Gesture based Human Multi-Robot Interaction

Gerard Canal
Facultat d'Informàtica de Barcelona (UPC)
Computer Vision Center, UAB
Campus UAB, Edifici O
08193 Bellaterra, Barcelona, Spain
Email: gerard.canal@cvc.uab.cat

Cecilio Angulo
Facultat d'Informàtica de Barcelona
Universitat Politècnica de Catalunya (UPC)
Jordi Girona Salgado 1-3
08034 Barcelona, Spain
Email: cecilio.angulo@upc.edu

Sergio Escalera
Computer Vision Center, UAB
Matemàtica Aplicada i Anàlisi
Universitat de Barcelona
Gran Via 585, 08007 Barcelona, Spain
Email: sergio@maia.ub.es

***Abstract*—The emergence of robot applications for non-technical users implies designing new ways of interaction between robotic platforms and users. The main goal of this work is the development of a gestural interface to interact with robots in a similar way as humans do, allowing the user to provide information of the task with non-verbal communication. The gesture recognition application has been implemented using the Microsoft's Kinect$^{\text{TM}}$ v2 sensor. Hence, a real-time algorithm based on skeletal features is described to deal with both, static gestures and dynamic ones, being the latter recognized using a weighted Dynamic Time Warping method. The gesture recognition application has been implemented in a multi-robot case. A NAO humanoid robot is in charge of interacting with the users and respond to the visual signals they produce. Moreover, a wheeled Wifibot robot carries both the sensor and the NAO robot, easing navigation when necessary. A broad set of user tests have been carried out demonstrating that the system is, indeed, a natural approach to human robot interaction, with a fast response and easy to use, showing high gesture recognition rates.**

I. Introduction

Robotic systems are able to autonomously navigate, walk, talk, understand spoken language, detect objects, people, obstacles. Moreover, reasoning methodologies have also been applied, allowing robots to design plans to achieve their objectives. Once robots are able to fulfil thousands of useful tasks, they also need to communicate with human beings. Many Human Robot Interaction (HRI) techniques focus on spoken dialogues, often restricted and constrained to simple questions – response or vocal orders. However, most of the human communication is performed by non-verbal channels [1], [2]. For instance, humans tend to use deictic gestures in order to refer to an object which is nearby, rather than performing a verbal description of it.

This work is concerned in natural interaction with a robot by means of gestures. We focus in the recognition of dynamic gestures such as a wave, and in static gestures like the pointing gesture. It has been implemented in a multi-robot system with an Aldebaran's NAO robot and a Wifibot robot, as well as a Kinect$^{\text{TM}}$ 2 sensor is used to get the data related to the human gestures. The Wifibot is in charge of the long term navigation and the transportation of the Kinect$^{\text{TM}}$ sensor, and the NAO robot, which is seated on the Wifibot, is used to perform the verbal and gestural interaction from the robotics side, being able to go down the wheeled robot to finish a task by standing.

The task has been evaluated with 24 users from different backgrounds and age groups, showing that the system performs well in terms of time and accuracy, as well as it is a natural way of interaction with the robot.

The rest of the paper is organized as follows: related work is reviewed in Section II. Section III introduces the hardware and software resources available to perform the interaction. Next, it is explained how are they included in the system in Section IV. Section V highlights the main theory behind the gesture recognition methods and Section VI explains how are they used to interact with the user. The obtained results from offline experiments and the user evaluation are shown in Section VII, meanwhile Section VIII concludes the work and give some insights about future improvements.

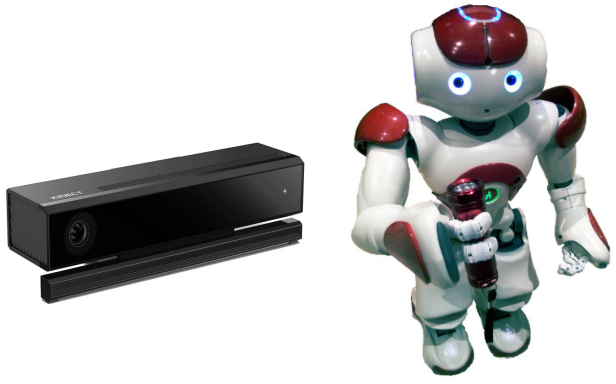
II. Related work

Human Robot Interaction for social robotics is an active research field from many different points of view: from making humans understand the robot states through verbal and non verbal communication to doing it the other way around, making the robot understand humans.

Focusing in the gesture recognition part, a real time gesture recognition method using Artificial Neural Networks is introduced in [3]. The recognition is performed in both hands, using a hand independent representation which is obtained from salient motion features extracted from depth data. The gestures are represented as a sequence of such motion patterns. Then, Self Organising Maps (SOM) are used to cluster the motion data. Experiments on HRI data to operate a robot with gestures showed good performance with high recognition rates. Dynamic Time Warping (DTW) approaches, as the one used in this work, are also widely used for gesture recognition purposes. A gesture recognition method developed in [4] is applied on data coming from accelerometers and gyroscopes in real time; the method is applied to RGB and depth data using a probability approach in [5], and [6], [7] applies DTW in weighted skeletal features obtained from a depth sensor.

As for explicit applications to HRI, a low cost RGB-D sensor was used in [8] to perform dynamic gesture recognition by skeleton tracking. The recognition method uses a Finite State Machine which encodes the temporal signature of the gesture. Another Kinect$^{\text{TM}}$ application to gesture recognition with Hidden Markov Models (HMMs) and skeletal data is presented in [9], in which the user performs gestures to control the robot and it responds with either voice or a message in the display. Deep Neural Networks have also been used to recognise gestures, as performed in [10], aiming to recognise gestures in real time with minimal preprocessing in RGB images. High accuracy was obtained with on line performance, where the robot provides speech feedback. User defined gestures can be added in a semi supervised way to the system from [11], which contributes a non-parametric stochastic segmentation algorithm, the Change Point Model. This procedure does not need to be supplied with the gesture's starting and ending points, making the user able to define its own gestures to control a robot and thus being highly customisable without the need of explicit user learning or adaptation. Applied to elderly people caring, a Kinect$^{\text{TM}}$ based approach to recognise calling gestures is proposed in [12]. This approach uses a skeleton based recognition system to detect when the user is standing up, and an octree one when the skeleton is not properly tracked. Erroneous skeletons are filtered by face detection in order to determine whether the data is actually a person or a false positive. An application to object handling to the user is implemented and tested with different elder users.

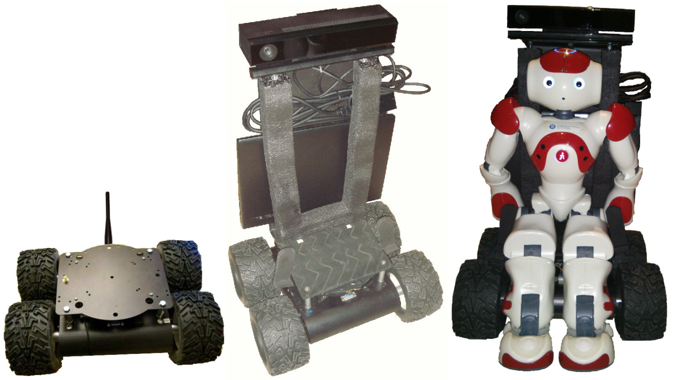
Deictic and pointing gestures are also widely studied. Pointing gesture recognition and direction estimation is performed in [13] by means of a cascade of HMMs and a particle filter to recognise the gesture in stereo images to which hand and face tracking is applied to capture the pointing direction. A similar HMM approach is used in [14] to recognise pointing gestures. A ROS-based robot is used in [15] to detect pointing gestures by means of a Haarlet-based hand gesture recognition system, extract the pointing direction and translate it to movement goals in a map. A tracking system is presented in [16] which recognises the pointing gesture so that a person can tell the robot where is another person who wants to interact with it. Finger segmentation is performed to compute the angle to which the robot has to turn its head. A research about how people refer to objects in the world is carried out in [17]. This deictic interaction comes from both speech and gesture channels. Spatial information from objects is extracted in form of features such as distance to the hand or its direction relative to the object. A K-SVD algorithm is trained to perform the classification. The pointed location on a wall is obtained in the system of [18], which uses geometry analysis to identify shoulders and elbows to understand gestures and obtain the direction. Some constraints in the study include high illumination environments and user wearing half sleeves to better segment him. In [19], pointing gestures are used to refer to objects by means of a time-of-flight camera to get depth information. They use the line between the person's eyes and their hand as the pointing direction. Knowledge about possible object locations is exploited in [20] in order to discern between which object might be pointed, using the Dempster-Shafer theory of evidence to join information from the head pose and the pointing hand's orientation.



(a) Microsoft's Kinect$^{\text{TM}}$ v2 sensor.

(b) Aldebaran's NAO robot.

Fig. 1: Kinect$^{\text{TM}}$ sensor and NAO robot.



(a) Initial Wifibot robot.

(b) Wifibot adaptation.

(c) Wifibot carrying the Kinect$^{\text{TM}}$ and the NAO.

Fig. 2: Wifibot robot and its modifications.

III. Resources

Several resources have been used for this work, both hardware and software. The hardware ones include:

- A Microsoft's Kinect$^{\text{TM}}$ version 2 sensor (see Figure 1a). Released in July 2014, it provides an infrared sensor, a depth one and a high definition RGB camera along with a microphone array. Some improvements with respect to its previous version include better colour and depth image quality. The SDK also includes software for tracking up to six people at the same time and body information of 24 joints.
- A middle size humanoid robot, Aldebaran's NAO version 3.2, which is shown in Figure 1b.
- A wheeled robotic platform, the Wifibot lab v3 from Nexter Robotics, is used to carry the Kinect$^{\text{TM}}$ sensor and the NAO robot. Some adaptations were added in order to carry the vision sensor, a laptop to get the information from the Kinect$^{\text{TM}}$ and the humanoid. The platform is shown in Figure 2, with the original version, the modified one and the modified version with the NAO seating on it.

The software resources include:

- The Kinect$^{\text{TM}}$ for Windows SDK version 2.0 to get Kinect$^{\text{TM}}$ body tracking information.
- The Point Cloud Library (PCL) to handle and process the depth data.

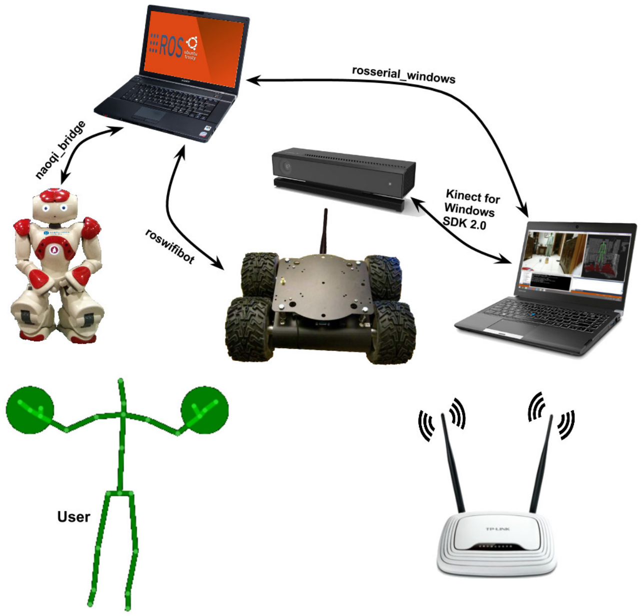


Fig. 3: System architecture.

- The Robot Operating System (ROS) to program the robots.

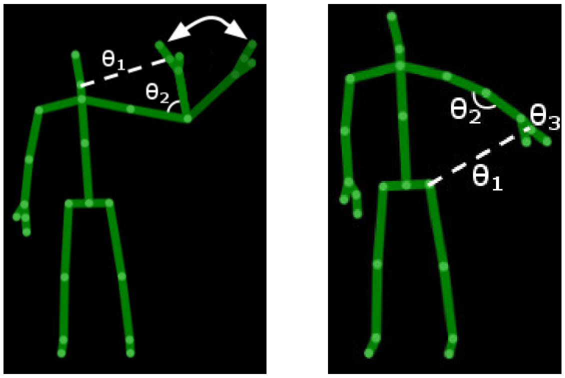
IV. System overview

The implemented system gathers all the resources and connects them to conform the application. The system architecture is displayed in Figure 3.

The user interacts with the system by using gestures. For the experiment, two gestures have been considered: the wave gesture (‘salute’) and point at one. When the salute is performed, the robot is expected to wave back at the user. In case the gesture performed is to point at some object, the robot should navigate to the object location – provided that the user has pointed at an object and neither to an empty space nor a non-ground position – and make clear that it has recognized which object the user was selecting. In case of ambiguity about the selected object, the robot will ask the user some questions related to the size or position of the detected objects, so the user can clarify, using spoken language, which object was the desired one.

V. Gesture recognition

This section focuses on the methodology used in order to perform gesture recognition. Given the high time constraints of the system, the recognition must be done in real time. A system in which the user performs the gesture and sees the reaction of its gesture after a long time is not much robust nor reliable, and would give raise to a non pleasant and confusing interaction. To achieve a natural, human-like interaction, the system must be able to understand human-like gestures taking roughly the same time that would take to another human. In the proposed solution, the robot reaction to a user gesture can be seen just after its execution has been finished.



(a) Skeletal wave gesture and used features.

(b) Skeletal pointing gesture and used features.

Fig. 4: Examples of skeletal features.

Two types of gesture are described in the system: the static and the dynamic gestures.

- A *static gesture* is defined by a static position without any movement of the user in neither the whole nor a limb of their body. The pointing gesture belongs to this category.
- In contrast, a *dynamic gesture* is defined by the movement the user performs with a part of the body as, for instance, the right arm. The wave gesture is an example.

A. Skeletal features for gesture recognition

The gesture recognition has been performed by extracting body tracking information from the Kinect$^{\text{TM}}$ v2 sensor. Using the Kinect$^{\text{TM}}$ for Windows SDK v2.0, skeletal information can be extracted from the depth images of the sensor [21]. The method consists of a Ranfom Forest classification at voxel level of body limbs and a skeletal design obtained from body limbs segmentation.

In the gesture recognition approach that is defined in this work, a different feature space representation is defined for each gesture. There is no need that all the gestures are defined by the same parameters or set of features, given that some gesture may be better characterised by using some specific information of a limb. The following paragraphs describe the features used for each gesture. Note that as the skeletal information is obtained in real world coordinates, all the used features are scale and body orientation invariant. Such properties avoid the need of feature preprocessing and normalisation, speeding up the process.

1) Wave gesture’s features: The wave gesture is performed by moving the forearm near and far, while keeping the upper arm in an horizontal position with the floor. Figure 4a illustrates the gesture. To characterise it, only two features are used:

- θ_1 Euclidean distance between the Neck joint and the Hand joint.
- θ_2 Angle in the Elbow joint, which is the angle between the vector from the Elbow joint to the Shoulder, and the vector from the Elbow to the Hand joints.

2) Point At gesture's features: The *Point At* gesture is a static gesture, so no movement is involved. The gesture is defined by the elongated arm position of the user. The features for this gesture are:

- θ_1 Distance between the Hand and the Hip joints (both of the same body part).
- θ_2 Angle in the Elbow joint, as in the wave gesture.
- θ_3 Position of the Hand joint.

A skeletal representation of the gesture and its features is shown in Figure 4b.

B. Dynamic time warping gesture recognition

The main gesture recognition method used in this work is the Dynamic Time Warping (DTW), and it is applied to dynamic gesture recognition. This algorithm is generally employed as a template matching method to measure the similarity or the cost of alignment between two temporal sequences which may differ in speed or length, finding an alignment warping path.

The method, originally described in [22], is also widely used to recognise gestures by detecting input sequences which are similar enough to a given reference gesture. Once detected, the whole gesture can be segmented from the input sequence by getting its warping path. Many examples of application can be found in the literature. For instance, [5] proposes a probability based DTW to recognise gestures in video streams with colour and depth information. They use a Bag-of-Visual-and-Depth-Words (BoVDW) representation for the gesture information. Their approach uses the DTW to perform the segmentation of an idle gesture which is performed between gestures. Once they have the input sequence segmented, a BoVDW classification is performed by using a k-Nearest Neighbours classifier. The authors of [7] propose a robust recognition based on feature preprocessing and weighting, using as features the whole body skeleton (joint values). They use weights for the different joints and gestures to improve the discriminant capabilities of the DTW. It is a similar approach as the one in [6], from which this work is mainly based on, where the authors propose a begin-end gesture recognition system with DTW. They also use skeletal joints information as the input features to the algorithm, and weight those features (each joint) depending on its participation in a particular gesture (for instance, legs are not much important in a handshaking gesture). These weights are obtained by a training algorithm based on the ground truth of the gestures.

The method which we propose is based on the contribution of [6], however it presents some differences. First, our features are not the whole skeleton but some metrics extracted from the joints of interest. Hence, the position of the non related limbs are not taken into account, avoiding the noise they would generate (as in the handshaking example). Secondly, we do not need the actual segmentation of the gesture in a begin-ends manner, because knowing which gesture has the user performed is enough. Furthermore, different number of features are allowed for each gesture by the framework, along with a weighting on those features to add discriminant power in case of some metrics being more important, or for numerical scaling purposes (to set them to have equal importance).

The DTW algorithm works as follows: be the gesture reference model a sequence $R = \{r_1, \ldots, r_m\}$ and the current input sequence $S = \{s_1, \ldots, s_\infty\}$. An alignment matrix $M_{m \times n}$, where n is the length of the temporal window from the input sequence S, is derived in which $M_{i,j}$ contains the distance between r_i and s_j. The input sequence S has infinite length as the system keeps getting feature frames and processing them until a gesture has been recognized. The distance metric which has been used to compute the alignment cost between two feature vectors is a weighted L_1 distance, being it defined as

$$d_1(r, s) = \sum_{i=1}^{k} \alpha_i |r_i - s_i| \,, \tag{1}$$

where α_i are the positive weights associated with the i-th feature, and k is the number of features of the gesture ($k = 2$ in the case of the wave gesture).

A warping path is defined as a set of neighbouring matrix elements which define a mapping between the reference model R and the current sequence S. More formally, a warping path $W = \{w_1, \ldots, w_T\}$ can be defined, being T the length of the path, and each element w_t corresponding to a matrix position $M[w_t] = M_{i,j}$. The objective warping path is the one which minimizes the warping cost,

$$DTW(R, S) = min \left\{ \frac{1}{T} \sqrt{\sum_{t=1}^{T} M[w_t]} \right\} . \tag{2}$$

Note that, even though the warping path computation has been implemented in the system for testing purposes, it is not used while the online gesture recognition is being performed.

The recurrence which the dynamic programming algorithm computes to get the alignment cost is

$$M_{i,j} = d_1(r_i, s_j) + min\{M_{i-1,j}, M_{i-1,j-1}, M_{i,j-1}\}. \tag{3}$$

To perform the detection of the ending of a known gesture in an input sequence S, a segment of it which is similar enough to a model gesture should be found. Given that a perfect match is almost impossible, a test sequence is considered similar enough to a model sequence if the following condition is satisfied,

$$M_{m,k} < \mu, \; k \in [1, \ldots, \infty], \tag{4}$$

where μ is a cost threshold associated with the gesture. The algorithm runs in a thread which is in charge of the corresponding gesture, being all the gestures processed in a parallel way, and keep running until one of the threads finds a gesture in the input sequence. Once this happens, all the gesture recognition threads stop their execution to return the recognition result.

The different parameters of the algorithm, such as the α_i and μ of each gesture, have been experimentally chosen with a parameter selection method based on different example sequences which have been manually labeled. Before this parameter selection, some tests were performed to observe the value of the different features while performing the gestures, obtaining from them a set of feasible parameter values for the α and μ parameters. After this, the parameter selection method consisted on using the recorded sequences, which were performed by different users, to get the values from the set which maximised the performance in terms of overlap. Such performance was computed by testing each sequence as if it was a real input sequence, using the DTW with the current parameters and checking the obtained performance, keeping those parameters that got better results.

C. Static gesture recognition

Given that the static gesture recognition does not require from temporal warping but just its spatial configuration, the DTW has not been used to recognise the pointing gesture. The proposed solution to this problem was to adapt the recognition and make a method to handle static gestures. The method is simple: it checks whether the input frames features are above some recognition thresholds during a certain number of frames. Another constraint is that a characteristic joint of the limbs which feature the gestures' movement is also within a given threshold.

More specifically, the method checks that the distance between hand and hip (θ_1) and the elbow angle (θ_2) are greater than certain threshold values ($\theta_1 > T_1,\ \theta_2 > T_2$) and the hand position (θ_3) is hold still during the given number of frames for the gesture (around 20 frames). Consequently, the parameters involved in the static gesture recognition are the feature thresholds, the minimum number of frames the gesture has to be performed and a reference to the limb position such as the hand. Those parameters are obtained in a similar way as the one used for the dynamic gestures.

D. Joint static and dynamic gesture recognition

A single multi-threaded algorithm takes care of both static and dynamic gestures by distributing the gestures in different threads, and stopping them in case a thread recognises a gesture. It also handles the possible situation of multiple gesture detections in the same frames. In case this happens, the gesture with less cost is the final recognized one.

E. Pointed point extraction and object segmentation

Provided that the recognized gesture is the pointing one, some post processing is needed in order to obtain gesture-related information. For instance, the pointed location needs to be extracted, and the objects near this position need also to be detected.

Just three elements are needed to obtain the pointing position: the ground plane description (such as a vector which is orthogonal to it), a point from the ground plane and the pointing direction. With this, a simple geometric line-plane intersection can be computed to obtain the desired point. The floor plane is obtained by means of the PCL's plane segmentation algorithm, described in [23]. The depth image from the Kinect^{TM} is represented as a Point Cloud and the segmentation algorithm is applied to find the plane which corresponds to the floor, by comparing it with a direction vector which is stored by the system (or user intervention is required in the opposite case, in which the user has to select three points of the ground).

Then, the line equation of the pointing direction is obtained in order to be able to get the intersection with the plane. Such line is extracted from the skeletal data, using the mean of the joint's position during ten contiguous frames from the middle-end of the gesture to make sure the correct direction is obtained and overcome possible tracking errors. The pointed direction is the one from the elbow to the hand joints. Tests performed by using the Hand to HandTip joints direction to get the actual finger pointing direction did not improve the results but rather produced more deviated locations, due to skeleton estimation inaccuracies. A last verification is performed in order to check that the pointing was directed to the ground. The method does not need that the pointed direction is inside the sensor's field of view, and it could also be used with other planar surfaces such as tables or shelves in case the sensor was installed in a height enough to see them.

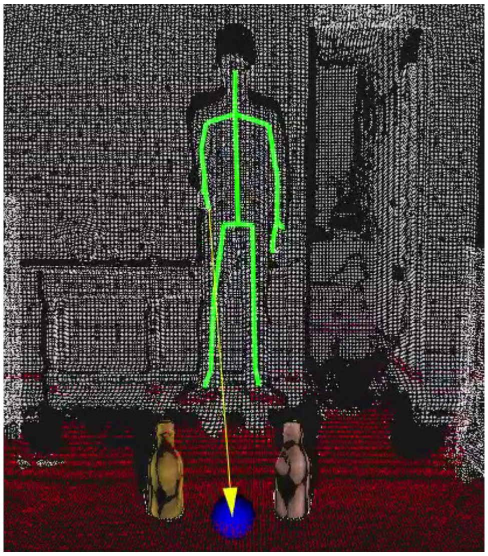


Fig. 5: Detecting objects around the pointed point.

Once the pointed point has been located, the next step is to detect which objects are around this location. As the aim is to make the robot know which is the object the user is referring to, there is not much need in recognising the objects but just detecting them, knowing there are objects there. Therefore, object recognition has not been used even though the system could be extended to handle it and actually recognise the objects and tell them by their name. The objects are detected by extracting a sphere of the point cloud and applying a clustering algorithm to it in order to isolate the objects in different clusters. An Euclidean cluster extraction method is used for this purpose [23]. Figure 5 shows an example of the result of this methods in the system's GUI.

VI. Mobile robotics interaction

The computer vision methods explained are used by the robotics system in order to perform the interaction with the human user. Previously to this interaction, some skills were added to the robots. Firstly, a simple PID controller [24] to control the heading direction of the Wifibot was implemented in order to approach the pointed location, assuming free path to the place. Secondly, a movement for the NAO was developed in order to make it go down the Wifibot to finish the task walking towards the object. Finally, smaller behaviours were implemented in the form of a hierarchical Finite State Machine (FSM) to control all the application flow.

One special sub FSM is the one that takes care of the object disambiguation. In case that the pointed object is not clear, that is, the distances between the pointed point and each detected object are similar. Provided this situation takes place, the robot starts a disambiguation interaction process with the user by means of an oral dialogue. If the detected objects are of different size, the robot asks simple questions like if the desired object is the biggest one. In case of the user's answer being negative, it asks about if it is the smallest one in case there are more than two objects, or it knows it is the remaining one in case of two objects. Similarly, the robot asks if it is the left-most one and performs the same procedure when the object's size is not discriminative enough.

Furthermore, the speech utterances performed by the robot are different each time, choosing them at random from a pool of sentences. With this, the interaction feels less repetitive, more natural and less boring.

VII. Results

The proposed methods have been evaluated both in offline tests and experiments carried out with volunteer users.

The overlap (also known as Jaccard Index), defined as

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}, \tag{5}$$

has been used to assess the offline performance of the gesture recognition methodology. A labelled set of seven sequences in which three different users perform gestures has been used for this purposes. Those sequences contain a total of 2082 gesture frames and 61 gestures, 27 of them being static gestures and the other 34 dynamic gestures. Notice that the reference model sequences used for the DTW were specifically recorded for it and are not part of this set.

To obtain a general measure of the performance of the algorithm on the recorded skeletal sequences, a LOOCV strategy has been used. Hence, one sequence is left out from the threshold selection method and the other ones are used to compute the better thresholds to recognise the gestures they include.

After the parameters were computed, the test sequence was evaluated to obtain its overlap measure in unseen data. This procedure is repeated for each one of the sequences, obtaining the performance measures both for the static gestures and the dynamic ones. Those measures were averaged for the seven sequences, obtaining a general Jaccard Index of each type of gesture, and the average of the mean overlap of both categories gives the final gesture recognition performance value of the system. The results, shown in Table I, where each fold is a test sequence, and the results of the parameter selection with all the sequences but the fold sequence are in the left columns, while the right ones show the performance on the fold sequence with the parameters obtained with the rest of the data. The "Global" row shows the results of the parameter selection when all the sequences are used to compute the parameters, and the test sequence columns show the mean of the above rows.

As it can be seen in Table I, the mean Jaccard Index in unseen sequences is about 0.49, segmenting most of the gestures but being not accurate at begin-end frame level spotting, which is not a critical point for gesture recognition since most of the gestures are recognized even with this gap. Dynamic gestures are better segmented than the static ones, even though more data could improve these results.

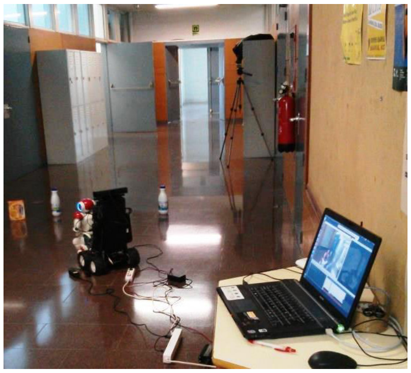


Fig. 6: User evaluation tests environment.

For the user evaluation tests, 24 volunteers used the system and provided their feedback. Each user performed three tries of the task: one in which only one object was placed in the robot area, another one with two of them and a final one with one of the two objects replaced. Those who desired it could make more tests. The objects which were used were two milk bottles and an empty cookie box. The order of the tries was changed between users to avoid any bias in the results due to user fatigue. The objects used in each of the test was also varied, being some tests performed with the two bottles and others with a bottle and the box. Figure 6 shows the tests set-up and environment. There were no restrictions about the order of the gestures to be performed, but users tended to begin with a wave gesture to then point at an object. Also, the objects were usually placed by the test controller, but those who asked were allowed to place the objects by themselves.

At the end of the test, users filled a questionnaire about the experience. This survey included demographic questions, obtaining a varied set of age groups and backgrounds, as seen in Figure 7, but being most of them male users. The answers to the interaction questions were positive. Almost all the users got the behaviour they expected from the robotic system. The answer to the wave gesture was considered fast, meanwhile the pointing one could be faster and some users thought they had to point for too much time. Moreover, the robot clearly showed which was the referred object, with a hand gesture. Also, the disambiguation part proved to be successful.

About the naturalness of the interaction, Figure 8 proves it was agreed to be natural, intuitive and easy to perform. Observed externally, the users showed a good learning curve, adapting their gestures to possible deviations of the pointing direction, pointing to the objects base rather than on top.

VIII. Conclusions and future work

A real-time gesture based HRI system has been proposed and implemented in this work using the Microsoft's Kinect$^{\text{TM}}$ v2 sensor. Two types of gestures have been recognized, the static gestures and the dynamic ones, and a gesture of each type has been included in the system. Features obtained from skeletal information have been used in the algorithm and

TABLE I: Gesture recognition performance evaluation results.

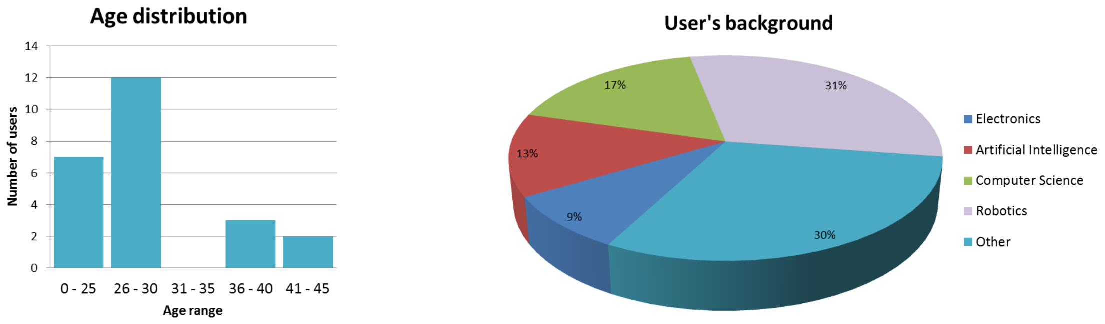
	Parameter selection			Test sequence		
Fold	Static gestures	Dynamic gestures	Mean	Static gestures	Dynamic gestures	Mean
0	0.703642	0.552158	0.627611	0.349593	0.636364	0.49298
1	0.641827	0.658219	0.650023	0.711538	0.000000	0.35577
2	0.713837	0.557703	0.635430	0.279476	0.603093	0.44128
3	0.71538	0.625359	0.670370	0.172078	0.186992	0.17954
4	0.640198	0.554127	0.597163	0.721311	0.624549	0.67293
5	0.667304	0.528129	0.597720	0.543605	0.77037	0.65699
6	0.595776	0.541330	0.568550	-[a]	0.620818	0.62082
Global	0.653063	0.564187	0.608625	0.462930	0.491744	0.48862

[a] Sequence 6 does not contain any static gesture.

defined per gesture in a problem dependent way. Moreover, the system allows easy addition of new gestures with its specified features. An implementation of a feature weighted Dynamic Time Warping algorithm has been applied to the dynamic gesture recognition.

A middle size humanoid robot, Aldebaran's NAO, has been used to interact with the user via both speech and gestures, and a wheeled platform, Nexter Robotics' Wifibot robot, is employed to ease NAO's navigation and sensor movement. NAO is able to ride Wifibot and to go down of it once they have approached a given goal. Both robots work in an independent way, and they are able to collaborate with each other in order to fulfil a task which right now includes, but it is not limited to, finding an object which has been referred by the user with a pointing gesture, with a speech based disambiguation using spatial or dimensional characteristics. Some extensions to this task could be adding more types of interactions or using the Wifibot to see the elements from other points of view.

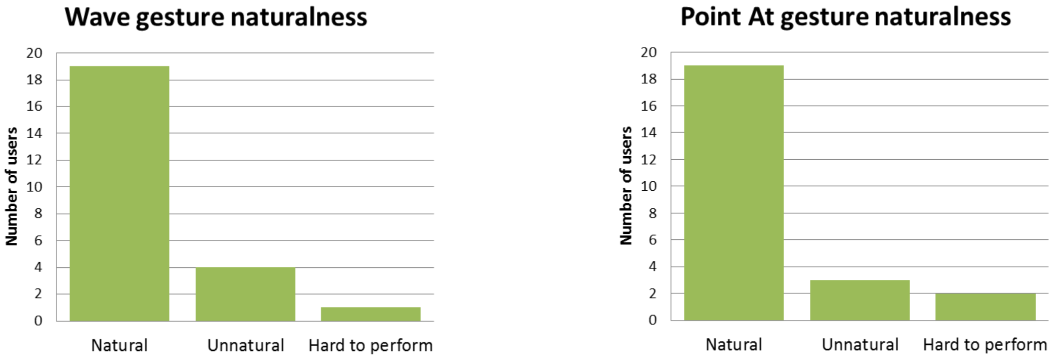
Furthermore, a series of tests with a varied set of users have been carried out, resulting in a good experience for them.



(a) Age groups.

(b) User's academic background.

Fig. 7: Demographic data of the users.



(a) Wave gesture naturalness.

(b) Pointing gesture naturalness.

Fig. 8: Users response to the naturalness of the gestures.

Most of the users thought the gestures were a natural way of interacting with the robot, and the response the robot had was the expected one and fast enough. This means that the system is easy to be used for human beings with minor indications on how to perform the gestures and, thus, the initial objectives of this project are considered as successfully accomplished.

The proposed system can be applied in household environments. Make robots bring something they pointed could be useful for elderly people or those with mobility difficulties. Many other gestures could be added in order to improve this interaction that rather than teleoperating the robot, intends to be a source of information to ease robotic task fulfilment, everything made in a natural, non forced way.

As future work some enhancement of the pointing at location estimation could be done, as the elbow-hand direction tends to point to further places, and also humans tend to point above the object. This is not a problem for us to distinguish the object, but it is a handicap for a robotic system. Some solutions to this issue may be the use of a learning method in order to adapt the gesture to the user, be it a general user or user specific, or a fixed factor could be applied to solve the major deviations. Also, other cues could be employed to improve the estimation of the pointing direction, such as the use of the gaze trajectory in other to adapt the arm one, as humans tend to look to the place they are pointing at.

Acknowledgment

We would like to thank the volunteer users who test the system, giving sincere feedback about their experience; Dr. Marta Díaz for her guidelines in the user test analyses, as well as Joan Guasch and Josep Maria Canal for their help in the Wifibot adaptations to carry the NAO robot and the Kinect^{TM} sensor. This work has been partially supported by the Spanish Ministry of Economy and Competitiveness, through the projects TIN 2012-38416-C03-01 and TIN2013-43478-P. The research fellow Gerard Canal thanks the funding through a grant issued by the Catalunya - La Pedrera Foundation.

References

[1] J. DeVito and M. Hecht, *The Nonverbal Communication Reader*. Waveland Press, 1990.

[2] C. Breazeal, C. Kidd, A. Thomaz, G. Hoffman, and M. Berlin, “Effects of nonverbal communication on efficiency and robustness in human-robot teamwork,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005. (IROS 2005)*, Aug 2005, pp. 708–713.

[3] G. I. Parisi, D. Jirak, and S. Wermter, “Handsom - neural clustering of hand motion for gesture recognition in real time,” in *Proceedings of the 2014 RO-MAN: The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, August 2014, pp. 981–986.

[4] F. Bettens and T. Todoroff, “Real-time DTW-based gesture recognition external object for Max/MSP and Puredata,” in *Proceedings of the Sound and Music Computing conference (SMC ’09)*, 2009, pp. 30–35.

[5] A. Hernández-Vela, M. A. Bautista, X. Pérez-Sala, V. Ponce-López, S. Escalera, X. Baró, O. Pujol, and C. Angulo, “Probability-based Dynamic Time Warping and Bag-of-Visual-and-Depth-Words for Human Gesture Recognition in RGB-D,” *Pattern Recognition Letters*, vol. 50, no. 0, pp. 112–121, 2014, depth Image Analysis.

[6] M. Reyes, G. Domínguez, and S. Escalera, “Feature weighting in Dynamic Time Warping for gesture recognition in depth data,” in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, November 2011, pp. 1182–1188.

[7] T. Arici, S. Celebi, A. S. Aydin, and T. T. Temiz, “Robust gesture recognition using feature pre-processing and weighted dynamic time warping,” *Multimedia Tools and Applications*, vol. 72, no. 3, pp. 3045–3062, 2014.

[8] 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 *Proceedings of the 6th International Conference on Human-robot Interaction*, ser. HRI ’11. New York, NY, USA: ACM, 2011, pp. 229–230.

[9] T. Fujii, J. Hoon Lee, and S. Okamoto, “Gesture recognition system for human-robot interaction and its application to robotic service task,” in *Proceedings of The International MultiConference of Engineers and Computer Scientists (IMECS 2014)*, vol. I, International Association of Engineers. Newswood Limited, 2014, pp. 63–68.

[10] P. Barros, G. I. Parisi, D. Jirak, and S. Wermter, “Real-time gesture recognition using a humanoid robot with a deepneural architecture,” in *Proceedings of the IEEE-RAS International Conference on Humanoid Robots (Humanoids ’14)*. IEEE, 2014, pp. 83–88.

[11] E. Bernier, R. Chellali, and I. M. Thouvenin, “Human gesture segmentation based on change point model for efficient gesture interface,” in *Proceedings of the 2013 IEEE RO-MAN*, Aug 2013, pp. 258–263.

[12] X. Zhao, A. M. Naguib, and S. Lee, “Kinect based calling gesture recognition for taking order service of elderly care robot,” in *The 23rd IEEE International Symposium on Robot and Human Interactive Communication, 2014 RO-MAN*, Aug 2014, pp. 525–530.

[13] C. Park and S. Lee, “Real-time 3d pointing gesture recognition for mobile robots with cascade {HMM} and particle filter,” *Image and Vision Computing*, vol. 29, no. 1, pp. 51 – 63, 2011.

[14] K. Nickel and R. Stiefelhagen, “Visual recognition of pointing gestures for humanrobot interaction,” *Image and Vision Computing*, vol. 25, no. 12, pp. 1875 – 1884, 2007, the age of human computer interaction.

[15] M. Van den Bergh, D. Carton, R. de Nijs, N. Mitsou, C. Landsiedel, K. Khnlenz, D. Wollherr, L. J. Van Gool, and M. Buss, “Real-time 3d hand gesture interaction with a robot for understanding directions from humans,” in *Proceedings of the 2011 IEEE RO-MAN*, H. I. Christensen, Ed. IEEE, 2011, pp. 357–362.

[16] R. Luo, S. Chang, and Y. Yang, “Tracking with pointing gesture recognition for human-robot interaction,” in *System Integration (SII), 2011 IEEE/SICE International Symposium on*, December 2011, pp. 1220–1225.

[17] C. Matuszek, L. Bo, L. Zettlemoyer, and D. Fox, “Learning from unscripted deictic gesture and language for human-robot interactions,” in *Proceedings of the 28th National Conference on Artificial Intelligence (AAAI)*, Québec City, Quebec, Canada, March 2014.

[18] J. L. Raheja, A. Chaudhary, and S. Maheshwari, “Hand gesture pointing location detection,” *Optik - International Journal for Light and Electron Optics*, vol. 125, no. 3, pp. 993 – 996, 2014.

[19] D. Droeschel, J. Stuckler, and S. Behnke, “Learning to interpret pointing gestures with a time-of-flight camera,” in *Proceedings of the 2011 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, March 2011, pp. 481–488.

[20] M. Pateraki, H. Baltzakis, and P. Trahanias, “Visual estimation of pointed targets for robot guidance via fusion of face pose and hand orientation,” *Computer Vision and Image Understanding*, vol. 120, no. 0, pp. 1 – 13, 2014.

[21] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, “Real-time human pose recognition in parts from single depth images,” in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR ’11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1297–1304.

[22] H. Sakoe and S. Chiba, “Dynamic programming algorithm optimization for spoken word recognition,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 26, no. 1, pp. 43–49, February 1978.

[23] R. B. Rusu, “Clustering and segmentation,” in *Semantic 3D Object Maps for Everyday Robot Manipulation*, ser. Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg, 2013, vol. 85, ch. 6, pp. 75–85.

[24] K. Ogata, *Modern Control Engineering*, ser. Instrumentation and controls series. Prentice Hall, 2010.