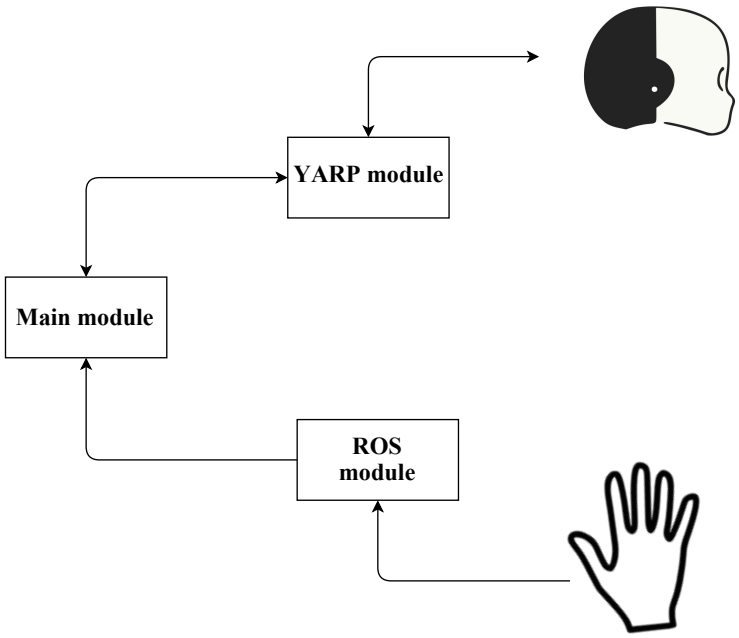

Figure 4.1: Graphical overview of the system architectures

4.1 ROS architecture

4.1.1 Prerequisites

Quigley et al. describe ROS as a language-neutral operating system with a micro-kernel design that allows to improve code re-usability and interpretability (61). ROS, which is free and open-source, supports different languages (e.g., C++ and python) and is characterized by a number of processes (also referred to as *nodes*) connected at runtime in a peer-to-peer topology. Communication between different nodes occurs in a TCP/IP-based transport, by passing strictly typed data structures, which are referred to as *messages*. Messages include primitive types (e.g., integer, floating-point) and more complex ones (e.g., Imu, PoseStamped, Twist). Also, messages are exchanged among nodes within named buses called *topics*. There are several ways in which messages are exchanged in ROS, like the Publisher-Subscriber Design Pattern. This mechanism involves the presence of two or more nodes called publishers and subscribers. The former publishes messages in a given topic, while the latter reads from it, in a decoupled way.

In the context of this Thesis, messages belong to the Imu type. As the name suggests, Imu messages hold data measured from Inertial Measurement Units (IMUs): orientation expressed as unit quaternions; angular velocity ($[rad/sec]$);

linear accelerations ($[m/s^2]$). Moreover, Imu messages also contain: "covariance fields", describing the covariance of the measured data for each of the three different inertial information; a header, a ROS standard message that contains a timestamp and an identifier. The covariance fields (named orientation covariance, angular velocity covariance and linear acceleration covariance) may not be expressed, if the measuring device is unable to calculate them (as is the case here).

4.1.2 Operation

The functioning of this sub-system is explained following the flow of information that occurs during the experiment.
The first step is the collection of inertial data, performed by the IMUs in the glove at the frequency of 28 Hz. The synchronisation and management of the raw data is carried out by the ESP32 MCU. As soon as data is available, the ESP32 transmits it via a UDP connection to a ROS node. This node performs a bridging function, since, once it receives the inertial data via the UDP connection, it constructs eleven IMU messages (one for each sensor) that are respectively published in eleven different topics (one for each IMU). Eleven subscribers subscribe to these topics, one for each topic. These nodes are responsible for: reading IMU messages as soon as they are available in the respective topic; accumulating and organising all messages referring to a certain type of gesture and a certain example; saving this set of data, organised in tables, in eleven different csv files (one for each phalanx/IMU).

4.2 YARP architecture

4.2.1 Prerequisites

Metta et al. describe Yet Another Robot Platform (YARP) as a C++ open-source projects based on the following principles:

- Modularity and multi-processing
- Code reusability
- Inter-process communication

These principles are strictly connected and allow to build robot control systems that are characterized by sets of location-independent modules, running on different machines and communicating among themselves according to one of the supported protocols (e.g., TCP, UDP, multicast).

Similarly to ROS, where modules are organized as collections of nodes that exchange messages to each other, YARP communication is organized in processing units called "Ports". A Port is an active object that manages multiple connections (either as input or output) distributed in a network of machines. Moreover, the communication supported by YARP is fully asynchronous. Ports can behave either as input, if they receive data from one or more ports, or output, if they send information to one or more ports. By default, as in the Publisher-Subscriber ROS scenario, YARP ports do not support replies to messages. However, this is possible considering Specialized Remote Procedure Call (RPC) ports.

4.2.2 Description

As introduced earlier, the role of this module is to make iCub capable of automatically managing the collection of the dataset. In other words, this module sends messages that allow the robot to speak, move and change facial expressions, and thus it is what makes the human-robot interaction possible. Communication developed to interact with the robot is based on RPC ports which, as described above, allow data to be sent and a response to be received. More specifically, three different ports are used.
The first to be described is the one aimed at making the robot talk. This port, which takes character strings as input, sends data to a specific port called "/acapelaSpeak/speech:i", which is part of a pre-defined module ("acapelaSpeak") that allows the text to be synthesised by the robot.
An important aspect of this port is that it is blocking: once the YARP module sends data, the execution of the program is interrupted until the response from iCub is received, which comes at the end of the synthesization. This is important because it allows to program predetermined robot behaviors (e.g. when iCub pronounces a certain word, it can make specific movements and/or facial expressions). It also allows respecting timings, as the execution of the program cannot continue until feedback has been received from the robot.
The second RPC type is the one that allows iCub to move the arm. It sends data directly to the robot's internal services that govern arm movements, which are "/ctpservice/right_arm/rpc" for the right arm and "/ctpservice/left_arm/rpc" for the left arm. It follows that two different connections are needed to control both arms of the robot. The type of message that is exchanged between the two ports contains the following information: time of the movement execution, target position (always null in this context), target orientation. Since the control of the arms include the hands, the degrees of freedom are 16, and thus the number of orientations the previous message is composed is 16. For the sake of completeness, Table 4.1 describes the degree of freedom of the right arm (those of the left arm represent a similar description and nomenclature).

Table 4.1: Right arm degrees of freedom description (16)

Joint	Description	Notes
0	Shoulder pitch	Front-back movement when the arm is aligned with gravity (post decoupling in firmware)
1	Shoulder roll	Adduction-abduction movement of the arm (post decoupling in firmware)
2	Shoulder yaw	Yaw movement when the arm principal axis is aligned with gravity (post decoupling in firmware)
3	Elbow	none
4	Wrist pronosupination	Forearm rotation along the arm principal axis
5	Wrist pitch	when hand-wrist aligned with the arm principal axis: i.e., this is relative to the forearm (not necessarily to gravity). Decoupling made in firmware
6	Wrist yaw	Decoupling made in firmware
7	Hand finger adduction/abduction	None
8	Thumb opposition	None
9	Thumb proximal flexion/extension	Single tendon looped
10	Thumb distal flexion	Single tendon + return spring for extension spanning two physical joints
11	Index proximal flexion/extension	Single tendon looped
12	Index distal flexion	Single tendon + return spring for extension spanning two physical joints
13	Middle proximal flexion/extension	Single tendon looped
14	Middle distal flexion	Single tendon + return spring for extension spanning two physical joints
15	Ring and little finger flexion	Single tendon + return spring spanning six joints on two fingers

The actual control of the robot movement, which takes place in the joint space, is performed automatically, giving as input the previously described message, by the built-in iCub modules.
The last messages taken into consideration are those responsible for the robot's emotions, which are expressed in the face through the LEDs that make up the mouth and eyebrows of iCub. Again, an RPC port is used, which connects to a pre-fabricated port named "/icub/face/emotions/in". In this case, the data exchanged with the robot is composed of an array of three strings. The strings inside the array are part of a well-defined dictionary, which is then decoded inside the iCub emotion module to make the emotions visible.

Considering RPC ports allows iCub to perform exactly the same motions, emotions and speeches every time the experiment is repeated. This is a key element in the context of social gestures, because every participant experience exactly the same human-robot interaction, and thus, their gesture performances will not be influenced by different stimuli.
Besides emotions, expressions, arm movements and voice descriptions, we decided to introduce three other functionalities, to make the interaction between humans and robots more fluid. These are: Recognition and tracking, via gaze and facial movements, of the participant's face; eyes blinking; breathing movements. These functionalities has been already developed by some researchers at the Italian Institute of Technology.

4.2.3 Operation

As mentioned above, this module has the task of acting as a link between the main module and the iCub architecture. It receives encoded messages from the main module, which allow it to identify the behavior to be followed by the robot at that given time instant. Examples of behavior are: introduction of the experiment, introduction of the gestures to be carried out, conclusion of the experiment. These messages are then decoded by the YARP module, which associates each of them with a series of operations to be carried out by the robot (movements, speech). The YARP module then produces the messages necessary for the robot to perform the predetermined actions and transmits them to iCub via the RPC ports discussed above.

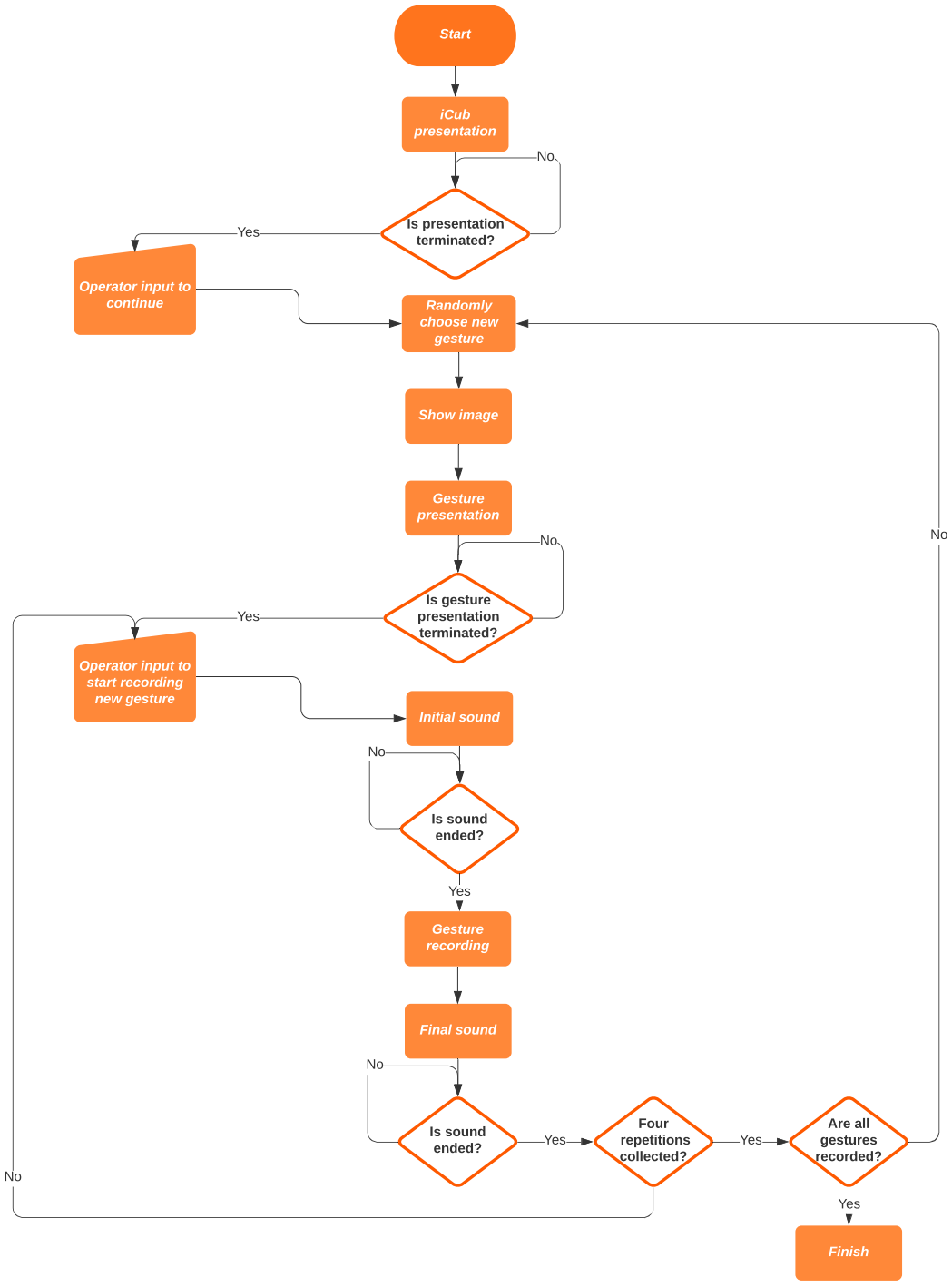

Figure 4.2: Flowchart of the experiment

4.3 Main architecture

4.3.1 Description

As described above, the main tasks of this module concern the correct evolution of the experiment, taking into account the inertial data in input and the behaviour that the robot must perform at each time instant. In addition, this module deals with less important tasks such as the audio-visual management of the experiment; it manages the images, videos and sounds that the participants observe during the experiment.
As described above, this module communicates with the "ROS module" via a "publisher-subscriber" design pattern, while it communicates with the "YARP module" via a dedicated port. Within the latter, unique codes are sent; these are divided into two types, depending on the meaning the message carries, referred to as:

- Contextual
- Gestural

Contextual messages are used to manage the initial and final phases of the experiment. In the initial phase, iCub introduces itself to the participant and describes what its role will be during the experiment: in the final phase, iCub informs them of the end and thanks them for their participation.
Gesture messages are used to communicate, during the experiment, which of the 12 types of Italian gestures should be performed. These messages contain an integer value belonging to the range $[0, 11]$, where each value is uniquely associated with a certain Italian gesture.

4.3.2 Operation

Figure 4.2 shows the flowchart of the experiment. It can be categorized into three main sections. In the first one, iCub briefly describes the tasks the participant has to complete. This is graphically depicted at the top of the figure in the block identified as "iCub presentation". In this phase, the YARP module, after receiving the corresponding message from the Main module, sends a series of predefined RPC messages to the robot.
As can be seen from the diagram, the initial phase ends automatically due to the "blocking" information reading mechanism, which prevents the algorithm from moving forward until the robot has carried out all the behaviors specified by the commands received.
As soon as the first phase of the experiment is over, the execution of the algorithm is momentarily interrupted, until the human operator resumes it by means

of a keyboard input. This is represented by the block at the top-left of the figure identified as "Operator input to continue". This mechanism, which allows to be sure that participants have understood how to carry out the experiment, implies the presence of a small waiting period in which they can ask for clarification on unclear aspects of the experiment.
If this is the case, the human operator keeps the experiment paused, to resolve the participant's doubts; otherwise, after a short wait of 4 - 6 seconds, the operator presses the key from the keyboard and then the experiment can proceed.
The second phase of the experiment is divided into two sub-phases, where the collection of the participant's inertial data is completed.
In the first phase, referred to as "no priming phase", the participant is asked to perform Italian gestures according to their experience. This means that they are free to make the gestures available in the dictionary, according to their own habits.
As depicted in the flowchart in Figure 4.2, this phase is repeated twelve times. For each of the twelve gesture classes, the algorithm randomly chooses one gesture at a time, which will soon be carried out by the participant. This choice is motivated by the preference to avoid introducing bias effects due to possible orders of gesture execution, which could occur, for example, if predominantly "static" gestures were executed before "dynamic" ones.
The next step consists in showing a photo of the gesture to be carried out. Below, Figure 4.3 depicts one of the images shown to participants during the experiment. The photo management is dynamic: each image is shown for 5 seconds, after which the robot provides context to the gesture, giving the participant a possible use-case scenario. The reasons for such a choice are twofold: one of the primary objectives of the experiment is to establish a human-robot interaction and, therefore, having the robot interact with the participant allows this aim to be pursued; the second is to help the participant understand which gesture they are asked to carry out. As a matter of fact, this may not be immediate from the images shown, which do not provide any information about gesture characteristics, i.e., execution time, speed, acceleration, number of repetitions.
Once the gesture presentation ends, the execution of the algorithm is automatically interrupted, following the same principle discussed above. In this case, however, the human operator waits a few seconds to check whether or not the participant has any doubts about the gesture to be performed. In fact, it could happen that the image and the context provided are not sufficient for the participant to identify which gesture to perform. In these cases, the participant, well aware that they can ask for information in these specific time intervals, can ask the human operator for clarifications about the gesture to be made. Otherwise, as usual, the execution of the algorithm was interrupted for a few seconds before the inertial data recording started.

Figure 4.3: Image of the "victory" gesture shown to participants during the experiments

As explained above, in this phase each gesture is recorded four consecutive times, each time by manually starting the recording for security reasons similar to those described above.
Each repetition has to be performed within a time interval of 6 seconds, signalled by a high-pitched start sound and a low-pitched exhaustion sound, as shown in the flow chart. It should be noted that, during preliminary studies of Italian hand gestures, their duration has been assessed to be between 1 and 4 seconds; the choice of giving 6-seconds does not constrain the participant to modify their gesture performances. It was deliberately chosen larger than the maximum duration for safety reasons.

The second part of data collection takes place immediately after all gestures have been collected four times each. Formally, this collection follows the same reasoning as the previous one: four gestures will be collected again for each class. What distinguishes this sub-phase, referred to as the priming phase, is the different way in which iCub asks participants which gestures to perform. In this case, iCub no longer provides a possible use-case scenario, but explicitly asks what gesture to do, i.e., calling the gesture class by its name; moreover, the image that was shown in the previous phase has been replaced by a video showing how the operator carries out the gesture. The participant, who is asked to carefully watch the video, can take in dynamic information (temporal duration, speed, number of oscillations); hence the term "priming", aimed at indicating a phenomenon

whereby exposure to one stimulus may influences a response to a subsequent stimulus.
The third and final phase of the experiment occurs automatically after all classes of gestures have been collected eight times each, four for the first and four for the second phase. During this phase, iCub thanks the volunteer for taking part in the experiment. The way this phase is implemented, from an architectural point of view, is similar to the previous one and is referred to as "finish" in the last flowchart block, at the bottom of Figure 4.2. The Main Module communicates, through the YARP port discussed above, to the YARP module that the end of the experiment has to be completed; after that, the YARP module uses the RCP ports to carry out the final behaviour of iCub.

Chapter 5

Features analysis

This chapter addresses an in-depth analysis of the dataset. At first, we will look for possible similarities in the way participants perform gestures. Subsequently, we will explore whether it is possible to reduce the number of features, e.g., by considering a subset of them.
For further reference, Table 5.1 shows the features that were collected during the experiments, where (x, y, z) are the components, a stands for linear acceleration, ω for angular velocity and θ for orientation expressed as unit quaternion.

Table 5.1: Features

	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Base	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Thumb proximal	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Thumb intermediate	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Index proximal	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Index intermediate	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Middle proximal	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Middle intermediate	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Ring proximal	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Ring intermediate	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Pinkie proximal	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Pinkie intermediate	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

5.1 Inter-class analysis

This section explores possible similarities in the way people carried out the gestures, identifying potential clusters in the dataset and investigating if there are any areas of particular overlapping.

5.1.1 Premises

As mentioned in the previous chapters, the number of features available is rather large, i.e., 110. This makes it difficult to visually interpret how features are related. To overcome this issue, we took a feature reduction approach, decreasing the number to 2 new features, built from the initial ones. These resulting features will be plotted in a plane to detect the presence of clusters.

To reduce the number of features, we consider the *Pairwise Controlled Manifold Approximation Projection* (PaCMAP) (63). It is a recent dimensional reduction algorithm that, compared to methods like UMAP (64) and TriMAP (64), allows to obtain a low-dimensional representation of the dataset, preserving each original observation's neighborhood (*local structure*) and each original relative positions of neighborhoods (*global structure*). PaCMAP is an iterative algorithm that works by minimizing a loss function that consists of three terms: one takes into account the contribution of the neighbors, one of the mid-near pairs and another of the further points.

Before carrying out the feature reduction, a clarification is necessary. As explained above, the complete execution of one gesture is defined by the *temporal evolution* of 110 features. The temporal dimension can be seen by looking at the gesture in Figure 5.1, where is shown the evolution of *one single feature*, i.e., the linear acceleration of the proximal phalanx of the index finger. The blue dots in the figure indicate the *samples*, sampled at a frequency of 28 Hz, while the continuous line is displayed to give a more concrete idea of the velocity profile. If we were to carry out feature reduction from this dataset, as it is the case in Figure 5.2, the result would not be very satisfactory, because each gesture would be entirely defined by a *sequence of samples*, as in the original dataset.
From a graphical point of view, it would be more difficult to understand the similarities between different classes, as there are many more points in the chart with possible overlaps.

To overcome these difficulties, we derived a new set of features, and by doing so, we removed the temporal dimension from the dataset: each repetition of a gesture is now a *single sample*, and not a *time-series* as in the original dataset. Furthermore, the flattening of the temporal dimension helped highlight similarities that were not evident with the original features, as we will see later.
The reduction in the number of features takes place in two distinct stages.

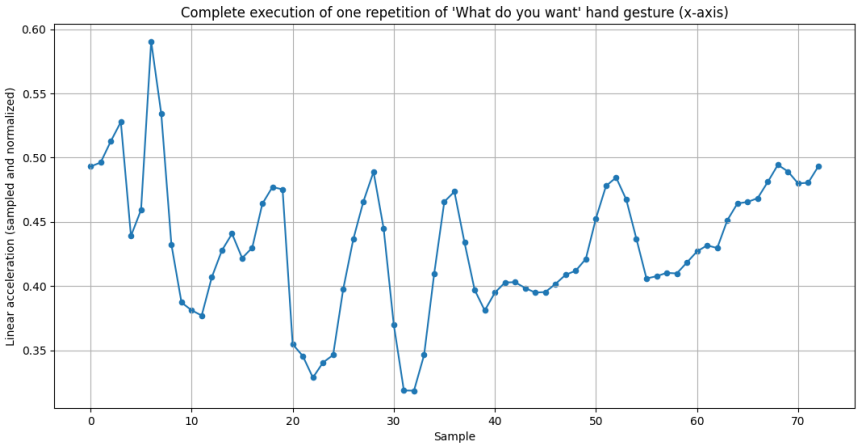

Figure 5.1: Linear acceleration (x-component) of the index proximal phalanx

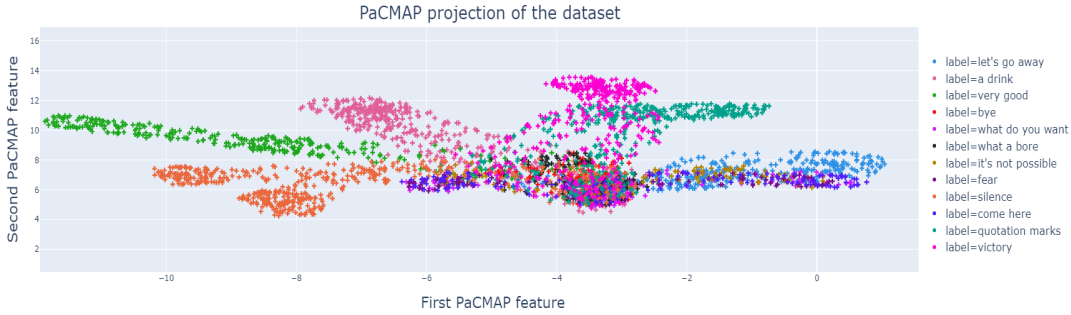

Figure 5.2: PaCMAP projection of the original dataset

In the first one, to remove the temporal dimension from the dataset, 11 features are computed from every original feature.
The dimensionality of one single gesture in the dataset is:

$$\mathfrak{G} = [m, n]$$

where $n = 110$ is the number of features and m is the number of samples that constitute that gesture (i.e., *temporal dimension*). Indeed the number of samples m is not generally fixed, as gestures do not have a fixed temporal duration.
During the initial phase, for each of the n features, we computed the following new ones: *mean*, *standard deviation*, *minimum*, *maximum*, *median value*, *variance*, *skew*, *kurtosis*, *standard error*, *mean absolute deviation*.

From this follows that, after the initial phase, each gesture is now characterized

by the following dimensionality:

$$\mathfrak{G} = [1, l \cdot n]$$

Where $l = 11$ is the number of new features to be computed for each of the original n features. As the dataset consists of the repetition of 2884 gestures, the overall dimensionality of the dataset is therefore equal to:

$$\mathfrak{G} = [2884, l \cdot n]$$

At the end of this first phase, the size of the dataset has considerably increased (10 times) along the column dimension (from 110 to 1210 features), while it has considerably decreased (190 times) along the time, row, dimension (from 461.440 to 2884).
In the second phase, the PaCMAP algorithm is applied to $\mathfrak{G}$, by setting its three hyperparameters:

— ***Number of neighbors***
considered in the k-Nearest Neighbor graph, here set to 5

— ***MN_ratio***
the ratio of the number of mid-near pairs to the number of neighbors, here set to 0.7

— ***FP_ratio***
the ratio of the number of further pairs to the number of neighbors, here set to 1.9

The output of the PaCMAP is a new dataset in which each gesture has the following dimensionality:

$$\mathfrak{G} = [1, 2]$$

Since the number of examples recorded was 2884, it follows that the reduced a-temporal dataset has the following dimensionality:

$$\mathfrak{G} = [2884, 2]$$

which is indeed greatly reduced with respect to the original one.

5.1.2 Clusters

The following analyses are carried out considering the dataset derived previously, which is now split into two distinct sets: one contains the *no priming* data, while the other the *priming data*.

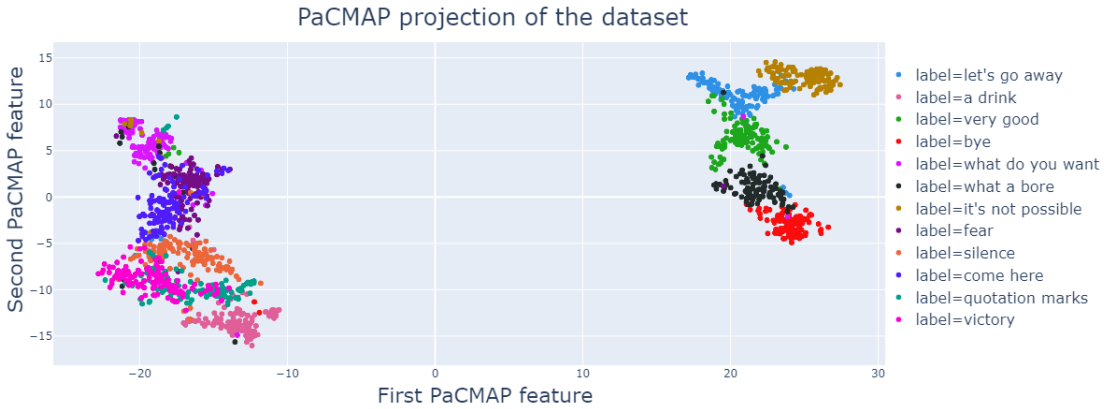

Figure 5.3: PaCMAP projection of priming dataset

In Figure 5.3, it is shown how one feature of the dataset, i.e., *First PaCMAP feature*, represented in the x-axis, varies with respect to the other one, i.e., *Second PaCMAP feature*, represented in the y-axis. The figure shows all the gestures performed by all the people; each point in the plane represent a complete execution *of one single gesture*, and is marked with a color that depends on the class the gesture belongs to. From the figure, it is possible to observe two well-defined clusters. The one on the right, referred to as *C1*, includes the most dynamic gestures, with well-defined swinging motions. They are: "Let's go away", "Very good", "Bye", "What a bore", "It's not possible". Such gestures do not have distinct movements of the phalanges of the fingers[1]; the oscillatory motions are generally the same for all phalanges and roughly coincide with those of the hand (metacarpal bones). The most evident example can be found in the "Bye" gesture, where the hand rotates with respect to the wrist, without any internal hand movement.
From the cluster on the left, referred to as *C2*, it is possible to observe gestures that, in a broad sense, are normally more static. These gestures are: "A drink", "Fear", "What do you want", "Silence", "Come here", "Quotation marks", "Victory".
Within this cluster, it is possible to carry out a further analysis, dividing it into two sets: one at the top and one at the bottom. In the upper part there are the gestures "What do you want","Fear" and "Come here", while in the lower part "A drink", "Quotation marks" and "Victory".
The main difference between these two clusters concerns the fingers that bring "more information"[2] during the hand movement. In the cluster at the bottom,

[1]The reader may refer to Table 3.2 for a description on the Italian hand gestures

[2]The fingers that bring more information are those that allow the gesture comprehension at a social level

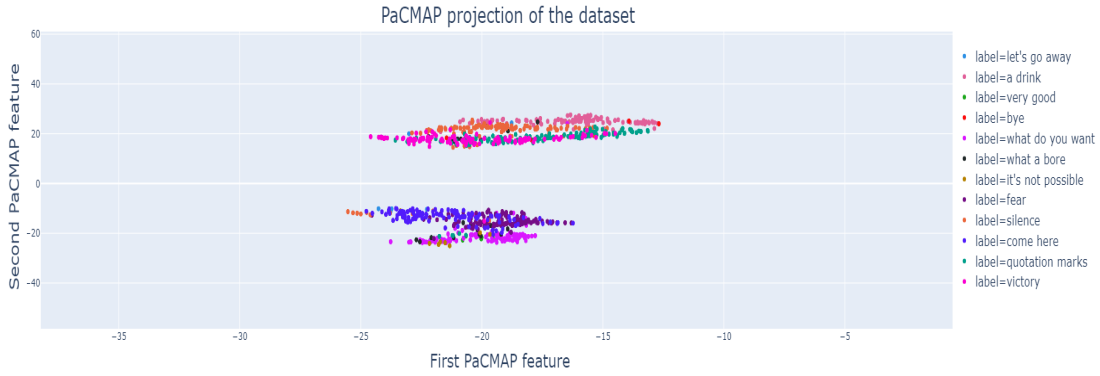

Figure 5.4: Focus on sub-cluster of Figure 5.3

gestures are all performed with index and middle as most informative fingers. In contrast, in the upper cluster, gestures include movements by several phalanges of the hand. This is not particularly evident for the gesture "What do you want", but it is for the gesture "Fear", where all phalanges open and closes repeatedly. By slightly modifying the parameters with which the PaCMAP algorithm is carried out, it is possible to highlight the distance between the two sub-clusters. Figure 5.4 shows the result of the PaCMAP giving as parameters: n_neighbors=5, MN_ratio=0.56, FP_ratio=2.9. Note that, compared to the figure 5.4, now the clusters are inverted: gestures performed with the index and middle fingers (e.g., "Victory") are now at the cluster at the top, while those that includes several phalanges (e.g., "Fear") are at the bottom of the figure. Nevertheless, the principle is the same: despite the presence of some outliers, the figure depicts much more clearly these two sub-clusters.

Let us now repeat this analysis considering the other dataset, collected *before* the priming phase. It should be remembered that, as described earlier in Chapter 3, this data collection was carried out without providing any external stimulus that could in any way influence the way participants perform gestures (*priming*). Participants had to reproduce gestures according to their experience and habits. With these premises, it is reasonable to assume that the variability of this dataset is much higher than the one of the other dataset, collected after the priming phase. This was evident from the beginning, simply observing how hand configurations and movements assumed during the gestures varied between participants.
Figure 5.5 shows the *no priming* dataset, First feature against Second feature, depicted in the x and y axis respectively. Again, the figure shows all gestures performed by all people, and every point in the plane represent a complete execution of one single gesture (different colours correspond to different classes).

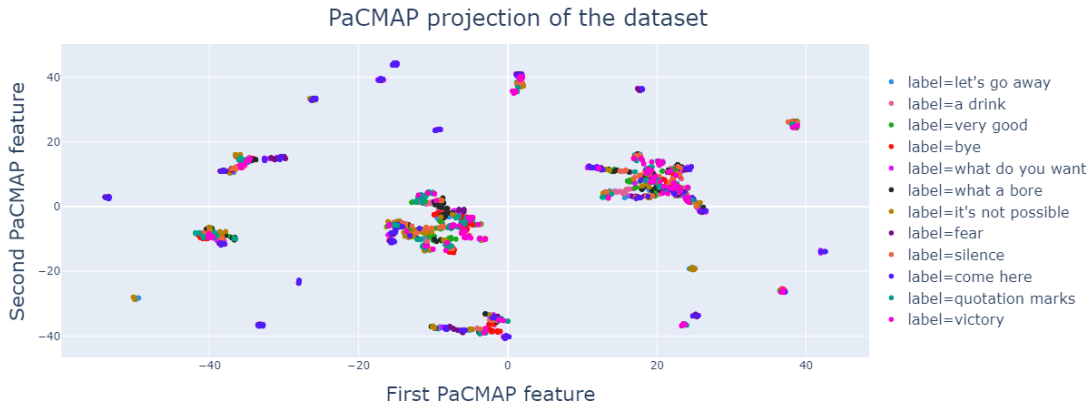

Figure 5.5: PaCMAP projection of no priming dataset

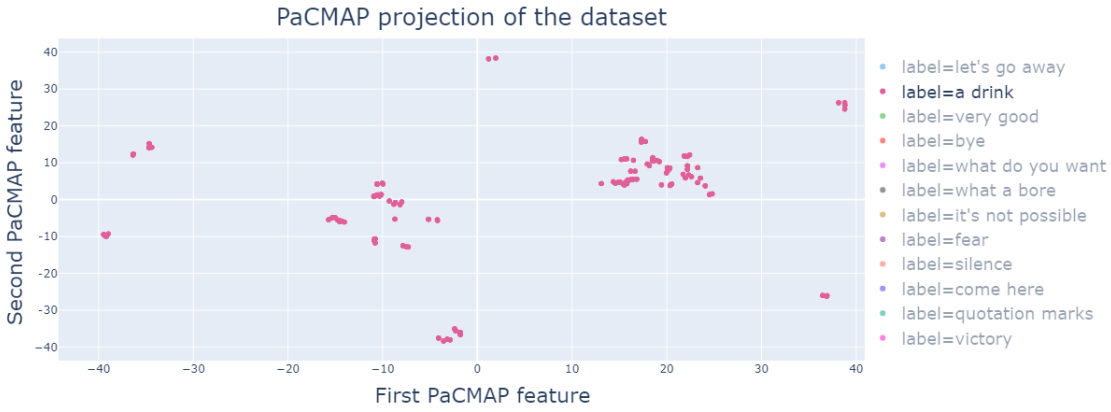

Figure 5.6: Focus on "A drink" of Figure 5.5

Looking at the figure, we can see the high variability of this dataset; unlike the data obtained after the priming, where clusters were definitely present, this is not the case here. There are certainly two main clusters, one in the middle and the other on the right of the figure. However, it is not possible to identify distinct zones, populated only by examples belonging to one single class. This aspect highlights the fact that gestures made by different people can be very dissimilar, since they are located in distant positions in the plane (as shown in Figure 5.6, where is provided a focus on the "A drink" gesture).

5.1.3 Inter class similarities

Let us now consider the figures above to see whether there are, in general, similarities between classes of gestures.
Figure 5.3 showed that gestures can be divided into two macro-categories: one of the most swinging gestures, and another whose gestures are mainly characterized by finger movements. In the first category, it can be observed that there is *no overlapping* between examples belonging to different classes. This aspect emphasizes that, when participants are asked to perform these gestures, they all behave similarly. Gestures from a given class are very similar to each other because they are very close in plane. Instead, gestures of different classes are more distant and thus dissimilar enough to be further clustered.
On the other hand, looking at the gestures in the cluster on the left of Figure 5.3, it is possible to identify some clear similarities. This is particularly evident for the two classes "Victory" and "Quotation marks", which are shown in Figure 5.7 (zoom of Figure 5.3). From the above figure, is possible to observe some overlapping between the examples of these two classes. This indicates two important aspects: the first is that "Victory" and "Quotation marks" are two very similar classes. This behavior was predictable, as the configuration assumed by the hand during these gestures was almost identical.
The second important aspect concerns the fact that the similarity between these two classes is evident for most of the participants. This allows to conclude that, globally, when participants are asked to perform these gestures, they all behave similarly and carry out these gestures in a comparable manner.

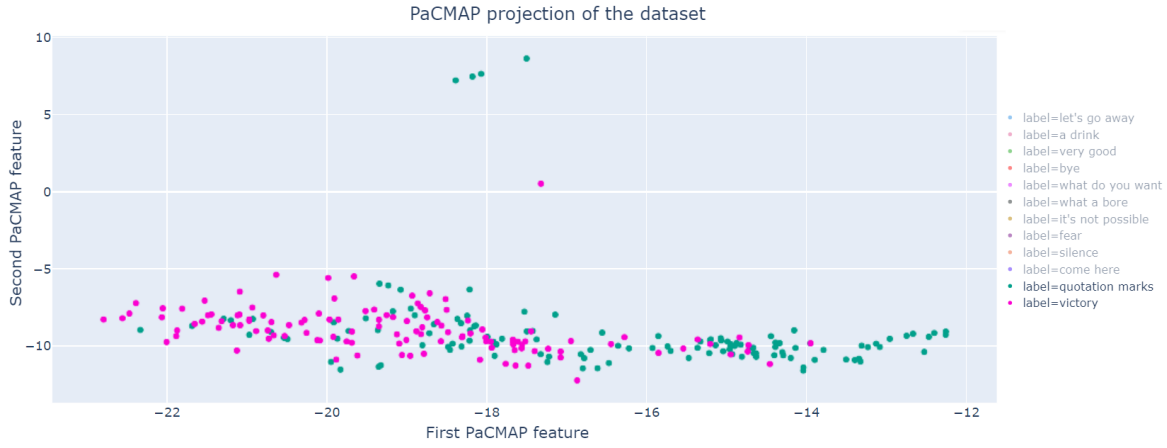

Figure 5.7: Focus on "Quotation marks" (in purple) and "Victory" (in green)

5.2 Feature Selection

What characterizes the dataset of Italian gestures is the high number of features, that is 110. This is a consequence of the large number of sensors used in data collection: we employed all the available IMUs because it was not clear, a priori, which features were the most informative. Now, understanding those that are the most informative features is important for two reasons:

- reduce the number of sensors employed for gesture recognition
- reduce computational complexity

In the previous chapter, we achieved this by reducing the dimensionality of the dataset through data-driven and model-based transformations (e.g., PaCMAP). However, it would be interesting to consider only some of the original features, without the need of applying any transformation.
This section discusses Recursive Feature Elimination, one approach that allows to determine a subset of the initial features. Therefore, reducing the *samples*, i.e., the number of rows of the dataset, is beyond the scope of this section.

5.2.1 Recursive Feature Elimination

Recursive Feature Elimination (RFE) is a feature selection algorithm that belongs to "wrapper" methods (65). They work by creating several models, with different subsets of input features, and selecting those that perform best according to given metrics.
RFE selects a desired number of features, by recursively considering smaller and smaller sets, where the least important features are pruned at every iteration through a ranking mechanism. This is possible considering an external estimator that, at each iteration, assigns weights to each feature.
As mentioned above, the RFE approach needs as input the number of features to be preserved. However, this may not be known a priori, as is the case here. To overcome this issue, RFE and cross-validation (RFECV) are considered. RFECV allows determining the optimal number of features through a cross-validation approach, where different subsets of features are assigned and the best collection of scored features is selected.

5.2.2 Feature selection

RFE is carried out two times, considering two different models: a Linear Regression and then a Random Forest Classifier. For the implementation, we used sklearn, a python library that includes all the elements needed for carrying out

this analysis.
Considering the Linear Regression model, we found the following features (Table 5.1:

Table 5.2: Features selected considering Linear Regression model

	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Base	✓	✓	✓	✓	-	✓	-	-	-	-
Thumb metacarpal	✓	✓	-	-	-	-	-	-	-	-
Thumb intermediate	✓	✓	-	✓	-	-	-	-	-	-
Index proximal	✓	✓	✓	✓	-	✓	-	-	-	-
Index intermediate	✓	✓	-	-	-	-	-	-	-	-
Middle proximal	-	✓	✓	-	✓	-	-	-	-	-
Middle intermediate	-	✓	✓	-	-	✓	-	-	-	-
Ring proximal	✓	-	✓	-	-	-	-	-	-	-
Ring intermediate	✓	✓	-	-	-	✓	-	-	-	-
Pinkie proximal	-	-	-	-	-	-	-	-	-	-
Pinkie intermediate	-	-	-	-	-	-	-	-	-	-

“ - ” indicates the features that are no longer part of the original features set (Table 5.1)

On the other hand, considering a Random Forest Classifier, we obtained the reduced features shown in Table 5.3.
Generally speaking, both feature reduction approaches allow to significantly reduce the number of features. In fact, it is possible to consider 10 IMUs out of 11. In terms of individual features, the linear regression model allows 28 features out of 110 to be considered (-74.5%), while the random forest 37 features (-66.3%).

Comparing the features extracted from the two models, we can observe some elements in common and some differences. Both tables show that the features produced by IMUs in the pinkie are not relevant, as they are never selected by the RFE algorithm. The most informative features are the linear accelerations. More specifically, those of the base-index (proximal phalanx) are equally important, as selected by both approaches. Regarding the other accelerations, we can observe some differences: in Table 5.2 (RFE model: linear regression) the z-component of the linear acceleration of the index (intermediate phalanx) is excluded from the set, while this is not the case in Table 5.3 (RFE model: random forest). Another very distinct difference concerns the choice of angular velocities compared to that of orientations. In the first case (linear regression), there is no quaternion among the selected features; on the contrary, considering a Random Forest as a

Table 5.3: Features selected considering Random Forest model

	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Base	✓	✓	✓	-	-	-	✓	✓	-	✓
Thumb metacarpal	✓	✓	✓	-	-	-	-	-	-	-
Thumb intermediate	✓	✓	✓	-	-	-	-	-	-	-
Index proximal	✓	✓	✓	-	-	-	✓	✓	✓	✓
Index intermediate	✓	✓	✓	-	-	-	✓	✓	✓	✓
Middle proximal	-	✓	✓	-	-	-	-	-	-	-
Middle intermediate	✓	✓	✓	✓	-	-	-	-	-	-
Ring proximal	✓	-	✓	-	-	-	-	-	-	-
Ring intermediate	-	✓	✓	-	-	-	-	-	✓	-
Pinkie proximal	-	-	-	-	-	-	-	-	-	-
Pinkie intermediate	-	-	-	-	-	-	-	-	-	-

- indicates the features that are no longer part of the original features set (Table 5.1)

model, there is no angular velocity (apart from one single outlier). This aspect underlines that the choice of the features may depend on the model considered. Nevertheless, it can happen that the angular velocity and the quaternion orientation features refer to the same phalanges: in the base and in the index, the linear regression selects the angular velocities, while the random forest selects the orientations. This aspect underlines again the importance of such phalanges for gesture recognition.

5.2.3 Feature selection on static and dynamic clusters

In section 5.1.2, it was shown that gestures can be divided in two main clusters, below repeated for simplicity:

- *C1*, characterized by gestures with swinging motions ("Let's go away", "Very good", "Bye", "What a bore", "It's not possible")
- *C2*, characterized by more static gestures ("A drink", "Fear", "What do you want", "Silence", "Come here", "Quotation marks", "Victory")

In this section, we will repeat the RFE analysis considering firstly the gestures in *C1*, and then those in *C2*, with a random forest model. Note that the features extracted during these analyzes are compared with those in Table 5.3, henceforth referred to as the *reference table*. In the reference table, as previously discussed, features have been extracted considering the entire dataset and a random forest model. The purpose of this analysis is to confirm the different characteristics between the classes in the two clusters.

Table 6.1 shows the features extracted by the RFE algorithm, considering only those gestures that are part of *C2*. It is possible to observe some differences from the reference table 5.3. In particular, the following features are not included in the new subset:

- the **x**-component of the linear **acceleration** of the *thumb* and *index* finger (*intermediate* phalanges)
- the **y**-component of the quaternion **orientation** of the *base* and *index* finger (*intermediate* phalanx)

On the other hand, there are new features, not initially present in the original table, which are:

- **x** and **y** components of the **angular velocity** of the *index* finger (*proximal* phalanx)
- **z**-component of the **angular velocity** of the *ring* (*intermediate* phalanx)
- **y**-component of the quaternion **orientation** of the *middle* (*proximal* phalanx)

Table 5.5 shows the features extracted by the RFE algorithm, considering only those gestures that are part of *C2*. In this case, the features missing from those extracted in the reference table are:

- **x**-component of the linear **acceleration** of the *thumb* (*metacarpal* phalanx), index and ring (proximal phalanges)

- **y**-component of the linear **acceleration** of the *thumb* and *ring* (*intermediate* phalanges)
- **z**-component of the linear **acceleration** of the *ring* (*proximal* phalanx)
- **x**-component of the **angular velocity** of the *middle* (*intermediate* phalanx)

On the contrary, new features have been added, corresponding to:

- **x** and **y** components of the quaternion **orientation** of the *thumb* (*metacarpal* phalanx)
- **z**-component of the *middle* (*proximal* phalanx), *base* quaternion **orientations**
- **w**-component of the quaternion **orientations** of the *middle* (*proximal* phalanx), *ring* (*inter* phalanx).

Table 5.4: Reduced features, computed giving as input the "static" dataset

	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Base	✓	✓	✓	-	-	-	✓	✖	-	✓
Thumb metacarpal	✓	✓	✓	-	-	-	-	-	-	-
Thumb intermediate	✖	✓	✓	-	-	-	-	-	-	-
Index proximal	✓	✓	✓	-	-	-	✓	✓	✓	✓
Index intermediate	✖	✓	✓	☑	-	☑	✓	✖	✓	✓
Middle proximal	-	✓	✓	-	-	-	-	☑	-	-
Middle intermediate	✓	✓	✓	✓	-	-	-	-	-	-
Ring proximal	✓	-	✓	-	-	-	-	-	-	-
Ring intermediate	-	✓	✓	-	-	☑	-	-	✓	-
Pinkie proximal	-	-	-	-	-	-	-	-	-	-
Pinkie intermediate	-	-	-	-	-	-	-	-	-	-

✓indicates the features that were present in the reference table.
☑ indicates the features that were *not* present in the reference table.
✖ indicates the features that are *no more* present with respect to the reference table.

Table 5.5: Reduced features, computed giving as input the "dynamic" dataset

	a_x	a_y	a_z	ω_x	ω_y	ω_z	θ_x	θ_y	θ_z	θ_w
Base	✓	✓	✓	-	-	-	✓	✓	☑	✓
Thumb metacarpal	✖	✓	✓	-	-	-	☑	☑	-	-
Thumb intermediate	✓	✖	✓	-	-	-	-	-	-	-
Index proximal	✖	✓	✓	-	-	-	✓	✓	✓	✓
Index intermediate	✓	✓	✓	✓	-	-	✓	✓	✓	✓
Middle proximal	☑	✓	✓	-	-	-	-	-	☑	☑
Middle intermediate	✓	✓	✓	✖	-	-	-	-	-	-
Ring proximal	✖	-	✖	-	-	-	-	-	-	-
Ring intermediate	-	✖	✓	-	-	-	-	-	✓	☑
Pinkie proximal	-	-	-	-	-	-	-	-	-	-
Pinkie intermediate	-	-	-	-	-	-	-	-	-	-

✓indicates the features that were present in the reference table.
☑ indicates the features that were *not* present in the reference table.
✖ indicates the features that are *no more* present with respect to the reference table.

Comparing the features introduced/removed during these analyses, we can notice that there are no cases where some IMUs are more important than others; this happens because the features added/removed are *isolated* [1].
Despite this, it is possible to observe that the features of the gestures in *C1* are devoid of many of the linear accelerations of the phalanges that were present in the reference case (-25%), as they were considered less informative. This aspect is consistent with the nature of these gestures, which are mainly characterized by wrist movements, without marked movements of the phalanges (as in *C2*).
On the other hand, the cluster in *C2* continues to have more or less the same number of linear acceleration features (-8%). Once again, this aspect is consistent with the nature of *C2* gestures, which are mainly characterized by movements within the phalanges. In addition, in this case, the importance of some phalanxes (e.g.: intermediate component of the index finger) is underlined by the presence of angular velocities.

[1] the term *isolated* is used to indicate features that have been introduced singularly, without taking into account other features they are related to. As an example, in Table 6.1 the RFE algorithm introduced the y-component of the quaternion orientation of the middle (proximal phalanx). However, all the other components, i.e., x-y-w, are not part of the feature set, and thus this makes the new feature *isolated*

Chapter 6

Gesture classification

In this chapter, we address the topic of gesture recognition, following an approach consistent with the one that is already standard in the literature (24). To this end, we will address the problem of gesture recognition in two distinct phases. Initially, we will pre-process the dataset, to make it more suitable for the next steps. Then, we will: define an LSTM-based model, a consolidated solution available in the literature (10); establish the best hyperparameters through cross-validation; train the model with the dataset obtained *before* and *after* the *priming*, considering both *all* the features (110) and a *subset* of them (30); study the obtained results and establish which model best fits. Note that the following study is addressed *offline* and assuming that every gesture provided to the neural network is one of the gestures in the dictionary (*closed-world assumption*).

6.1 Data pre-processing

As pointed out in the literature, the first step in the gesture recognition pipeline often consists in carrying out a pre-processing of the dataset. In the context of this study, this involves three distinct phases. At first, the dataset is segmented to extract only those information relevant for the training of the probabilistic model. Next, it is normalized and finally padded to match the length of the gestures, which are generally different.

6.1.1 Automatic segmentation

When collecting the dataset, the participant had much more time available than the average time needed to perform gestures. This resulted in all inertial data presenting a section devoid of any movement. Therefore, it is necessary to apply segmentation techniques, which allows to extract the relevant portions of data

from the raw dataset.
We developed an automatic segmentation algorithm to extract the relevant portions of data from the input sequences. Given a gesture, the algorithm computes, for each IMU, the norm of the acceleration components and identifies the start and the end points of a gesture by applying a threshold, here set to $1.2 \cdot 9.81\ m/s^2 = 11.76\ m/s^2$, estimated when the hand was held steady in the initial rest position. This assumes, obviously enough, that the relevant motion in each trial corresponds to the gesture execution.

6.1.2 Normalization

Typically, normalization is one of the preliminary steps for machine learning; it allows to rescale on a common scale the values assumed by the features, without distorting them. Usually, in these cases, the variations of the features differ from each other by several units of magnitude. In this context, there is a significant difference between the range of variation of the linear accelerations, whose values vary in the order of thousands of units, and that of the orientation features, whose values range between $[0, 1]$.
Therefore, we normalized all linear accelerations and angular velocities in the dataset. To do so, we considered the minimum and maximum values measurable by the IMUs in the glove, which correspond to: $|4 \cdot \vec{g}|$ for the triaxial linear accelerations; $|2000|$ rad/sec for the triaxial angular velocities. Note that the unit quaternion was not part of the normalization, since it is already within the target range.

6.1.3 Data padding

As explained in the previous chapters, gestures usually have a different number of samples. This occurs because each participant carries out gestures in their way, taking a slightly different amount of time. Since the following neural network needs gestures with an equal number of samples, we padded the data, adding zeros to the end of each trial, obtaining sequences with the same number of samples, here set to the maximum number of samples found in the dataset, i.e., 160.

6.2 Model architecture

We selected a state-of-the-art network for these kinds of studies, i.e., a Long-Short Term Memory (LSTM) Recurrent Neural Network (RNN), for its capability to learn time-dependent information (66). We selected the best hyperparameters considering standard cross-validation approaches, later discussed. Overall, the model architecture is defined by:

- an initial *masking layer*;
- a *bi-dimensional LSTM* layer, with 264 neurons, "tanh" activation function, "sigmoid" recurrent activation functions and $L2$ kernel regularizer with penalty equal to 0.001
- a *dropout layer*
- a *dense layer* with 184 neurons and a "relu" activation function
- a *dense layer* with 12 neurons and a "softmax" activation function

The classification model is implemented using Keras and TensorFlow. As explained in the previous chapters, each Italian gesture may have different lengths because, generally, participants carry out gestures in their own way, taking a slightly different amount of time. For this reason, we padded the data to obtain sequences of 160 samples and used an initial masking layer; the latter allows to exclude, from the following computations, the values added during the padding. The model is based on a bidirectional LSTM, which is an extension of traditional LSTM layers that involves duplicating the first recurrent layer in the network. Having two (duplicated) layers allows to give as input both data as it is and its reversed copy. This mechanism is useful when, in cases like those of this study, data sequences are available all at once, and thus it is possible to exploit both *past* (input) and *future* (reversed input) information.
The *L2* kernel regularizer, also known as *weight decay*, is a regularization technique that reduces overfitting, by intuitively allowing the model to prefer learning small weights. This is achieved adding a term to the cost function, which is scaled by the *regularization penalty*. As described above, the LSTM layer contains a *L2* regularizer with penalty equal to 0.001. Moreover, to prevent overfitting, we recur to Dropout regularization. It is a regularization technique that, unlike kernel and bias regularizers, modifies the network structure. The latter is randomly deprived of some neurons, selected with a frequency here set to 0.3.
Following the LSTM layer, two dense layers are employed. The first one receives the last LSTM hidden state, whose dimensionality coincides with the number of neurons, i.e., 312; the second one follows the dimensionality of the previous dense

layer, i.e., 156 neurons, and brings it back to the original number of features (12). Note the use of the final "softmax" activation function, which allows to obtain the probabilities that the input gesture belongs to each of the 12 classes.
As initially mentioned, the choice of model architecture was made taking into account similar studies (31) - (10).

6.3 Approach

Hereafter, the model is trained *offline*, having all the gestures available in advance. Hence, no *online* analysis takes place. The neural network takes as input gestures characterized by time-series of 160 timestamps and 110 features. Since the dataset contains multiple examples, the model input is a three-dimensional matrix, where:

- the first dimension (rows) depicts the *timestamps*
- the second dimension (columns) corresponds to the *features*
- the third dimension (depth) represents the *samples*, where each samples is a complete gesture execution (160 timestamps, 110 features)

Note that the maximum LSTM ability to learn time dependent information extends over the duration of a gesture, which corresponds to 160 samples, i.e., 5.7 seconds.

In the following analyses, the dataset is split into three different parts, defined automatically and randomly at run time. A portion of the dataset (60% approximately) is given to the *training set*, which is used during the *learning*; a small section (15%), is reserved to the *validation set*, which allows to check the performances during the training and, eventually, to interrupt it in case they are satisfactory enough (*early stopping*).
A small portion of the dataset (25%) is assigned to the *test set*, which is fed to the trained model for evaluating its performance.
How training, test and validation sets are built is fundamental to carry out a statistical analysis of the model performances. If data were split in a purely random way, there would be no control over the number of examples per class in the training and in the test sets. This could lead to unbalanced sets, where some classes occur more often than others. This is also valid when reasoning in terms of participants, not gesture classes. Because data from wearable sensors are highly variable and person-dependent, it is important to ensure that the contribution of participants is balanced.
With these principles in mind, the following analyses are carried out considering *k-fold* cross validation (kFCV). In general, k-fold divides all samples into different groups, i.e., folds, which should approximately be of equal size. Given k folds, the

model is trained considering a training set of $k - n$ folds, and evaluated on the remaining ones. The overall performance is computed as the mean of all the fold performances. More specifically, we consider: *Stratified* kFCV, which provides stratified folds, where each fold contains approximately the same percentage of samples of every target class; *GroupKFold*, a variation of k-fold which ensures that the same group of participants (25% of the total) is not represented in both testing and training sets.
We consider stratified kFCV because, in the case of offline studies, it effectively allows the validation of the system. However, we also consider group-kFCV approaches because, as pointed out in the literature, data generated by wearable sensors are highly participant-dependent. Therefore, to get an idea of what performance the system would have if used online, we need to consider this type of cross-validation.

6.4 Hardware characteristics

The model is trained and evaluated on one single computer, whose characteristics are stated below:

Table 6.1: Hardware requirements

OS	Ubuntu 20.04.03 LTS
OS Type	64-bit
GNOME Version	3.36.8
Processor	Intel Xeon(R) CPU E5-2630 v2 @ 2.60 GHz × 12
Memory	16 GB
Graphics	NVIDIA Corporation GK110 GeForce GTX Titan, 6 GB GDDR5 memory

6.5 Models trained with original features

In this section, we analyse the performance of the model, considering all available information. In other words, the dataset used to train the model includes all 110 features. As in the previous sections, the Italian hand gesture dataset is divided into two sets, the one obtained before the *priming*, and that obtained after the latter. As a consequence, the analysis is carried out two times, one for each subset, considering stratified cross-validation. Table 6.2 summarizes the model performances, while Figure 6.1 shows one possible evolution of the validation

accuracy and loss during the training of the model. Both models show a fair generalization capability, as the accuracy is higher than 80%, with a standard deviation that does not vary much. This is a satisfactory result for two reasons: the higher number of classes, compared to those available in the literature (8; 9); the variability among gestures performed by people with different experiences does not compromise the generalization capabilities of the model. Table 6.2 shows that the accuracy of the *priming-based* model is higher than that of the *non-priming-based* model. This result is reasonable, since, in the analyses provided in the previous chapters, it was pointed out the high variability of this dataset. Figure 6.2 shows one of the confusion matrices evaluated during the stratified kFCV, giving as input the *priming* dataset. In the diagonal of the matrix, we can see the number of trials correctly classified, which is 92.0% of the total. Note that the maximum number of trials, for each class, is 31 on average. The matrix allows to visualize graphically the performance of the model: almost all classes are correctly categorized; nevertheless, the matrix shows that the two classes "What do you want" and "Fear" are the ones that are most confused by each other. The confusion matrix obtained feeding the model with the *no priming* data is not provided, as the results are analogous.

Table 6.2: Model performances - original dataset

	no priming	*priming*
Training accuracy	96.2±2.6%	98.4±2.3%
Test accuracy	85.0±2.5%	94.3±3.0%

The same analysis is carried out considering groupKfold cross-validation. In this case, as previously mentioned, data were divided in terms of participants (i.e., training set: 60%, validation set: 15%, test set: 25% of the dataset), so that the contribution of the ones considered in the training set did not contaminate the test set. In this case, the *priming* accuracy drops to 96.3±1.4% (training set) and 77.8 ± 5.1% (test set). For the *no priming* dataset, the training set accuracy is equal to 98.0 ± 0.4%, while the test set accuracy drops to 69.5 ± 5.6%. Once again, this highlights the high variability of the *no priming* dataset.
Overall, given the high participant-variability of wearable sensors, the previous accuracies are a satisfactory result, which could be improved by collecting more data.

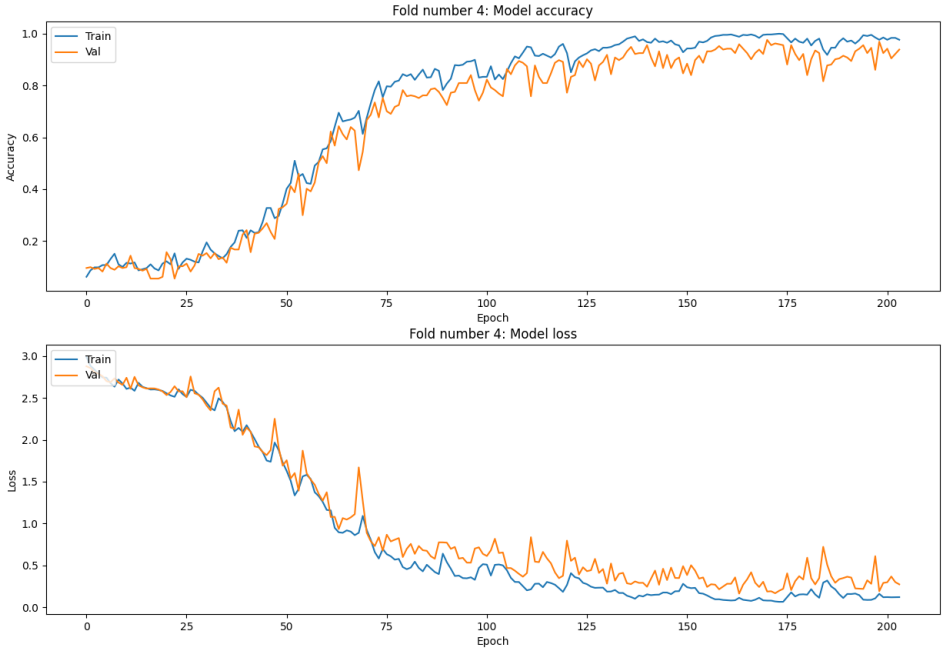

Figure 6.1: Validation accuracy and loss evolution - *priming* dataset

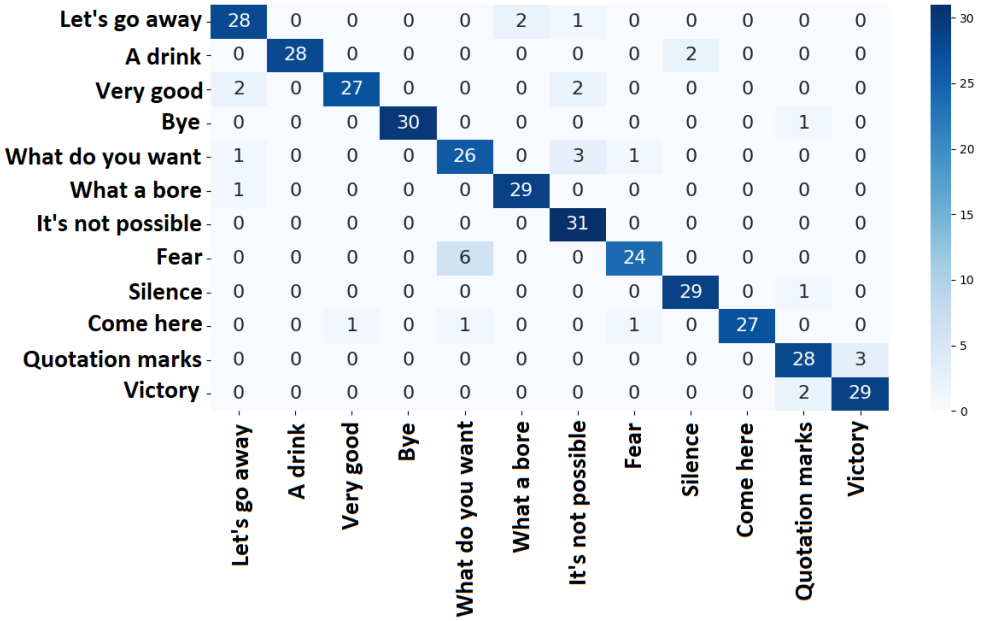

Figure 6.2: Confusion matrix computed on priming dataset, considering all features

6.6 Reduced features

In this section, we show the model performances when it is trained with a subset of features. For clarity, the subset of features includes the accelerations and angular velocities of the index, ring and base components (proximal and intermediate phalanges). We extracted these features from Table 5.2 (Chapter 5), considering all the components of the most informative fingers.
Table 6.3 summarizes the model performances obtained considering groupKfold cross-validation. As in the previous case, both average accuracies are sufficiently high. This is especially evident in the priming dataset. Furthermore, having low standard deviations allows us to conclude that the generalization capability of the model is satisfactory enough to allow its deployment. In Figure 6.3 we show one of the confusion matrices evaluated during the cross-validation, giving as input the *no priming* dataset. As the matrix suggests, almost all the classes are correctly classified.

Table 6.3: Model performances - reduced dataset

	no priming	*priming*
Training accuracy	95.9±1.6%	97.5±0.6%
Test accuracy	78.2±2.3%	85.3(±3.4)%

6.7 Model comparison

In this section, we evaluate the performance of models trained with all features, i.e., 110, and with a subset of them, i.e., 30. Hereafter, the models are respectively referred to as *original* model and *reduced* model.
The model architecture, i.e., the layers and the hyperparameters, is similar to the previous one. The only difference regards the smaller number of inputs, which influences the dimensionality of the LSTM hidden layer. Comparing the performances of the original model, evaluated considering a group kFCV, with those of the reduced model, we can observe that the accuracy of the models has slightly improved (+8.7% for the *no priming* data and +7.5% for the *priming* data). Besides, the reduced model is much less complex: the main reason is the lower number of parameters, which is a consequence of the lower number of features. More precisely, the number of parameters is 17% lower than in the original model, which was characterized by about 500000 parameters. This is important because, having fewer trainable parameters, reduces the time spent training the model. In fact, it was reduced by 20% compared to the original

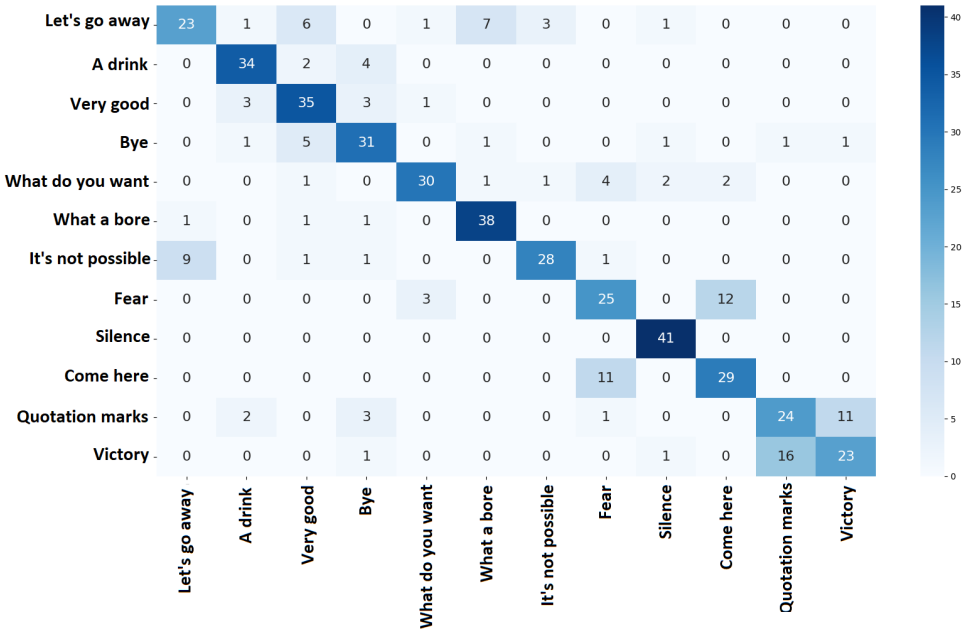

Figure 6.3: Confusion matrix computed on *no priming* dataset considering the subset of features

model (2.00 ± 0.15 minutes against 2.50 ± 0.26 minutes). As a marginal note, reducing the number of parameters also resulted in a significant reduction (16.5% less) in the amount of memory required to store the model, i.e., around 20 MB. From this analysis, it appears that the reduced model result in a good trade-off between computational complexity and accuracy of the system. Hence, the model can be trained with the reduced feature set, as it is a good trade-off between model-dataset complexity and system accuracy.

Chapter 7

Conclusions

Nowadays, the technical and technological evolution allows researchers to investigate deeply the Human-Robot Interaction possibilities. To obtain an efficient HRI, it is of most importance to understand the mechanisms that handle the human interaction. Once such mechanisms are well-defined, the focus can be extended to interactions with robots.
As we saw in the literature review, human interaction is complex due to the multitude of mechanisms and ways it is carried out. An individual can communicate with words, gestures, body part movements, eye gaze and even silence. As a consequence, having the robot to model all these aspects is a big challenge.
As seen in chapter 2, there are numerous studies addressing gesture recognition, a key element in non-verbal communication. However, they focus on unnatural and synthetic gestures, with low social relevance. Therefore, we decided to analyse from scratch the Italian gestures, which are distinguishable for their social impact and for the naturalness and spontaneity with which they have been evolving over the centuries (67).

With that being said, this Thesis aimed to provide a pioneering study of Italian gestures, including: glove-based data collection organized as human-robot interactions, which reflect the social nature of gestures; study of similarity behaviors between the classes examined; gesture recognition via consolidated data-driven approaches.
To complete the data collection, we set up experiments in which iCub was responsible for the acquisitions. During the experiments, iCub led participants to perform Italian gestures, by verbally providing imaginary contexts designed to bring them into concrete situations, where they could use the gestures. Using a robot allowed to precisely provide the same stimuli to all participants, an important aspect due to the social nature of this study.
Thirty-one people participated in the experiments; each of them reproduced 8 times each of the 12 classes of gestures. Hence, we collected 2884 gestures in

total. More specifically, gestures were collected in two distinct phases, each of 4 repetitions for each class. In the first phase, participants reproduced gestures based on their experience; in the second phase, a video showing the execution of each gesture class was shown before reproducing the gestures (*priming*). This division was preserved throughout the Study, indicating the data collected in the two phases as *no priming* and *priming* respectively.
Once the experiments were finished, the focus shifted to analysing the data collected through the custom made inertial glove. Initially, we removed the failed trials and applied normalization and automatic segmentation to all the gestures in the dataset. At this stage, it was possible to address the complexity of the dataset, characterised by the temporal evolution of 110 unique time-series. To reduce such complexity, we extracted atemporal features from each time-series and applied Pairwise Controlled Manifold Approximation (PaCMAP), a data-driven approach that allowed to describe each example through two unique features.

This allowed us to carry out the gesture analysis, by which we discovered that *priming* gestures have an intra-class similarity that is much higher than that found in the *no priming* data. Moreover, we pointed out the occurrence of two main clusters, one whose gestures are characterized by phalanx movements, and another with more general hand movements. Lastly, we concluded on the marked similarity between the classes "Quotation marks" and "Victory".

During this study, we examined the possibility of reducing, a priori and without necessarily recurring to data-driven techniques, the number of features, selecting the most informative ones, to reduce the complexity of this pioneering dataset as much as possible. Performing this analysis, implemented through an approach known as Recursive Feature Elimination (RFE), we observed that the most informative features are only 30 out of 110, i.e. 27% of the total.

The last contribution of this study was to address the recognition of natural and spontaneous gestures. More specifically, the purpose of this analysis was to understand if it was possible to recognize gestures *offline* and under a *close-world assumption*[1], evaluating the model's performance and establishing a trade-off between computational complexity and system accuracy, rather than its online implementation. Therefore, we used a standard solution shown in the literature, based on a Long-Short Term Memory Recurrent Neural Network, for its ability to learn time-dependent information.
To evaluate the model performances, we applied standard cross-validation approaches, considering a training, validation, and test set respectively equal to 60%, 15%, and 25% of the dataset. We repeated the training of the model, feeding it with the *no priming* and then *priming* datasets. At first, we considered all

[1]As explained in the previous chapter, with the *closed-world assumption*, we assume that, during training, every input gesture is necessarily one of the gestures in the dictionary.

the features, but then only the most informative ones.
By assessing the performance in the test set, it emerged that, in general, the *priming* model works better than the one trained with the *no priming* data. This is due to the high variability of the latter dataset, where each participant reproduced gestures based on their own experience. This result highlighted the importance of the *priming* phase, which allowed us to obtain less variable inertial data.
Furthermore, comparing the model trained with fewer features to the one trained with all features, it was found that performances slightly increased (around 7%). Moreover, the *reduced model* was less computationally complex, with significantly lower time required for the training (20% less). Overall, the model trained with a reduced number of features turned out to be a satisfactory trade-off between computational complexity and accuracy of the system, and was therefore selected as the preferred solution.

7.1 Limitations and Future work

A drawback of the proposed gesture recognition model concerns the *closed-world* assumption: every input gesture is necessarily one of those in the dictionary with which the model was trained. If this is not the case, the model will try to classify the gesture as such and will make a mistake. However, in the context of this Thesis, the aim was not to implement from scratch a model capable of recognizing gestures online, but rather to undertake a study of natural and socially useful gestures. Hence, it was proven that not all IMUs are equally important in the gesture recognition, and we extracted only those most informative, then used to train the final model.
The natural progression of this work will involve the online deployment of the model, which would classify gestures on the fly. Naturally, we could use the model trained with fewer features. However, to proceed with online gesture recognition, the closed-world assumption should be relaxed, implementing, for instance, an approach similar to the one carried out in (10), where an indirect detection module is put after the classification module. At this point, the interest could be focused back to the human-robot interaction, developing experiments where iCub recognizes participant's gestures on the fly.
Since, during the dataset collection, we also recorded the scene with a camera at a frequency of 30 Hz, we could investigate gesture classification from a vision point of view. Furthermore, we could compare the performance of the vision-model with the that of the inertial-model, to evaluate the preferred solution.

One limitation that emerged during the data collection was related to the inertial glove. More specifically, the limitation concerns the flexible connections,

required by IMUs to communicate with the microcontroller. These connections, although stable, were not sufficiently robust: with very pronounced hand movements, they could be compromised, resulting in a reduction of the frequency of measured data. This aspect will be addressed in the next version of the inertial glove, where the flexible connections will be completely excluded from the hardware architecture; instead, we will opt for communication based on Bluetooth technology.

References

[1] M. Candidi, A. Curioni, F. Donnarumma, L. M. Sacheli, and G. Pezzulo, "Interactional leader–follower sensorimotor communication strategies during repetitive joint actions," Journal of the Royal Society Interface, vol. 12, no. 110, p. 20150644, 2015. vi, 7, 8

[2] A. Sciutti, L. Patane, F. Nori, and G. Sandini, "Understanding object weight from human and humanoid lifting actions," IEEE Transactions on Autonomous Mental Development, vol. 6, no. 2, pp. 80–92, 2014. vi, 8, 9

[3] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in 2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 301–308, IEEE, 2013. vi, 10

[4] M. Mori, K. F. MacDorman, and N. Kageki, "The uncanny valley [from the field]," IEEE Robotics & Automation Magazine, vol. 19, no. 2, pp. 98–100, 2012. vi, 10, 11

[5] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," Computer Vision and Image Understanding, vol. 108, no. 1-2, pp. 52–73, 2007. vi, 15, 16

[6] L. C. Ebert, G. Hatch, M. J. Thali, and S. Ross, "Invisible touch—control of a dicom viewer with finger gestures using the kinect depth camera," Journal of Forensic Radiology and Imaging, vol. 1, no. 1, pp. 10–14, 2013. vi, 17, 21

[7] L. Lastrico, A. Carfì, V. Alessia, A. Sciutti, F. Mastrogiovanni, and F. Rea, "Careful with that! observation of human movements to estimate objects properties," 13th International Workshop on Human Friendly Robotics, 2020. vi, 10, 12, 23, 24

[8] R. Xie and J. Cao, "Accelerometer-based hand gesture recognition by neural network and similarity matching," IEEE Sensors Journal, vol. 16, no. 11, pp. 4537–4545, 2016. vi, 3, 19, 22, 24, 25, 26, 27, 28, 72

[9] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan, "uwave: Accelerometer-based personalized gesture recognition and its applications," Pervasive and Mobile Computing, vol. 5, no. 6, pp. 657–675, 2009. vi, viii, 3, 21, 25, 26, 27, 28, 72

[10] A. Carfi, C. Motolese, B. Bruno, and F. Mastrogiovanni, "Online human gesture recognition using recurrent neural networks and wearable sensors," in 2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), pp. 188–195, IEEE, 2018. vi, 4, 20, 27, 28, 67, 70, 78

[11] H. Liu and L. Wang, "Gesture recognition for human-robot collaboration: A review," International Journal of Industrial Ergonomics, vol. 68, pp. 355–367, 2018. viii, 17, 18

[12] B. Munari, Supplemento al dizionario italiano. Muggiani, 1963. viii, 32, 34

[13] G. B. Leoni, "Italian gestures," 2021. viii, 34

[14] I. Poggi, "Symbolic gestures: The case of the italian gestionary," Gesture, vol. 2, no. 1, pp. 71–98, 2002. viii, 13, 34

[15] I. Poggi, "Gesti," 2010. viii, 32, 34

[16] "icub joints." https://icub-tech-iit.github.io/documentation/icub_kinematics/icub-joints/icub-joints/. Accessed: 2021-09-30. viii, 46

[17] A. Bonarini, "Communication in human-robot interaction," Current Robotics Reports, pp. 1–7, 2020. 2

[18] A. Mehrabian, "Nonverbal communication. piscataway," 1972. 2

[19] K. S. Lohan, H. Lehmann, C. Dondrup, F. Broz, and H. Kose, "Enriching the human-robot interaction loop with natural, semantic and symbolic gestures," Humanoid Robotics: A Reference, pp. 2199–2219, 2019. 2

[20] G. Sandini, A. Sciutti, F. Rea, A. Goswami, and P. Vadakkepat, "Movement-based communication for humanoid-human interaction," Humanoid Robotics: A Reference, pp. 2169–2197, 2019. 2

[21] A. Mehrabian, Nonverbal communication. Routledge, 2017. 2

[22] A. Moin, A. Zhou, A. Rahimi, A. Menon, S. Benatti, G. Alexandrov, S. Tamakloe, J. Ting, N. Yamamoto, Y. Khan, et al., "A wearable biosensing system with in-sensor adaptive machine learning for hand gesture recognition," Nature Electronics, vol. 4, no. 1, pp. 54–63, 2021. 3

[23] A. S. Kundu, O. Mazumder, P. K. Lenka, and S. Bhaumik, "Hand gesture recognition based omnidirectional wheelchair control using imu and emg sensors," Journal of Intelligent & Robotic Systems, vol. 91, no. 3, pp. 529–541, 2018. 3

[24] E. V. Añazco, S. J. Han, K. Kim, P. R. Lopez, T.-S. Kim, and S. Lee, "Hand gesture recognition using single patchable six-axis inertial measurement unit via recurrent neural networks," Sensors, vol. 21, no. 4, p. 1404, 2021. 3, 67

[25] O. Palinko, F. Rea, G. Sandini, and A. Sciutti, "Robot reading human gaze: Why eye tracking is better than head tracking for human-robot collaboration," in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5048–5054, IEEE, 2016. 6

[26] G. Pezzulo, F. Donnarumma, and H. Dindo, "Human sensorimotor communication: A theory of signaling in online social interactions," PloS one, vol. 8, no. 11, p. e79876, 2013. 7

[27] L. Schmitz, C. Vesper, N. Sebanz, and G. Knoblich, "When height carries weight: communicating hidden object properties for joint action," Cognitive science, vol. 42, no. 6, pp. 2021–2059, 2018. 7

[28] G. Metta, L. Natale, F. Nori, G. Sandini, D. Vernon, L. Fadiga, C. Von Hofsten, K. Rosander, M. Lopes, J. Santos-Victor, et al., "The icub humanoid robot: An open-systems platform for research in cognitive development," Neural networks, vol. 23, no. 8-9, pp. 1125–1134, 2010. 8

[29] A. Vignolo, N. Noceti, F. Rea, A. Sciutti, F. Odone, and G. Sandini, "Detecting biological motion for human–robot interaction: A link between perception and action," Frontiers in Robotics and AI, vol. 4, p. 14, 2017. 9

[30] S. S. Srinivasa, D. Berenson, M. Cakmak, A. Collet, M. R. Dogar, A. D. Dragan, R. A. Knepper, T. Niemueller, K. Strabala, M. V. Weghe, et al., "Herb 2.0: Lessons learned from developing a mobile manipulator for the home," Proceedings of the IEEE, vol. 100, no. 8, pp. 2410–2428, 2012. 10

[31] C. Alessandro and M. Fulvio, "Gesture-based human-machine interaction: Taxonomy, problem definition, and analysis," IEEE transactions on cybernetics, 2020. 11, 13, 25, 28, 70

[32] T. Vuletic, A. Duffy, L. Hay, C. McTeague, G. Campbell, and M. Grealy, "Systematic literature review of hand gestures used in human computer interaction interfaces," International Journal of Human-Computer Studies, vol. 129, pp. 74–94, 2019. 12

[33] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," IEEE Transactions on pattern analysis and machine intelligence, vol. 19, no. 7, pp. 677–695, 1997. 14

[34] J. Lee and T. L. Kunii, "Model-based analysis of hand posture," IEEE Computer Graphics and applications, vol. 15, no. 5, pp. 77–86, 1995. 14, 15

[35] J. C. Nunez, R. Cabido, J. J. Pantrigo, A. S. Montemayor, and J. F. Velez, "Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition," Pattern Recognition, vol. 76, pp. 80–94, 2018. 15

[36] T. L. Baldi, S. Scheggi, L. Meli, M. Mohammadi, and D. Prattichizzo, "Gesto: A glove for enhanced sensing and touching based on inertial and magnetic sensors for hand tracking and cutaneous feedback," IEEE Transactions on Human-Machine Systems, vol. 47, no. 6, pp. 1066–1076, 2017. 15, 17, 20

[37] L. Yun, Z. Lifeng, and Z. Shujun, "A hand gesture recognition method based on multi-feature fusion and template matching," Procedia Engineering, vol. 29, pp. 1678–1684, 2012. 18

[38] M. Müller, "Dynamic time warping," Information retrieval for music and motion, pp. 69–84, 2007. 18

[39] T. Fletcher, "Support vector machines explained," Tutorial paper., Mar, p. 28, 2009. 18

[40] D. Svozil, V. Kvasnicka, and J. Pospichal, "Introduction to multi-layer feed-forward neural networks," Chemometrics and intelligent laboratory systems, vol. 39, no. 1, pp. 43–62, 1997. 19

[41] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," nature, vol. 521, no. 7553, pp. 436–444, 2015. 19

[42] H.-K. Lee and J.-H. Kim, "An hmm-based threshold model approach for gesture recognition," IEEE Transactions on pattern analysis and machine intelligence, vol. 21, no. 10, pp. 961–973, 1999. 19

[43] S. Shin and W. Sung, "Dynamic hand gesture recognition for wearable devices with low complexity recurrent neural networks," in 2016 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 2274–2277, IEEE, 2016. 19

[44] M. Joselli and E. Clua, "grmobile: A framework for touch and accelerometer gesture recognition for mobile games," in 2009 VIII Brazilian Symposium on Games and Digital Entertainment, pp. 141–150, IEEE, 2009. 20, 23, 24

[45] Z. Zhang, "Microsoft kinect sensor and its effect," IEEE multimedia, vol. 19, no. 2, pp. 4–10, 2012. 20

[46] A. Ramey, V. González-Pacheco, and M. A. Salichs, "Integration of a low-cost rgb-d sensor in a social robot for gesture recognition," in 2011 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 229–230, IEEE, 2011. 21

[47] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust part-based hand gesture recognition using kinect sensor," IEEE transactions on multimedia, vol. 15, no. 5, pp. 1110–1120, 2013. 21

[48] G. Luzhnica, J. Simon, E. Lex, and V. Pammer, "A sliding window approach to natural hand gesture recognition using a custom data glove," in 2016 IEEE Symposium on 3D User Interfaces (3DUI), pp. 81–90, IEEE, 2016. 21, 23

[49] W. T. Higgins, "A comparison of complementary and kalman filtering," IEEE Transactions on Aerospace and Electronic Systems, no. 3, pp. 321–325, 1975. 21

[50] B. Bruno, F. Mastrogiovanni, A. Saffiotti, and A. Sgorbissa, "Using fuzzy logic to enhance classification of human motion primitives," in International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, pp. 596–605, Springer, 2014. 22

[51] G. Arce and M. McLoughlin, "Theoretical analysis of the max/median filter," IEEE transactions on acoustics, speech, and signal processing, vol. 35, no. 1, pp. 60–69, 1987. 22

[52] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan, "Vision-based hand-gesture applications," Communications of the ACM, vol. 54, no. 2, pp. 60–71, 2011. 22

[53] M. Hasanuzzaman, V. Ampornaramveth, T. Zhang, M. Bhuiyan, Y. Shirai, and H. Ueno, "Real-time vision-based gesture recognition for human robot interaction," in 2004 IEEE International Conference on Robotics and Biomimetics, pp. 413–418, IEEE, 2004. 22

[54] J.-S. Wang and F.-C. Chuang, "An accelerometer-based digital pen with a trajectory recognition algorithm for handwritten digit and gesture recognition," IEEE Transactions on Industrial Electronics, vol. 59, no. 7, pp. 2998–3007, 2011. 23

[55] Z. He, "Accelerometer based gesture recognition using fusion features and svm.," JSW, vol. 6, no. 6, pp. 1042–1049, 2011. 25

[56] W. T. Cochran, J. W. Cooley, D. L. Favin, H. D. Helms, R. A. Kaenel, W. W. Lang, G. C. Maling, D. E. Nelson, C. M. Rader, and P. D. Welch, "What is the fast fourier transform?," Proceedings of the IEEE, vol. 55, no. 10, pp. 1664–1674, 1967. 25

[57] M. Khan, S. I. Ahamed, M. Rahman, and J.-J. Yang, "Gesthaar: An accelerometer-based gesture recognition method and its application in nui driven pervasive healthcare," in 2012 IEEE International Conference on Emerging Signal Processing Applications, pp. 163–166, IEEE, 2012. 25, 26, 28

[58] R. S. Stanković and B. J. Falkowski, "The haar wavelet transform: its status and achievements," Computers & Electrical Engineering, vol. 29, no. 1, pp. 25–44, 2003. 25

[59] P. Neto, D. Pereira, J. N. Pires, and A. P. Moreira, "Real-time and continuous hand gesture spotting: An approach based on artificial neural networks," in 2013 IEEE International Conference on Robotics and Automation, pp. 178–183, IEEE, 2013. 25, 28

[60] G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori, "The icub humanoid robot: an open platform for research in embodied cognition," in Proceedings of the 8th workshop on performance metrics for intelligent systems, pp. 50–56, 2008. 42

[61] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, et al., "Ros: an open-source robot operating system," in ICRA workshop on open source software, vol. 3, p. 5, Kobe, Japan, 2009. 42, 43

[62] G. Metta, P. Fitzpatrick, and L. Natale, "Yarp: yet another robot platform," International Journal of Advanced Robotic Systems, vol. 3, no. 1, p. 8, 2006. 42

[63] Y. Wang, H. Huang, C. Rudin, and Y. Shaposhnik, "Understanding how dimension reduction tools work: an empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization," arXiv preprint arXiv:2012.04456, 2020. 54

[64] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," arXiv preprint arXiv:1802.03426, 2018. 54

[65] L. Talavera, "An evaluation of filter and wrapper methods for feature selection in categorical clustering," in International Symposium on Intelligent Data Analysis, pp. 440–451, Springer, 2005. 61

[66] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," arXiv preprint arXiv:1508.01991, 2015. 69

[67] A. De Jorio, La mimica degli antichi investigata nel gestire napoletano. Fibreno, 1832. 76

[68] T. Chaminade and G. Cheng, "Social cognitive neuroscience and humanoid robotics," Journal of Physiology-Paris, vol. 103, no. 3-5, pp. 286–295, 2009.

[69] J. A. DeVito, S. O'Rourke, and L. O'Neill, Human communication. Longman, 2000.

[70] "The development of sensitivity to causally relevant dynamic information," Child Development, vol. 55, no. 4, pp. 1614–1624, 1984.

[71] A. Sciutti, M. Mara, V. Tagliasco, and G. Sandini, "Humanizing human-robot interaction: On the importance of mutual understanding," IEEE Technology and Society Magazine, vol. 37, no. 1, pp. 22–29, 2018.

[72] S. Mitra and T. Acharya, "Gesture recognition: A survey," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 37, no. 3, pp. 311–324, 2007.

[73] F. Gustafsson, "Particle filter theory and practice with positioning applications," IEEE Aerospace and Electronic Systems Magazine, vol. 25, no. 7, pp. 53–82, 2010.

[74] L. Rabiner and B. Juang, "An introduction to hidden markov models," ieee assp magazine, vol. 3, no. 1, pp. 4–16, 1986.

[75] A. M. Khan, M. H. Siddiqi, and S.-W. Lee, "Exploratory data analysis of acceleration signals to select light-weight and accurate features for real-time activity recognition on smartphones," Sensors, vol. 13, no. 10, pp. 13099–13122, 2013.

[76] R. Xu, S. Zhou, and W. J. Li, "Mems accelerometer based nonspecific-user hand gesture recognition," IEEE sensors journal, vol. 12, no. 5, pp. 1166–1173, 2011.

[77] S. Waldherr, R. Romero, and S. Thrun, "A gesture based interface for human-robot interaction," Autonomous Robots, vol. 9, no. 2, pp. 151–173, 2000.

[78] F. Khoshnoud and C. W. de Silva, "Recent advances in mems sensor technology-mechanical applications," IEEE Instrumentation & Measurement Magazine, vol. 15, no. 2, pp. 14–24, 2012.

[79] A. De Jorio, La mimica degli antichi investigata nel gestire napoletano. Fibreno, 1832.

[80] E. Amid and M. K. Warmuth, "TriMap: Large-scale Dimensionality Reduction Using Triplets," ArXiv e-prints, 2019.