Hand Gesture Recognition System based in Computer Vision and Machine Learning

Paulo Trigueiros[1,2,4], Fernando Ribeiro[2,4] and Luís Paulo Reis[3,4,5]

[1]Insituto Politécnico do Porto, IPP, Porto, Portugal
[2]DEI/EEUM - Departamento de Electrónica Industrial, Escola de Engenharia, Universidade do Minho, Guimarães, Portugal
[3]DSI/EEUM – Departamento de Sistemas de Informação, Escola de Engenharia, Universidade do Minho, Guimarães, Portugal
[4]Centro Algoritmi, Universidade do Minho, Guimarães, Portugal
[5]LIACC – Laboratório de Inteligência Artificial e Ciência de Computadores, Portugal

[1]pjt@iscap.ipp.pt, [2]fernando@dei.uminho.pt, [3]lpreis@dsi.uminho.pt

Abstract: Hand gesture recognition is a natural way of human computer interaction and an area of very active research in computer vision and machine learning. This is an area with many different possible applications, giving users a simpler and more natural way to communicate with robots/systems interfaces, without the need for extra devices. So, the primary goal of gesture recognition research applied to Human-Computer Interaction (HCI) is to create systems, which can identify specific human gestures and use them to convey information or controlling devices. For that, vision-based hand gesture interfaces require fast and extremely robust hand detection, and gesture recognition in real time. This paper presents a solution, generic enough, with the help of machine learning algorithms, allowing its application in a wide range of human-computer interfaces, for real-time gesture recognition. Experiments carried out showed that the system was able to achieve an accuracy of 99.4% in terms of hand posture recognition and an average accuracy of 93.72% in terms of dynamic gesture recognition. To validate the proposed framework, two applications were implemented. The first one is a real-time system able to help a robotic soccer referee judge a game in real time. The prototype combines a vision-based hand gesture recognition system with a formal language definition, the *Referee CommLang*, into what is called the *Referee Command Language Interface System* (ReCLIS). The second one is a real-time system able to interpret the Portuguese Sign Language. Sign languages are not standard and universal and the grammars differ from country to country. Although the implemented prototype was only trained to recognize the vowels, it is easily extended to recognize the rest of the alphabet, being a solid foundation for the development of any vision-based sign language recognition user interface system.

Keywords: hand posture recognition, hand gesture recognition, computer vision, machine learning, remote robot control, human-computer interaction

1 INTRODUCTION

Hand gesture recognition for human computer interaction is an area of active research in computer vision and machine learning (Maung, 2009). One of the primary goals of gesture recognition research is to create systems, which can identify specific gestures and use them to convey information or to control a device. Though, gestures need to be modelled in the spatial and temporal domains, where a hand posture is the static structure of the hand and a gesture is the dynamic movement of the hand. Being hand-pose one of the most important communication tools in human's daily life, and with the continuous advances of image and video processing techniques, research on human-machine interaction through gesture recognition led to the use of such technology in a very broad range of possible applications (Mitra and Acharya, 2007, Bourennane and Fossati, 2010), of which some are here highlighted:

- **Virtual reality**: enable realistic manipulation of virtual objects using ones hands (Yoon et al., 2006, Buchmann et al., 2004), for 3D display interactions or 2D displays that simulate 3D interactions.
- **Robotics and Tele-presence**: gestures used to interact with robots and to control robots (Trigueiros et al., 2011) are similar to fully-immersed virtual reality interactions, however the worlds are often real, presenting the operator with video feed from cameras located on the robot. Here, for example, gestures can control a robots hand and arm movements to reach for and manipulate actual objects, as well as its movement through the world.
- **Desktop and Tablet PC Applications**: In desktop computing applications, gestures can provide an alternative interaction to mouse and keyboard (Vatavu et al., 2012, Wobbrock et al., 2007, Li, 2010, Kratz and Rohs, 2011). Many gestures for desktop computing tasks involve manipulating graphics, or annotating and editing documents using pen-based gestures.
- **Games**: track a player's hand or body position to control movement and orientation of interactive game objects such as cars, or use gestures to control the movement of avatars in a virtual world. Play Station 2 for example has introduced the Eye Toy (Kim, 2008), a camera that tracks hand movements for interactive games, and Microsoft introduced the Kinect (Chowdhury, 2012) that is able to track users full body to control games.
- **Sign Language**: this is an important case of communicative gestures. Since sign languages are highly structural, they are very suitable as test-beds for vision-based algorithms (Zafrulla et al., 2011, Ong and Ranganath, 2005, Holt et al., 2010, Tara et al., 2012).

There are areas where this trend is an asset, as for example in the application of these technologies on interfaces that can help people with physical disabilities, or areas where it is a complement to the normal way of communicating. Sign language, for example, is the most natural way of exchanging information among deaf people, although it has been observed that they have difficulties in interacting with normal people. Sign language consists of a vocabulary of signs in exactly the same way as spoken language consists of a vocabulary of words. Sign languages are not standard and universal and the grammars differ from country to country. The Portuguese Sign Language (PSL), for example, involves hand movements, body movements and facial expressions (Wikipedia, 2012). The purpose of Sign Language Recognition (SLR) systems is to provide an efficient and accurate way to convert sign language into text or voice has aids for the hearing impaired for example, or enabling very young children to interact with computers (recognizing sign language), among others. Since SLR implies conveying meaningful information through the use of hand gestures (Vijay et al., 2012), careful feature selection and extraction are very important aspects to consider

In terms of hand gesture recognition, there are basically two types of approaches: vision-based approaches and data glove methods. This paper focuses on creating a vision-based approach, to implement a system capable of performing posture and gesture recognition for real-time applications. Vision-based hand gesture recognition systems were the main focus of the work since they provide a simpler and more intuitive way of communication between a human and a computer. Using visual input in this context makes it possible to communicate remotely with computerized equipment, without the need for physical contact or any extra devices (Chaudhary et al., 2011, Trigueiros et al., 2012).

As Hasanuzzaman (Hasanuzzaman et al., 2004) argue, it is necessary to develop efficient and real time gesture recognition systems, in order to perform more human-like interfaces between humans and robots. Although it is difficult to implement a vision-based interface for generic usage, it is nevertheless possible to design this type of interface for a controlled environment (Murthy and Jadon, 2009, Huang and Pavlovic, 1995). Furthermore, computer vision based techniques have the advantage of being non-invasive and based on the way human beings perceive information from their surroundings (Trigueiros et al., 2013). However, to be able to implement such systems, there are a number of requirements that the system must satisfy, in order to be implemented in a successful way (Murthy and Jadon, 2009), which are:

- **Robustness**: the system should be user independent and robust enough to factors like visual noise, incomplete information due for example to occlusions, variations of illumination, etc.
- **Computational efficiency**: vision based interaction requires real-time systems, so the algorithms and learning techniques should be the most effective possible and computational cost effective.
- **Error tolerance**: mistakes on vision-based systems should be tolerated and accepted. If some mistake is made, the user should be able to repeat the command, instead of letting the system make wrong decisions.
- **Scalability**: the system must be easily adapted and configured so that it can serve a number of different applications. The core of vision based applications for human computer interaction should be the same, regardless of the application.

Also, we need to have systems that allow training gestures and learn models capable of being used in real-time interaction systems. These systems should be easily configurable in terms of the number and type of gestures that they can train, to ensure the necessary flexibility and scalability.

The rest of this paper is as follows. First we present the Vision-based Hand Gesture Recognition System Architecture in Sect. 2, where the modules that constitute it are described. In this section, the problem of hand detection and tracking are addressed, as well as the problem of hand segmentation. Also, hand posture classification and dynamic gesture classification implementations are described. In Sect. 3, the Referee Command Language Interface System (ReCLIS), built to validate the proposed framework and able to help a robotic soccer referee judge a game in real time is described. This section also discusses the problem of modelling the command semantics for command classification. Sect. 4 presents the Sign Language Recognition prototype architecture and discusses its implementation. The prototype can be used to supplement the normal form of communication for people with hearing impairment. Conclusions and future work are drawn in section 5.

2 VISION-BASED HAND GESTURE RECOGNITION SYSTEM ARCHITECTURE

The design of any gesture recognition system essentially involves the following three aspects: (1) *data acquisition and pre-processing*; (2) *data representation or feature extraction* and (3) *classification or decision-making*. Taking this into account, a possible solution to be used in any human-computer interaction system is represented in the diagram of Figure 2.1. As it can be seen in the diagram, the system first detects and tracks the user hand, segments the hand from the video image and extracts the necessary hand features. The features thus obtained are used to identify the user gesture. If a static gesture is being identified, the obtained features are first normalized and the obtained instance vector is then used for classification. On the other hand, if a dynamic gesture is being classified, the obtained hand path is first labelled according to the predefined alphabet, giving a discrete vector of labels, which is then translated to the origin and finally used for classification. Each detected gesture is used as input into a module that builds the command sequence, i.e. accumulates each received gesture until a predefined sequence defined in the *Command Language* is found. The sequence thus obtained is classified into one of a set of predefined number of commands that can be transmitted to a Generic System Interface (GSI) for robot/system control.

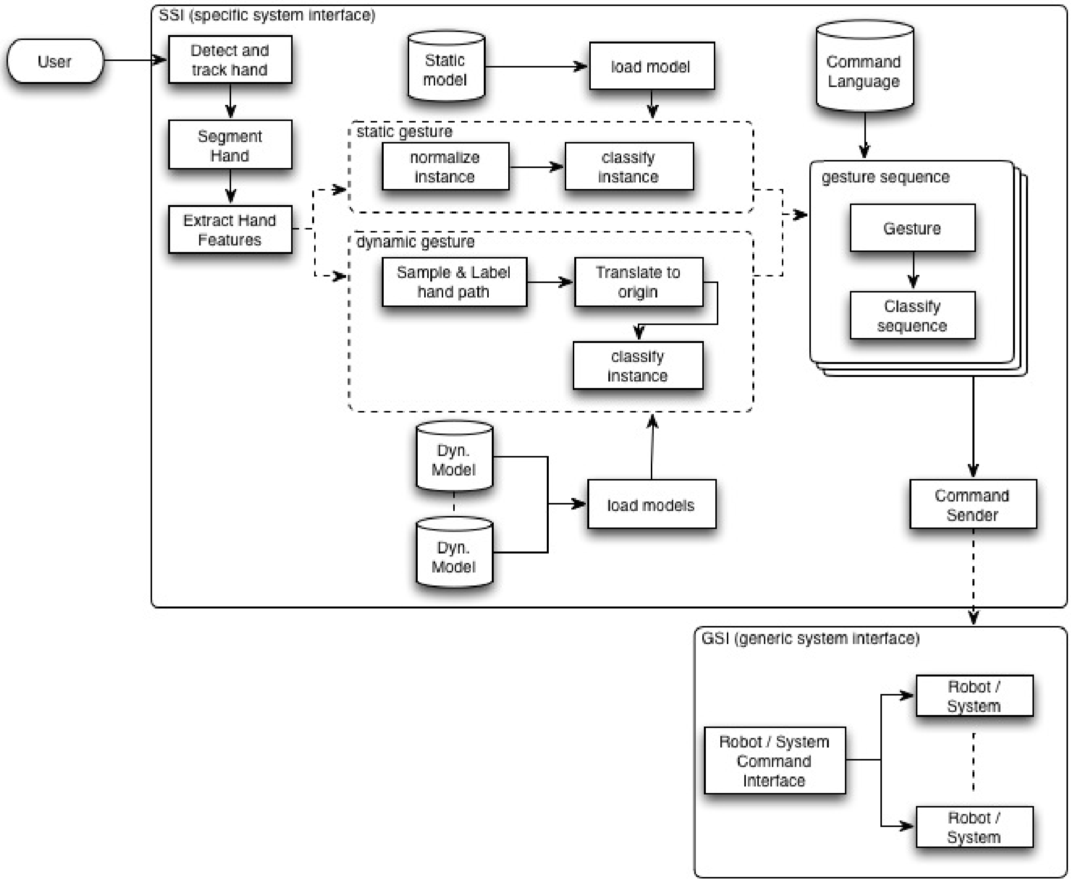


Figure 2.1. Vision-based hand gesture recognition system architecture

In the following sections we will describe the problems of hand posture classification and dynamic gesture classification.

2.1 HAND POSTURE CLASSIFICATION

For hand posture classification, hand segmentation and feature extraction is a crucial step in vision-based hand gesture recognition systems. The pre-processing stage prepares the input image and extracts features used later with classification algorithms (Trigueiros et al., 2013). The proposed system uses feature vectors composed of centroid distance values for hand posture classification. The centroid distance signature is a type of shape signature (Trigueiros et al., 2013) expressed by the distance of the hand contour boundary points, from the hand centroid (x_c, y_c) and is calculated in the following manner:

$$d(i) = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}, i = 0, \dots., N - 1 \tag{1}$$

This way, a one-dimensional function representing the hand shape is obtained. The number of equally spaced points N used in the implementation was 16. Due to the subtraction of centroid from the boundary coordinates, this operator is invariant to translation as shown by Rayi Yanu Tara (Tara et al., 2012) and a rotation of the hand results in a circularly shift version of the original image. All the features vectors are normalized, using the *z-normalization,* prior to training, by subtracting their mean and dividing by their standard deviation (Alpaydin, 2004, Montgomery and Runger, 1994) as follows,

$$Z = \left(a_{ij} - \bar{a}\right)/\sigma \tag{2}$$

where $\bar{a}$ is the mean of the instance *i,* and σ is the respective standard deviation, achieving this way scale invariance as desired. The vectors thus obtained have zero mean and a standard deviation of 1. The resulting feature vectors are used to train a multi-class Support Vector Machine (SVM) that is used to learn the set of hand postures shown in Figure 2.2, and used in the Referee Command Language Interface System (ReCLIS) and the hand postures shown in Figure 2.3 used with the Sign Language Recognition System. The SVM is a pattern recognition technique in the area of supervised machine learning, which works very well with high-dimensional data. SVM's select a small number of boundary feature vectors, *support vectors,* from each class and builds a linear discriminant function that separates them as widely as possible (Figure 2.4) - *maximum-margin hyperplane* (Witten et al., 2011). Maximum-margin hyperplanes have the advantage of being relatively stable, i.e., they only move if training instances that are support vectors are added or deleted. SVM's are non-probabilistic classifiers that predict for each given input the corresponding class. When more than two classes are present, there are several approaches that evolve around the 2-class case (Theodoridis and Koutroumbas, 2010). The one used in the system is the one-against-all, where *c* classifiers have to be designed. Each one of them is designed to separate one class from the rest.

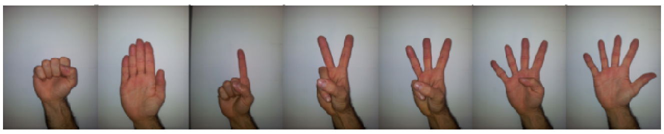


Figure 2.2. The defined and trained hand postures.

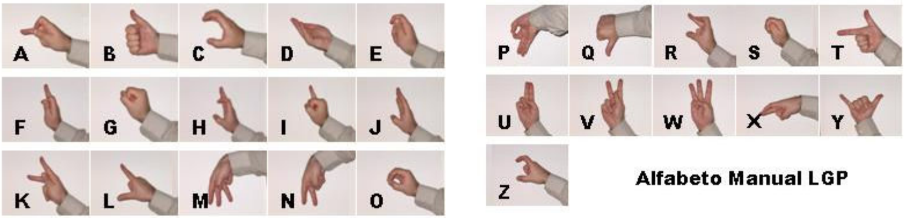


Figure 2.3. Manual alphabet for the Portuguese Language.

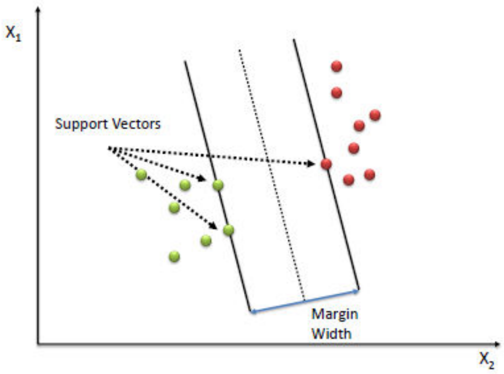
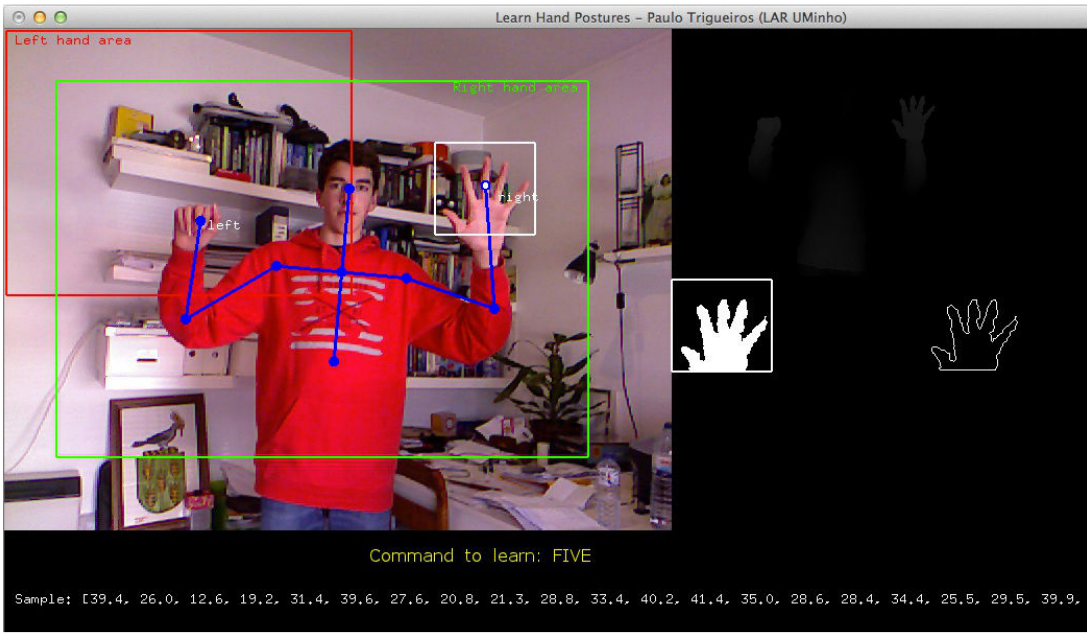


Figure 2.4. SVM: support vectors representation with maximum-margin hyperplane (Sayad, 2010)

2.1.1 Model training

For feature extraction, model learning and testing, a C++ application was built with openFrameworks (Lieberman et al., 2004), OpenCV (Bradski and Kaehler, 2008), OpenNI (OpenNI, 2013) and the Dlib machine-learning library (King, 2009). OpenCV was used for some of the vision-based operations like hand segmentation and contour extraction, and OpenNI was responsible for the RGB and depth image acquisition. Figure 2.5 shows the main user interface for the application, with a sample vector (feature vector) for the posture being learned displayed below the RGB image.



```
Command to learn: FIVE

Sample: [39.4, 26.0, 12.6, 19.2, 31.4, 39.6, 27.6, 20.8, 21.3, 28.8, 33.4, 40.2, 41.4, 35.0, 28.6, 28.4, 34.4, 25.5, 29.5, 39.9,
```

Figure 2.5. Static gesture feature extraction and model learning user interface.

Two centroid distance datasets were built: the first one for the first seven hand postures defined, with 7848 records and the second one for the Portuguese Sign Language vowels with a total of 2170 records, obtained from four users. The features thus obtained were analysed with the help of RapidMiner (Miner) in order to find the best kernel in terms of SVM classification for the datasets under study. The best kernel obtained with a parameter optimization process was the *linear kernel* with a *cost parameter* ***C*** equal to one. With these values, the final achieved accuracy was 99.4%.

In order to analyse how classification errors were distributed among classes, a confusion matrix for the two hand posture datasets was computed with the final results shown in Table 2.1 and Table 2.2.

Table 2.1. Confusion matrix for the seven hand postures trained.

		Actual class						
		1	**2**	**3**	**4**	**5**	**6**	**7**
Predicted class	**1**	**602**	0	0	0	0	0	0
	2	2	**712**	0	0	1	0	1
	3	0	1	**578**	1	0	0	0
	4	0	0	12	**715**	3	0	0
	5	0	1	1	13	**542**	1	3
	6	1	2	0	1	5	**701**	12
	7	0	0	0	2	0	1	**751**

Table 2.2. Confusion matrix for the Portuguese Sign Language vowels

		Actual class				
		1	**2**	**3**	**4**	**5**
Predicted class	**1**	**455**	0	0	2	0
	2	0	**394**	1	1	0
	3	0	0	**401**	1	0
	4	4	2	0	**382**	0
	5	0	0	1	0	**439**

2.2 DYNAMIC GESTURE CLASSIFICATION

Dynamic gestures are time-varying processes, which show statistical variations, making Hidden Markov Models (HMMs) a plausible choice for modelling the processes (Rabiner and Juang, 1986, Wu and Huang, 1999). A Markov Model is a typical model for a stochastic (i.e. random) sequence of a finite number of states (Fink, 2008). When the true states of the model $S = \{s_1, s_2, s_3, \dots, s_N\}$ are hidden in the sense that they cannot be directly observed, the Markov model is called a Hidden Markov Model (HMM). At each state an output symbol $O = \{o_1, o_2, o_3, \dots, o_N\}$ is emitted with some probability, and the state transitions to another with some probability, as shown in Figure 2.7. With discrete number of states and output symbols, this model is sometimes called a "*discrete HMM*" and the set of output symbols the *alphabet*. In summary, an HMM has the following elements:

- **N**: the number of states in the model $S = \{S_1, S_2, \dots, S_N\}$;
- **M**: the number of distinct symbols in the alphabet $V = \{v_1, v_2, \dots, v_M\}$;

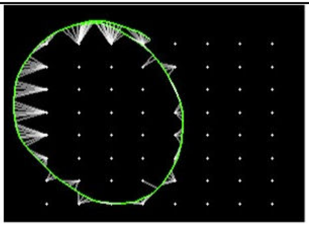
- State transition probabilities:

  $\boldsymbol{A} = [a_{ij}]\ where\ a_{ij} \equiv P(q_{t+1} = S_j | q_t = S_i)\ and\ q_t\ is\ the\ state\ at\ time\ t;$
- Observation probabilities:

  $\boldsymbol{B} = \{b_j(m)\}\ where\ b_j(m) \equiv P(O_t = v_m | q_t = S_j)\ and\ O\ is\ the\ observation\ sequence;$
- Initial state probabilities: $\boldsymbol{\Pi} = [\pi_i]\ where\ \pi_i \equiv P(q_1 = S_i);$

and is defined as $\lambda = (A, B, \Pi)$, where N and M are implicitly defined in the other parameters. The transition probabilities and the observation probabilities are learned during the training phase, with known data, which makes this is a supervised learning problem (Trigueiros et al., 2013).

In this sense, a human gesture can be understood as a HMM where the true states of the model are hidden in the sense that they cannot be directly observed. So, for the recognition of dynamic gestures a HMM model was trained for each possible gesture. HMMs have been widely used in a successfully way in speech recognition and hand writing recognition (Rabiner, 1989). In the implemented system, the 2D hand trajectory points are used and labelled according to the distance to the nearest centroid, based on Euclidean distance. The resulting vector is then translated to origin resulting in a discrete feature vector like the one shown in Figure 2.6.



[1 2 2 2 3 3 4 4 4 5 6 13 13 14 14 14 14 14 14 14 13 13
20 19 19 18 18 17 24 23 22 22 22 22 22 29 29 29 29
29 29 29]

Figure 2.6. Gesture path with respective feature vector.

The feature vectors thus obtained are used to train the different HMMs and learn the model parameters. In the recognition phase an output score for the sample gesture is calculated for each model, given the likelihood that the corresponding model generated the underlying gesture. The model with the highest output score represents the recognized gesture. The implemented system uses a Left-Right (LR) HMM (Camastra and Vinciarelli, 2008, Alpaydin, 2004), like the one shown in Figure 2.7. This kind of HMM has the states ordered in time so that as time increases, the state index increases or stays the same. This topology has been chosen, since it is perfectly suitable to model the kind of temporal gestures used.

2.2.1 Model training

For dynamic gesture model training, a C++ application for the acquisition of hand motion sequences (dynamic gestures) for each of the defined gestures, feature extraction and model training and testing was implemented. This application uses the same libraries as the previous application and an openFrameworks (Lieberman et al., 2004) add-on implementation of the HMM algorithm for classification and recognition of numeric sequences. This add-on is a C++ porting implementation of a MATLAB code from Kevin Murphy (Murphy, 1998).

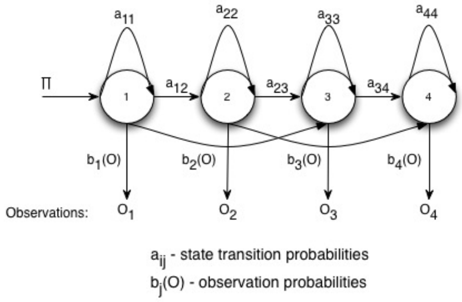
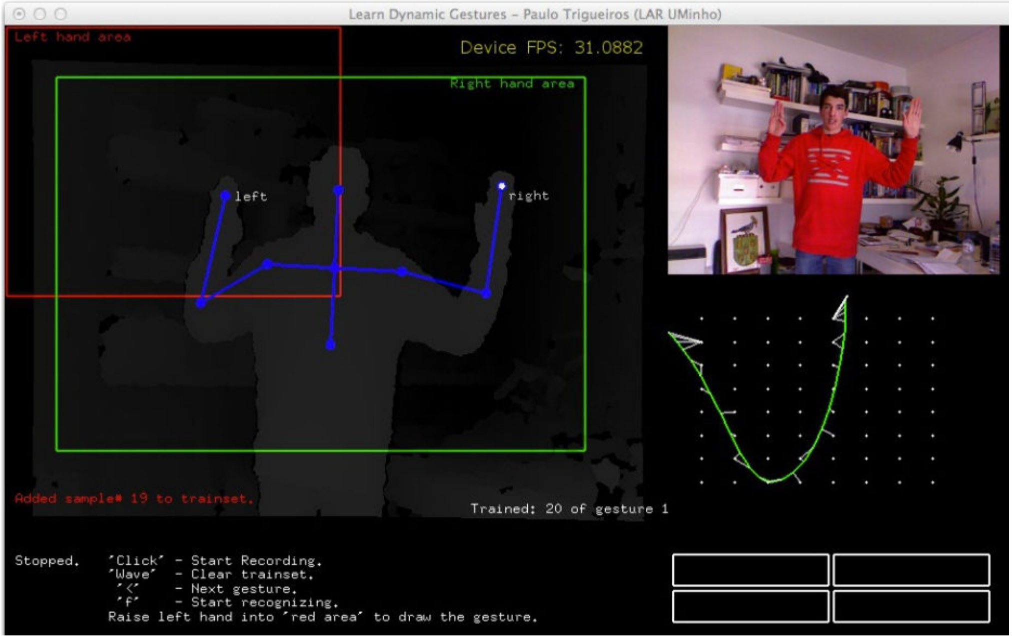


Figure 2.7. A 4-state Left-Right HMM model.

Figure 2.8 shows the main user interface for the application, with a hand path drawn on top of the centroids with the corresponding path distance to centroids drawn as white lines. For each gesture that required training, a dataset was built and the system trained in order to learn the corresponding model parameters. The number of observation symbols defined and implemented was 64 with 4 hidden states. Several values for the number of observations in the set {16, 25, 36, 49, 64, 81}, and hidden states, ranging from 2 to 12 were tried out during the experiments, without significant improvements for values greater than the selected ones.



```
Added sample# 19 to trainset.
Trained: 20 of gesture 1
Stopped.   'Click' - Start Recording.
           'Wave'  - Clear trainset.
            '<'    - Next gesture.
            'f'    - Start recognizing.
           Raise left hand into 'red area' to draw the gesture.
```

Figure 2.8. Dynamic gestures feature extraction and model training user interface.

For model testing, a new set of datasets were built with data from four different users with a total of 25 per gesture and per user, totalling 1100 records for the predefined 11 gestures (Figure 2.9).

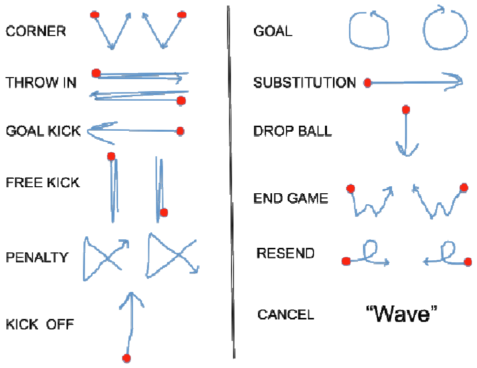


Figure 2.9. The set of dynamic gestures defined and used in the Referee CommLang.

These datasets were analysed with the previous obtained models and the final accuracy results obtained with equation 3 are represented in Table 2.3.

$$accuracy = \frac{\#\, correctly\ predicted\ class}{\#\, total\ testing\ class} \times 100\% \tag{3}$$

Table 2.3. Hidden Markov Models accuracy for each gesture defined

Gesture	1	2	3	4	5	6	7	8	9	10	11
Accuracy	75%	100%	100%	100%	92%	88%	92%	100%	100%	96%	88%

So, for the dynamic gesture recognition, with the obtained HMM models, an average accuracy of 93.72% was achieved.

3 Referee Command Language Interface System

To validate the proposed framework, an online system able to help a robotic soccer game referee judge a game in real time was implemented. The proposed solution combines a vision-based hand gesture recognition system with a formal language definition, the *Referee CommLang*, into what is called the *Referee Command Language Interface System* (ReCLIS). The system builds a command based on system-interpreted static and dynamic referee gestures, and is able to send it to a computer interface, which can then transmit the proper commands to the robots. The commands were defined in a new formal language described in Sect. 3.1.1. With the proposed solution, there is the possibility of eliminating the assistant referee, thereby allowing a more natural game interface.

The system uses only one camera, a Kinect camera (Chowdhury, 2012), and is based on a set of assumptions, hereby defined:

1. The user must be within a defined perimeter area, in front of the camera.

2. The user must be within a defined distance range, due to camera limitations. System defined values are 0.7m for the near plane and 3m for the far plane.
3. Hand pose is defined with a bare hand and not occluded by other objects.
4. The system must be used indoor, since the selected camera does not work well under sun light conditions.

The following sections describe the Referee Command Language Definition and the Referee CommLang Prototype implementation.

3.1.1 THE REFEREE COMMAND LANGUAGE DEFINITION

This section presents the *Referee CommLang* keywords with a syntax summary and description. The *Referee CommLang* is a new and formal definition of all commands that the system is able to identify. As in (Reis and Lau, 2002), the language must represent all the possible gesture combinations (static and dynamic) and at the same time be simple in its syntax. The language was defined with BNF (Bakus Normal Form or Bakus-Naur Form) (Backus et al., 1960):

- Terminal symbols (keywords and operator symbols) are in a constant-width typeface.
- Choices are separated by vertical bars '|' and in greater-than and less-than symbols (< choice>).
- Optional elements are in square brackets ([optional]).
- Sets of values are in curly braces ({set}).
- A syntax description is introduced with ::=.

The language has three types of commands: **Team commands**, **Player commands** and **Game commands**. This way, a language is defined to be a set of commands that can be a TEAM_COMMAND, a GAME_COMMAND or a PLAYER_COMMAND.
The TEAM_COMMAND is composed of the following ones: KICK_OFF, CORNER, THROW_IN, GOAL_KICK, FREE_KICK, PENALTY, GOAL or DROP_BALL.
A GAME_COMMAND can be the START or STOP of the game, a command to end the game (END_GAME), cancel the just defined command (CANCEL) or resend the last command (RESEND).
For the END_GAME command, it is necessary to define the game part, identified by PART_ID with one of four commands – 1ST, 2ND, EXTRA or PEN (penalties).

```
<LANGUAGE>::={<COMMAND>}
<COMMAND>::=<TEAM_COMMAND> | <GAME_COMMAND> | <PLAYER_COMMAND>□
<TEAM_COMMAND>::=<KICK_OFF> | <CORNER> | <THROW_IN> | <GOAL_KICK> | <FREE_KICK> |
      <PENALTY> | <GOAL> | <DROP_BALL>
<GAME_COMMAND>::=<START> | <STOP> | <END_GAME> | <CANCEL> | <RESEND>□
<PLAYER_COMMAND>::=<SUBSTITUTION> | <PLAYER_IN> | <PLAYER_OUT> | <YELLOW_CARD> |
      <RED_CARD>
```

For the TEAM_COMMANDS there are several options: KICK_OFF, CORNER, THROW_IN, GOAL_KICK, FREE_KICK, PENALTY and GOAL that need a TEAM_ID (team identification) command, that can be one of two values - CYAN or MAGENTA, and finally the DROP_BALL command.

```
<KICK_OFF> ::= KICK_OFF <TEAM_ID>
<CORNER> ::= CORNER <TEAM_ID>
<THROW_IN> ::= THROW_IN <TEAM_ID>
<GOAL_KICK> ::= GOAL_KICK <TEAM_ID>
<FREE_KICK> ::= FREE_KICK <TEAM_ID>
<PENALTY> ::= PENALTY <TEAM_ID>
<GOAL> ::= GOAL <TEAM_ID>
<DROP_BALL> ::= DROP_BALL
```

For the PLAYER_COMMAND, first there is a SUBSTITUTION command with the identification of the player out (PLAYER_OUT) and the player in (PLAYER_IN) the game with the PLAYER_ID command. The PLAYER_ID can take one of seven values (PL1, PL2, PL3, PL4, PL5, PL6, PL7). For the remaining commands: PLAYER_IN, PLAYER_OUT, YELLOW_CARD or RED_CARD, it is necessary to define the TEAM_ID as explained above, and the PLAYER_ID.

```
<SUBSTITUTION> ::= SUBSTITUTION <PLAYER_IN> <PLAYER_OUT>
<PLAYER_IN> ::= PLAYER_IN <TEAM_ID> <PLAYER_ID>
<PLAYER_OUT> ::= PLAYER_OUT <TEAM_ID> <PLAYER_ID>
<YELLOW_CARD> ::= YELLOW_CARD <TEAM_ID> <PLAYER_ID>
<RED_CARD> ::= RED_CARD <TEAM_ID> <PLAYER_ID>
<START> ::= START□
<STOP> ::= STOP□
<END_GAME> ::= END_GAME <PART_ID>
<CANCEL> ::= CANCEL□
<RESEND> ::= RESEND
<TEAM_ID > ::= CYAN | MAGENTA□
<PLAYER_ID> ::= PL1 | PL2 | PL3 | PL4 | PL5 | PL6 | PL7
<PART_ID> ::= 1ST | 2ND | EXTRA | PEN
```

3.2 Referee CommLang Prototype Implementation

The Human-Computer Interface (HCI) for the prototype was implemented using the C++ language, and the *openFrameworks* toolkit (Lieberman et al., 2004) with the OpenCV (Bradski and Kaehler, 2008) and the OpenNI (OpenNI, 2013) add-ons.

The proposed system involves three modules as can be seen in the diagram of Figure 3.1:

5. Data acquisition, pre-processing and feature extraction.
6. Gesture and posture classification with the models obtained in Sect. 2.1 and Sect. 2.2.
7. Gesture sequence construction or command classification.

As explained in Sect. 3, a referee command is composed by a set of dynamic gestures (Figure 2.9) and hand postures (Figure 2.2). The hand postures are used to identify one of the following commands: *team number*, *player number* or *game part*.

The problems of data acquisition, pre-processing, feature extraction and gesture classification were discussed in Sect. 2. The following section will describe the problem of modelling the command semantics for command classification.

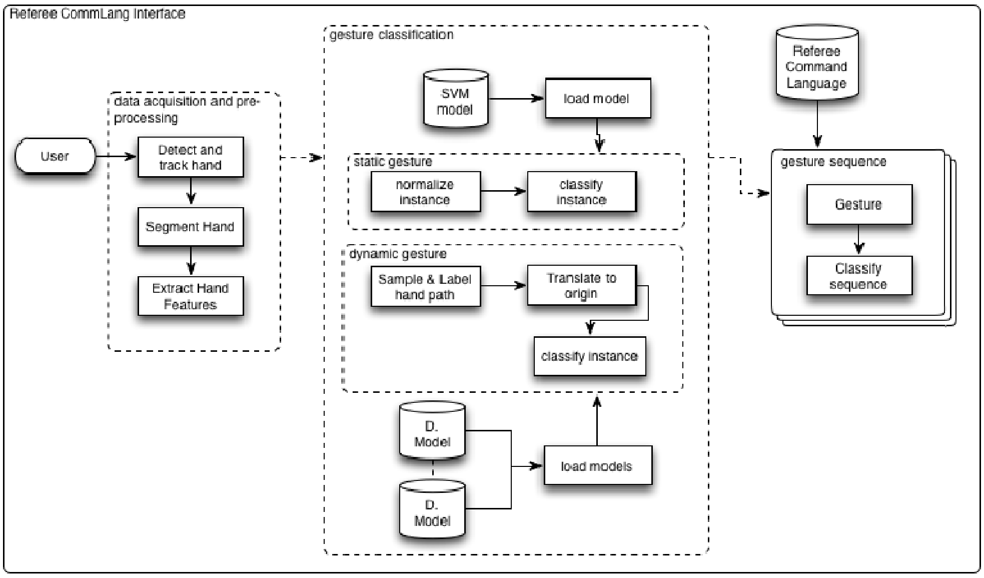


Figure 3.1. Referee CommLang Interface diagram.

3.3 COMMAND CLASSIFICATION

Since the system uses a combination of dynamic and static gestures, modelling the command semantics became necessary. A Finite State Machine is a usually employed technique to handle this situation (Buckland, 2005, Millington and Funge, 2009). In the implemented system, the FSM shown in the diagram of Figure 3.2 and described in the state transition Table 3.1 was implemented to control the transition between three possible defined states: DYNAMIC, STATIC and PAUSE. A *state transition table*, as the name implies, is a table that describes all the conditions and the states those conditions lead to. A PAUSE state is used to control the transitions between user postures and gestures and somehow eliminate all unintentional actions between DYNAMIC/STATIC and STATIC/STATIC gestures. This state is entered every time a gesture or hand posture is found, and exited after a predefined period of time or when a command sequence is identified, as can be seen in the state transition table.

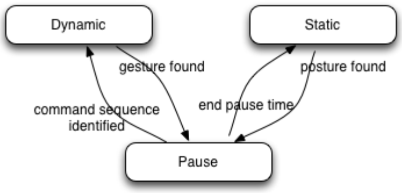


Figure 3.2. The Referee Command Language System finite state machine (FSM).

Table 3.1. The Referee CommLang state transition table.

Current State	Condition	Sate transition
Dynamic	Found gesture	Pause
Static	Found posture	Pause
Pause	End pause time	Static
Pause	Command sequence identified	Dynamic

The following sequence of images, Figure 3.3, Figure 3.4 and Figure 3.5, shows the Referee Command Language user interface with the "GOAL, TEAM1, PLAYER2" sequence of commands being recognized.

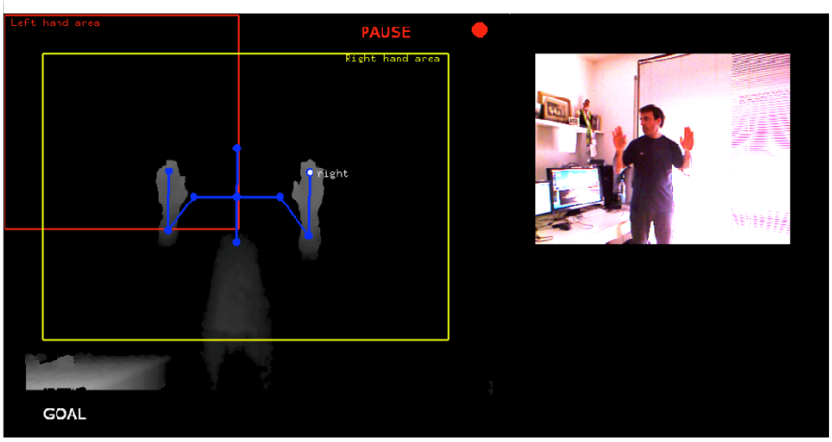


Figure 3.3. The "GOAL" gesture recognized.

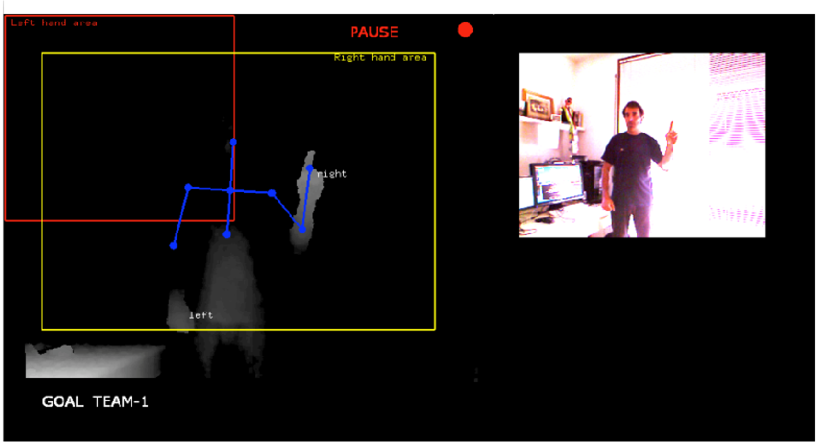


Figure 3.4. The "GOAL, TEAM1" sequence recognized.

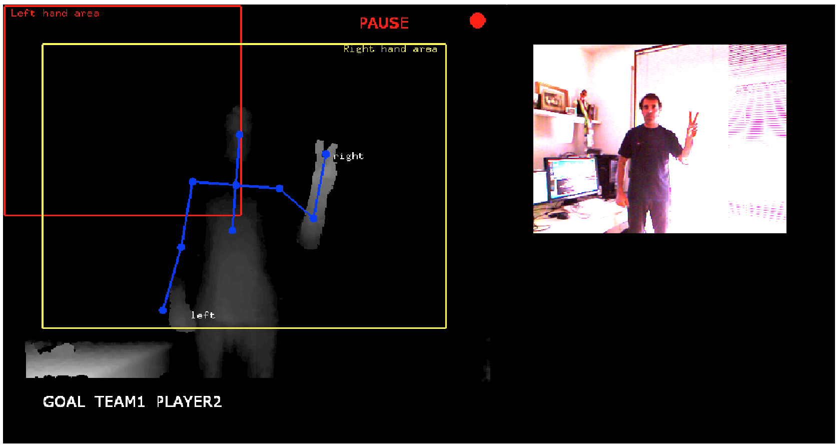


Figure 3.5. The "GOAL, TEAM1, PLAYER2" sequence recognized.

4 Sign Language Recognition Prototype

The *Sign Language Recognition Prototype* is a real-time vision-based system whose purpose is to recognize the Portuguese Sign Language given in the alphabet of Figure 2.3. The purpose of the prototype was to test and validate the proposed framework applied to the problem of real-time sign language recognition. For that, the user must be positioned in front of the camera, doing the sign language postures, that will be interpreted by the system and their classification will be displayed and spoken by the interface.

The diagram of Figure 4.1 shows the proposed system architecture, which consists of two modules, namely: the *data acquisition, pre-processing and feature extraction model* and the *sign language posture classification model*.

In the first module, the hand is detected, tracked and segmented from the video images. From the obtained segmented hand, features are extracted, as explained in section 2, for posture classification.

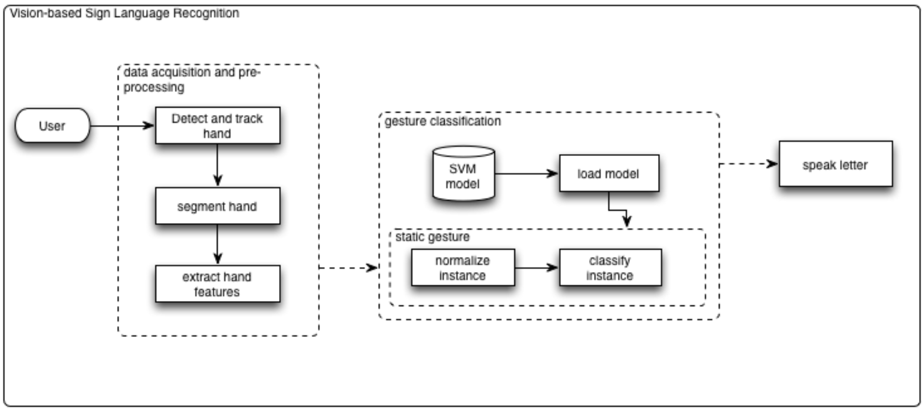


Figure 4.1. Sign Language Recognition Prototype diagram

4.1 Prototype Implementation

The Human-Computer Interface (HCI) for the prototype was developed using the C++ language, and the openFrameworks toolkit (Lieberman et al., 2004) with the OpenCV (Bradski and Kaehler, 2008) and the OpenNI (OpenNI, 2013) add-ons, ofxOpenCv and ofxOpenNI respectively. In the following two images it is possible to see the Sign Language Prototype with two vowels correctly classified and displayed on the right side of the user interface.

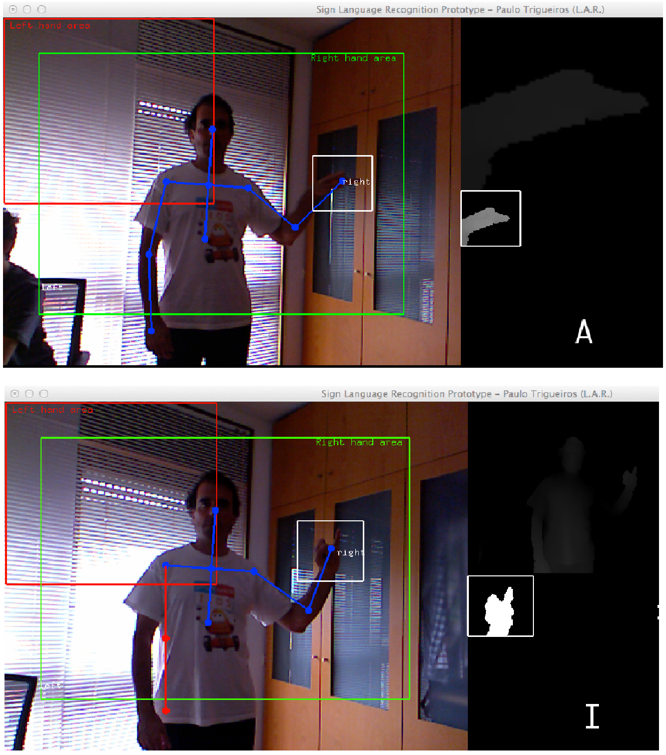

Sign Language Recognition Prototype – Paulo Trigueiros (L.A.R.)
Left hand area
Right hand area
right
A
Sign Language Recognition Prototype – Paulo Trigueiros (L.A.R.)
Left hand area
Right hand area
right
I

Figure 4.2. Sign Language prototype interface wit two vowels correctly classified.

5 Conclusions and Future Work

Hand gestures are a powerful way for human communication, with lots of potential applications in the area of human computer interaction. Vision-based hand gesture recognition techniques have many proven advantages compared with traditional devices. However, hand gesture recognition is a difficult problem and the current work is only a small contribution towards achieving the results needed in the field.

The main objective of this work was to study and implement solutions that could be generic enough, with the help of machine learning algorithms, allowing its application in a wide range of human-computer interfaces, for online gesture and posture recognition. To achieve this, a set of implementations for processing and retrieving hand user information, learn statistical models and able to do online classification were created. The final prototype is a generic solution for a vision-based hand gesture recognition system, which is able to integrate posture and gesture classification and that, can be integrated with any human-computer interface. The implemented solutions, based on supervised learning algorithms, are easily configured to process new hand features or to learn different hand postures and dynamic gestures, while creating statistical models that can be used in any real-time user interface for online gesture classification. For the problem of hand posture classification, hand features that give good classification results were identified, being at the same time simple in terms of computational complexity, for use in any real-time application. The selected features were tested with the help of the RapidMiner tool for machine learning and data mining. That way, it was possible to identify a learning algorithm that was able to achieve very good results in terms of pattern classification, and that was the one used in the final solution. For the case of dynamic gesture recognition, the choice fell on Hidden Markov Models, due to the nature of the data, gestures, which are time-varying processes. This type of models has proven to be very effective in other areas of application, and had already been applied successfully to the problem of gesture recognition. The evaluation of the trained gestures with the implemented prototypes proved that, it was possible to successfully integrate static and dynamic gestures with the generic framework and use them for human / computer interaction.

It was also possible to prove through this study, and with the various experiments, which were carried out, that proper feature selection for image classification is vital for the future performance of the recognition system. It was possible to learn and select sensible features that could be effectively used with machine learning algorithms in order to increase the performance and effectiveness of online static and dynamic gesture classification.

To demonstrate the effectiveness of our vision based gesture recognition system, the proposed methods were evaluated with two applications: the Referee CommLang Prototype and the Sign Language Recognition Prototype. The first one is able to interpret user commands defined in the new formal language, *the Referee CommLang*, created with the aim of interpreting a set of commands made by a robotic soccer referee. The second one is able to interpret Portuguese sign language hand postures.

An important aspect to report on the implemented solutions has to do with the fact that new users were able to learn and adapt to the systems very quickly and were able to start using them in a normal way after a short period of time, making them solutions that can be easily adapted and applied to other areas of application.

As future work and major development prospects it is suggested:

- Explore other machine learning algorithms applied to the problem of hand gesture classification and compare obtained results.
- Include not only the possibility of 3D gestures but also to work with several cameras to thereby obtain a full 3D environment and achieve view-independent recognition, thus eliminating some limitations of the current system.
- Explore the possibility of applying stereo vision instead of only depth range cameras, applied to human / computer interaction and particularly to hand gesture recognition.
- Introduce gesture recognition with both hands, enabling the creation of more natural interaction environments.
- Investigate and try to find more reliable solutions for the identification of the beginning and end of a gesture.
- Build systems that are able to recognize continuous gestures, i.e., without the need to introduce pauses for gesture or command construction.
- Explore reinforcement learning as a way to start with a reduced number of hand features per gesture, reducing the time to learn the models, and be able to learn with user interaction, possibly using multimodal dialog strategies.

- Explore unsupervised learning applied to gesture recognition. Give the robot/system the possibility to learn by interaction with the user, again with the possibility of multimodal strategies.

As a final conclusion one can say that although there is still much to do in the area, the implemented solutions are a solid foundation for the development of generic gesture recognition systems that could be used with any interface for human computer interaction. The interface language can be redefined and the system can be easily configured to train different set of postures and gestures that can be easily integrated with any desired solution.

6 ACKNOWLEDGMENTS

The authors wish to thank all members of the Laboratório de Automação e Robótica (LAR), at University of Minho, Guimarães. The authors would like to thank also, everyone who contributed to the hand data features acquisition phase, without which it would have been very difficult to carry out this study. Also special thanks to the Polytechnic Institute of Porto, the ALGORITMI Research Centre and the LIACC Research Center, for the opportunity to develop this research work.

7 REFERENCES

ALPAYDIN, E. 2004. Introduction to Machine Learning, MIT Press.

BACKUS, J. W., BAUER, F. L., GREEN, J., KATZ, C., MCCARTHY, J., PERLIS, A. J., RUTISHAUSER, H., SAMELSON, K., VAUQUOIS, B., WEGSTEIN, J. H., WIJNGAARDEN, A. V. & WOODGER, M. 1960. Revised Report on the Algorithmic Language ALGOL 60. Communications of the ACM. ACM.

BOURENNANE, S. & FOSSATI, C. 2010. Comparison of shape descriptors for hand posture recognition in video. Signal, Image and Video Processing, 6, 147-157.

BRADSKI, G. & KAEHLER, A. 2008. Learning OpenCV: Computer Vision with the OpenCV Library, O'Reilly Media.

BUCHMANN, V., VIOLICH, S., BILLINGHURST, M. & COCKBURN, A. 2004. FingARtips: gesture based direct manipulation in Augmented Reality. 2nd international Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia. Singapore: ACM.

BUCKLAND, M. 2005. Programming Game AI by Example, Wordware Publishing, Inc.

CAMASTRA, F. & VINCIARELLI, A. 2008. Machine Learning for Audio, Image and Video Analysis, Springer.

CHAUDHARY, A., RAHEJA, J. L., DAS, K. & RAHEJA, S. 2011. Intelligent Approaches to interact with Machines using Hand Gesture Recognition in Natural way: A Survey. International Journal of Computer Science & Engineering Survey, 2, 122-133.

CHOWDHURY, J. R. 2012. Kinect Sensor for Xbox Gaming. M.Tech CSE, IIT Kharagpur.

FINK, G. A. 2008. Markov Models for Pattern recognition - From Theory to Applications, Springer.

HASANUZZAMAN, M., AMPORNARAMVETH, V., ZHANG, T., BHUIYAN, M. A., SHIRAI, Y. & H.UENO. Real-time Vision-based Gesture Recognition for Human Robot Interaction. IEEE International Conference on Robotics and Biomimetics, August 22-26 2004 Shenyang, China. IEEE, 413-418.

HOLT, G. A. T., REINDERS, M. J. T., HENDRIKS, E. A., RIDDER, H. D. & DOORN, A. J. V. Influence of handshape information on automatic sign language recognition. 8th International Conference on Gesture in Embodied Communication and Human-Computer Interaction, February 25-27 2010 Bielefeld, Germany. 2127632: Springer-Verlag, 301-312.

HUANG, T. & PAVLOVIC, V. Hand Gesture Modeling, Analysis, and Synthesis. In Proc. of IEEE International Workshop on Automatic Face and Gesture Recognition, 1995. 73-79.

KIM, T. 2008. In-Depth: Eye To Eye - The History Of EyeToy [Online]. http://www.gamasutra.com. Available: http://www.gamasutra.com/php-bin/news_index.php?story=20975 [Accessed 29-03-2013 2013].

KING, D. E. 2009. Dlib-ml: A Machine Learning Toolkit. Journal of Machine Learning Research, 10, 1755-1758.

KRATZ, S. & ROHS, M. 2011. Protractor3D: a closed-form solution to rotation-invariant 3D gestures. 16th International Conference on Intelligent User Interfaces. Palo Alto, CA, USA: ACM.

LI, Y. 2010. Protractor: a fast and accurate gesture recognizer. Conference on Human Factors in Computing Systems. Atlanta, Georgia, USA: ACM.

LIEBERMAN, Z., WATSON, T. & CASTRO, A. 2004. openFrameworks [Online]. Available: http://www.openframeworks.cc/ 2011].

MAUNG, T. H. H. 2009. Real-Time Hand Tracking and Gesture Recognition System Using Neural Networks. Proceedings of World Academy of Science: Engineering & Technology, 50, 466-470.

MILLINGTON, I. & FUNGE, J. 2009. Artificial Intelligence for Games, Elsevier.

MINER, R. RapidMiner : Report the Future [Online]. Available: http://rapid-i.com/ [Accessed December 2011.

MITRA, S. & ACHARYA, T. 2007. Gesture recognition: A Survey. IEEE Transactions on Systems, Man and Cybernetics. IEEE.

MONTGOMERY, D. C. & RUNGER, G. C. 1994. Applied Statistics and Probability for Engineers, Wiley.

MURPHY, K. 1998. Hidden Markov Model (HMM) Toolbox for Matlab [Online]. Available: http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html 2012].

MURTHY, G. R. S. & JADON, R. S. 2009. A Review of Vision Based Hand Gestures Recognition. International Journal of Information Technology and Knowledge Management, 2, 405-410.

ONG, S. C. & RANGANATH, S. 2005. Automatic sign language analysis: a survey and the future beyond lexical meaning. IEEE Trans Pattern Anal Mach Intell, 27, 873-91.

OPENNI. 2013. The standard framework for 3D sensing [Online]. Available: http://www.openni.org/.

RABINER, L. R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 77, 257-286.

RABINER, L. R. & JUANG, B. H. 1986. An introduction to hidden Markov models. IEEE ASSp Magazine.

REIS, L. P. & LAU, N. 2002. COACH UNILANG - A Standard Language for Coaching a (Robo) Soccer Team. In: BIRK, A., CORADESCHI, S. & TADOKORO, S. (eds.) RoboCup 2001: Robot Soccer World Cup V. Springer Berlin Heidelberg.

SAYAD, D. S. 2010. Support Vector Machine - Classification (SVM) [Online]. Available: http://www.saedsayad.com/support_vector_machine.htm [Accessed 8 Nov 2012.

TARA, R. Y., SANTOSA, P. I. & ADJI, T. B. 2012. Sign Language Recognition in Robot Teleoperation using Centroid Distance Fourier Descriptors. International Journal of Computer Applications, 48.

THEODORIDIS, S. & KOUTROUMBAS, K. 2010. An Introduction to Pattern Recognition: A Matlab Approach, Academic Press.

TRIGUEIROS, P., RIBEIRO, F. & LOPES, G. Vision-based hand segmentation techniques for human-robot interaction for real-time applications. In: TAVARES, J. M. & JORGE, R. M. N., eds. III ECCOMAS Thematic Conference on Computational Vision and Medical Image Processing, 12-14 de Oubtubro 2011 Olhão, Algarve, Portugal. Taylor and Francis, Publication 31-35.

TRIGUEIROS, P., RIBEIRO, F. & REIS, L. P. A comparison of machine learning algorithms applied to hand gesture recognition. 7th Iberian Conference on Information Systems and Technologies, 20-23 July 2012 Madrid, Spain. 41-46.

TRIGUEIROS, P., RIBEIRO, F. & REIS, L. P. A Comparative Study of different image features for hand gesture machine learning. 5th International Conference on Agents and Artificial Intelligence, 15-18 February 2013 Barcelona, Spain.

VATAVU, R.-D., ANTHONY, L. & WOBBROCK, J. O. 2012. Gestures as point clouds: a $P recognizer for user interface prototypes. 14th ACM international conference on Multimodal interaction. Santa Monica, California, USA: ACM.

VIJAY, P. K., SUHAS, N. N., CHANDRASHEKHAR, C. S. & DHANANJAY, D. K. 2012. Recent Developments in Sign Language Recognition : A Review. International Journal on Advanced Computer Engineering and Communication Technology, 1, 21-26.

WIKIPEDIA. 2012. Língua gestual portuguesa [Online]. Available: http://pt.wikipedia.org/wiki/Lingua_gestual_portuguesa 2013].

WITTEN, I. H., FRANK, E. & HALL, M. A. 2011. Data Mining - Pratical Machine Learning Tools and Techniques, Elsevier.

WOBBROCK, J. O., WILSON, A. D. & LI, Y. 2007. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. Proceedings of the 20th annual ACM symposium on User interface software and technology. Newport, Rhode Island, USA: ACM.

WU, Y. & HUANG, T. S. 1999. Vision-Based Gesture Recognition: A Review. Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction. Springer-Verlag.

YOON, J.-H., PARK, J.-S. & SUNG, M. Y. Vision-Based bare-hand gesture interface for interactive augmented reality applications. 5th international conference on Entertainment Computing, September 20-22 2006 Cambridge, UK. 2092520: Springer-Verlag, 386-389.

ZAFRULLA, Z., BRASHEAR, H., STARNER, T., HAMILTON, H. & PRESTI, P. 2011. American sign language recognition with the kinect. 13th International Conference on Multimodal Interfaces. Alicante, Spain: ACM.