Detection of interaction intention with gaze and gesture analysis

Extended Abstract

João Salvado, Alexandre Bernardino, José Santos-Victor
Institute for Systems and Robotics (ISR/IST), LARSyS, Instituto Superior Tecnico,Univ Lisboa
Lisboa, Portugal
joao.m.salvado@ist.utl.pt,alexandre.bernardino@tecnico.ulisboa.pt,jasv@isr.tecnico.ulisboa.pt

ABSTRACT

This paper proposes a method to give a robots the ability to detect if humans users are willing to engage in interaction. We denote this skill *Interaction Intention*. It is possible to detect interaction intention by using nonverbal cues, verbal cues or even both. In the context of this work, it was studied the use of nonverbal communication to detect the interaction intent. The interaction intention detector can be used to help a robot choosing an interlocutor in a group of people. Our approach is based on a multimodal system that takes into account the gaze direction and gestures performed by a user. Gaze and gestures combined can be used to understand if a person wants to interact or not. However, it is necessary to synchronize and fuse them into a robust, fast and accurate detector. We trained and evaluated or detector from experiments with a total of 71 participants. The detector of interaction intent was capable of detecting if a person wants to interact with the robot or not with an accuracy of 91.1%.

KEYWORDS

interaction intention, gesture detection, gaze detection

1 INTRODUCTION

Social Robots are growing their presence among humans in several scenarios, so it is important to provide them with appropriate interaction skills. Before an interaction starts, it is fundamental for the robot to understand if a possible human interlocutor is willing to engage in an interaction. Thus it must be able to detect an *Interaction Intention*. Interaction Intention and Intention Recognition can be defined as inferring an agent's intention through its actions and their effects in the environment [12] Detection of user's intention is becoming more and more important and has gained attention in Human-Robot Interaction [17]. It is important that a robot has the capability to know when people want to interact with him and adapt to the situation [3]. Breazeal et al. [5] states that the use of nonverbal behavior in coordinating joint activity plays a very significant role to understand human's intention but is yet understudied compared with verbal behavior. In particular gestures (like waving) and eye gaze are reliable indicators of nonverbal interaction [8]. Gaze cues are an especially important set of information that helps to understand mental states and intentions [20]. Hansen and Ji [10] adds that the interpretation of gaze is a valuable tool, and besides the usual techniques for gaze detection it would be good to explore and research the development of applications that exploit a combination of gaze with gestures. According to [21] if a gesture is used interactively or as a way of communication, it is important to recognize whether the gesture is directed toward the current interaction partner or not.

In this work we combine gaze and gestures to develop a nonverbal detector of interaction intention. This skill is extremely important in human-robot interaction because a robot must be capable of deciding whether to pay attention or ignore a user's actions[24]. The developed system was implemented in a real robot that observes people on its view field and detects if they are willing to interact with it by analyzing the person's gaze direction and gestures. If a person is performing a relevant gesture for interaction (e.g hand waving) but is not looking at the robot, then probably his intention is not to interact with the robot. Likewise, a person may be looking at the robot but doing an irrelevant gesture for interaction (or no gesture at all), so, again, the robot should not assume any interaction intent. It is the combination of both interaction cues, gaze direction and relevant gestures, that should trigger the robot interaction behavior. We have performed several studies with subjects to understand which gestures are relevant for interaction intent, how accurate should be the gaze directed towards the robot, and the timings and synchronization aspects of the interaction that lead to a robust detector. We have used state of the art methods for gaze detection from video and gesture detection from Kinect V2 to implement a fully automated intent detector running in real time in a robotic platform.

The paper is organized as follows, first we present related work about interaction intent, then the system overview, after that the gesture detector and the gaze detector, then interaction intent detector. Finally it is presented the implementation aspects the experiences performed and the results obtained.

2 RELATED WORK

Most of the existing solutions to detect the intent of interaction are based on human position, speed and the distance between human and robot [13], [16]. Michalowski et al. [16] stated that a model of engagement based on stationary positions may be insufficient for increasing the frequency and quality of interactions, and they propose as future work that it would be important to use a solution that takes into account gaze and gestures to obtain information about the engagement and interaction intent.

Some methods fuse multimodal cues like the head pose, shoulder orientation, and vocal activity detection[17]. There are also some solutions implemented in a robot that take into account the spatial information, body pose, frontal face detection, speech detection, and sound localization to detect the engagement, however, neither of them take the gaze into account [3].

Another solution use gestures to detect interaction intent of the user in human-robot interaction [21], however, this solution does not take into account if the user is looking to the robot or not to decide if the person wants to interact with the robot or not. Some

of the solutions that use the gaze detection to decide if the person wants to interact are [11] and [1].

A solution that combines facial features, body pose, gesture, and gaze to detect interaction intent is proposed in [24]. However this solution is not implemented in a real robot and the author states that the solution is not fastest and most accurate.

Most of the solutions are not capable of tracking and detecting interaction intent of multiple persons at the same time in a real time environment. It is important to understand how to use gaze and gestures together and which gestures are important to take into account.

3 SYSTEM OVERVIEW

The developed interaction intent detector is composed of two detectors: a gesture detector and a gaze detector module. That it also another module called matching module that combines the information from the gaze detector and the gesture detector and outputs a concatenated list of all people detected. The results of matching module are used for a system that tracks the persons in the environment over time and filters the results of the interaction intent.

4 GAZE AND GESTURE DETECTOR

4.1 Gaze detector

In the context of this work, we will use a video based gaze detector to perform gaze direction detection and solve the problem of understanding if a person is looking to the robot or not. This problem is called gaze looking [25]. In [25] it is presented a solution that is capable of perceiving if the user is looking or not at a vision based interface. However the authors stated that their approach is prone to multiple errors like illumination or eyes occluded.

OpenFace is a solution presented by Baltrusaitis et al. [2]. With their method it is possible to extract the head pose (rotation and translation) and a gaze vector from a color image. These features can be used to feed a classifier and detect if one person is looking to robot or not, we present two solutions to solve this problem.

4.1.1 Gaze looking using head pose - solution one. In this modality we use only the information regarding the head pose with respect to the camera and not the gaze direction. This mode is particularly useful when the person is far(more than one meter) from the robot, where the gaze direction returned by the software is not reliable, but the head pose is still accurate.

The head pose detector returns the person's head pose in the camera reference frame as an axis-angle representation. In the camera reference frame the Y-axis points downwards, the Z-axis points forward, and X-axis points to the left. If the person is exactly in front of the camera, then the head pose angle is zero when the person's head if directed towards the robot. However, if the person is not directly in front, then the angle is not zero even if the person is looking at the robot. So, there is a need to work out these angles based on the location of the person in space. The main idea consists in rotating the optical axis of the camera towards the location of the head of the person, so that the measured angles are referred to that newn the camera reference frame after the rotation. After this rotation, the camera's Z axis points to the person and so, if the person is looking to the robot, the rotation angles should be close to zero.

The rotations that are done to make the camera Z axis point to person's face are based on the person location, the location of the individual (x_f, y_f, z_f) extracted from OpenFace [2] is given in 3D coordinates in the camera referential.

The technique used corresponds to do virtual tilt and pan rotations [26]. The steps are the following:

(1) The first step is calculate the rotation angle p ,the angle p corresponds to:

$$p = \arctan\left(\frac{x\_f}{z\_f}\right) \tag{1}$$

(2) After having the angle p it is necessary to calculate the rotation matrix associated to that angle, as the first rotation is about y axis. So the rotation matrix for the angle p will be:

$$R_1 = \begin{bmatrix} \cos p & 0 & \sin p \\ 0 & 1 & 0 \\ -\sin p & 0 & \cos p \end{bmatrix}$$

(3) Then we rotate an angle t about the new axis x' in order to have the z' axis pointing to persons face. By taking into account that the new axis $z' = \frac{z\_f}{\cos p}$, the t is given by:

$$t = \arctan\left(\frac{y\_f}{z'}\right) = \arctan\left(\frac{y\_f}{z\_f}\cos p\right) \tag{2}$$

(4) Now we calculate the rotation matrix associated with this angle. So the rotation matrix for the angle t will be:

$$R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos t & -\sin t \\ 0 & \sin t & \cos t \end{bmatrix}$$

(5) Now we convert the rotation vector that is given as output from the head pose detector to a rotation matrix. This matrix will be denoted as R_{ori}

(6) The final matrix will be given by the successive multiplication of these matrices :

$$R_f = R_1 R_2 R_{\text{ori}} \tag{3}$$

(7) Finally this rotation matrix is converted again to a rotation vector. This vector will be named h_p.

After following the steps explained above we have obtained values of head pose that can be used to understand if the person is looking at the camera or not. The z orientation was not considered.

4.1.2 Gaze looking using eye gaze - solution two. . In the solution we use a combination of head pose and gaze vector, the gaze vector is taken into account when the person is close to the robot. The gaze estimator returns two gaze vectors One per eye, they represent where the person is looking at in the camera reference frame they are Both used to predict where the person is looking at. To process these vectors it was necessary to perform the following:

(1) Normalize each vector to unitary norm

(2) Calculate the correspondent angle for each eye (left and right), this can be done by using arccos for each component of each normalized vector. So for each eye vector:

$$a\_x = \arccos(x\_n) \tag{4}$$

$$a\_y = \arccos(y\_n) \tag{5}$$

$$a\_z = \arccos(z\_n) \tag{6}$$

After this step we end up with two vectors the vector of the right eye (a_x_r, a_y_r, a_z_r) and the vector of the left eye (a_x_l, a_y_l, a_z_l)

(3) Calculate the F_v which is the average between left eye gaze vector components and right eye gaze vector components.

(4) Now we calculate the i_v, the i_v represents a vector of the person looking to the robot at a certain location. Then the v_p is the person head pose location and it is used to calculate the i_v which is a vector that points from that point in the space (represented by v_p) to the origin of camera reference. The i_v it is calculated by using the following formulas:

$$i\_v_x = \arccos(-v\_p_x) \tag{7}$$

$$i\_v_y = \arccos(-v\_p_y) \tag{8}$$

$$i\_v_z = \arccos(-v\_p_z) \tag{9}$$

The negative sign in the equations is due to the fact that it is desirable to have a vector pointing from the person's face to the camera and not the opposite.

(5) Now it is possible to calculate the difference between the i_v and F_v which is defined as d_g ideally if the person is looking to the camera the difference between them should be close to zero. The d_g is calculated according to this:

$$d\_g_x = i\_v_x - F\_v_x \tag{10}$$

$$d\_g_y = i\_v_y - F\_v_y \tag{11}$$

$$d\_g_z = i\_v_z - F\_v_z \tag{12}$$

As output of the gaze detection module there will be a list of persons O_e and for each person there will be information about the head pose (translation and rotation vector) , the location 2D of the face(for future processing) and the corrected orientation vector h_p and finally the processed gaze vector d_g.

4.2 Gesture detector

For gesture detection, it is important to choose a detector algorithm and what gestures should be considered. In [22], [4], [28], [14], [9] are proposed methods that use the Kinect camera. Le et al. [14] stated that Kinect detects with high accuracy the performed tests.

In this project, it will be used a similar using Kinect the software available (SDK/Kinect studio) and Visual Gesture Builder (VGB). The reviewed papers showed that Kinect gives good results, can detect multiple persons. It is also reported in articles that Kinect can detect with high accuracy.

According to the authors [21], the gestures can be classified into five major types: "Irrelevant"/Manipulative Gestures, Side Effect of Expressive Behavior , Symbolic Gestures, Interactional Gestures and Referential/Pointing Gestures. In the context of this work it was considered to detect Interactional gestures as these are the types of gestures used to start an interaction. Initially, it was proposed the following six Interactional gestures: Handshake with right hand, Handshake with left hand , Hand wave with the right hand, Hand wave with the left hand, Boxing gesture(two hands raised) and Thumbs up. The acronyms for these gestures are the following:,HS_R, HS_L, HW_R, HW_L, BX, T_U respectively, the acronym for no gesture is O, and the acronym for no gesture is N_G.

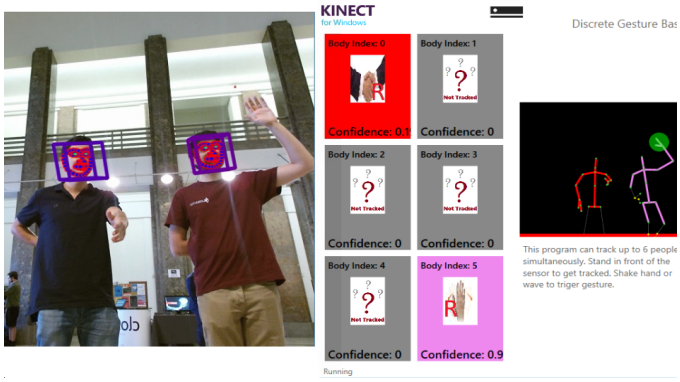

Figure 1: Hand shake gesture with right hand detected and Hand wave with right hand detected.

The output of the gesture detection module consists of a list of persons O_g, and, for each person, information about the head location (translation vector) and the gesture that each person detected is performing. Each gesture has an id associated, when there is no gesture the id associated is zero.

In figure 1 it can be seen multiple gestures being detected. On the left, it is possible to see the detection of the face from the OpenFace and on the right is possible to see the detection of two different gestures handshake right and hand wave right.

5 INTERACTION INTENT DETECTOR

5.1 Matching

In this module, it is necessary to concatenate the results from gaze detection(OpenFace) O_e with results from gesture detection(Kinect) O_g. The goal of this module is to know if a person is performing a gesture and if that person is looking to the robot. The O_g list already gives a list of persons with the head position and the gesture that the person is performing g_v. The O_e only returns raw features that should be used to feed a trained classifier that would output if the person is looking to the robot or not. After this, we have a list of persons e_v with an associated head position and an associated looking or not looking value. At each moment t if O_e and O_g are not empty, we proceed to the matching step.

The head position in e_v is represented in the camera reference frame where the Y-axis points downwards the X-axis points to the right, and Z-axis points towards while the head position g_v is in a referential where the Y-axis points upwards the X-axis points to the left and Z-axis points towards. The results of head location in the gaze module are expressed in millimeters while in gesture detector are in meters. So before doing the matching it is necessary to correct the units:

$$\begin{bmatrix} e\_v_x \\ e\_v_y \\ e\_v_z \end{bmatrix} = \frac{1}{1000} \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e\_v_x \\ e\_v_y \\ e\_v_z \end{bmatrix} \tag{13}$$

The matching process receives as input the e_v and the g_v and returns as output a r_v. The matching function used is the

Hungarian algorithm [19], where the cost function used is euclidean distance between each element of the e_v and each element of the g_v according to the formula:

$$d_{ij} = ||e\_v_i - g\_v_j|| \tag{14}$$

To do the attribution it is necessary that the euclidean distance is smaller than a defined threshold equals to one meter, otherwise it would be empty so the following rule as to be true:

$$\text{For each } i : r\_v_i = argmin\_j\ d_{ij}, \text{if } d_{ij} < T; r\_v_i = [] \quad o/w \tag{15}$$

After the assignment is performed the r_v will be a list of persons, and for each person, it will have information about the head position, the gesture being performed and if the individual is looking to the robot or not.

5.2 Interaction intent filtering and tracking

This module keeps the identity of the persons coherent among the frames. This module receives as input the r_v which is the output of interaction intent detector module.

The tracking is performed by doing the association of the persons between frames using the Hungarian algorithm [19], where the cost function used is the Euclidean distance between each person Persons already in the tracking system and each new person in r_v The fields of information of the list of Persons in tracking system are similar to the ones of the r_v but it also has an interaction_intent field which represents if the person wants to interact or not.

The interaction intent is calculated by the median of a buffer vector; the buffer vector has information of the interaction_intent in last N frames. This buffer vector is a circular buffer that initially is filled with zeros. In a circular buffer the oldest element in the vector is overwritten by the newest element, a methodology called first in first out. The length of the buffer may vary. If we want a faster detection but less precise, we would have to choose a lower N. If we want a slower detection but more precise we would choose a large value.

The d_i represents the interaction_intent at each frame and is calculated according to the formula:

$$d\_i = p\_l * g \tag{16}$$

where p_l is a binary value that represents if the person is looking or not to the robot and the g represents the gestures performed. So if the person is not looking the value of p_l would be zero. If the person is not making a gesture the value of g would be zero. If the person is looking and making a gesture, then the value of the d_i would be equal to the value represented by the gesture that is being done, which varies between one and six.

6 IMPLEMENTATION

The robot used in this project is Vizzy. To implement Vizzy's skills Moreno et al. used two different middlewares: Yet Another Robot Platform (YARP)[15] and Robot Operating System (ROS)[23].

The implemented system makes uses of two different operating systems, Windows and a Linux distribution, Ubuntu. By doing this, the processes could be distributed. The Windows computer is used to perform the gesture detection with the Kinect camera, The results of gesture detection are written to a NO SQL database (mongoDB[7]) which is a good option for real-time system. The windows computer is also used to perform gaze detection from the Kinect camera. The interaction intent detector module runs on the Linux distribution and combines the information from both detectors. This module outputs the results via ROS topic so that the tracking and the filtering could be performed.

Table 1: Number of occurrences per gesture using acronyms defined in section:gest

Gesture	HS_R	HS_L	HW_R	HW_L	T_U	BX	O
Ocurrences	8	0	20	2	1	0	0

7 DATA COLLECTION

The objective of the data collection session was to acquire data to train the gesture detector and the gaze detector. Thirty-one persons were participating in this data collection session; the persons were from different cultures with different ages, gender, backgrounds, and education.

The data collection session is subdivided into three different experiences; the first experience is a questionnaire to understand which gesture the persons use to start an interaction with the robot, the second experience was to record data to train the gesture detector the third experience was to record data to train the gaze detector.

7.1 Experience 1.1 - Choose the gestures to use

The aim of this experience was to understand if the gestures proposed in this work match the gestures proposed by the persons. If it is proposed a new gesture by the participants, then this new gesture should be included in the work.This experience would help to understand what is the best gesture to start an interaction with the robot.

It was asked to each participant to start an interaction performing a single gesture that they would use to start an interaction. The gestures initially considered in this work were not told to participants. So they had absolute freedom to choose the gesture to start an interaction. The participant was alone at around one meter from the robot; the robot was stopped.

With this experience, it was possible to verify that the gestures chosen by the persons to start an interaction, are covered by the gesture set initially. As it is possible to see in the table 1 the handshake right, the handwave right, the handwave left and the Thumbs up were chosen at least once to start an interaction, and the gesture with the higher number of occurrences is the handwave right with 64.5%.

Thus, the Boxing gesture and the handshake left were not chosen as the first gesture to start an interaction even tough there were left handed people in this experience. Nevertheless, they were used in the system for future work. As no one had chosen a gesture different from the proposed gestures it was not considered new gestures in the gesture detector.

7.2 Experience 1.2 - Record data for gaze detector

The objective of this experience was to acquire data to train the gaze classifier. It was asked to each person to look at the robot. Then it was asked to the person to do not look at Vizzy. The results were saved and tagged with the boolean value one and zero respectively.

In this experiment the person was alone at around one meter from the robot, the robot was stopped and looking to the persons. The person could move the head and the eyes during the experiments so we could get results from different perspectives and situations.

Each test had the duration of approximately two minutes per person, one minute of the test looking and one minute not looking to the robot. In this experience were saved the features: head pose h_p, and difference gaze vector d_g.

7.3 Experience 1.3 - Record data for gesture detector

The objective of this experience was to acquire data to train the gesture classifier. It was asked to every individual to perform all the proposed gesture. Seven videos of thirty seconds each were recorded per person. It was recorded a video of the individual making the following gestures to the robot: handshake right, handshake left, handwave right, handwave left, thumbs up and boxing gesture. It was also recorded a video where the person was not making any of those gestures.

8 TRAINING THE CLASSIFIERS

In this section it will be explained the procedures to train the gesture detector and the gaze detector and the methods to evaluate them.

8.1 Gaze Detector training

Several detector types were compared for gaze detector: K-Nearest Neighbors(KNN), Support vector machine(SVM), Random forest(RF), decision tree(DT), neural net(NN), Naive Bayes(NB), AdaBoost(AB) and Naive Bayes. All of them were trained and tested using the best parameters that were found by performing cross-validation on the training data set. This procedure was repeated sixteen times. The training data and testing data were randomly chosen in all the sixteen runs.

The accuracy metric was used to evaluate the performance of the classifier. The number of positive samples was 7658 while the number of negative samples was 7902.

The following steps were performed to choose the best classifier. First it is initialized a frequency table with all occurrences equals to zero. The table represents the number of occurrences that a machine learning algorithm had better than the others. It is chosen randomly 80% of the data for training and 20% of the data for the test and the data is standardized. Now for each machine learning algorithm choose a range of parameters and for each parameter perform cross-validation on the Training set. The training set is divided into ten folds to perform cross-validation. For each machine learning algorithm choose the parameters that got the best Cross validation result, which is the mean accuracy of all folds. Train each machine learning algorithm using the best parameters from the cross-validation and test the trained classifiers on the Test set. Finally increment the frequency of occurrences in the frequency table for the classifier that got the best accuracy. Repeat this procedure until sixteen runs have been done. Choose the best classifier by looking to the average accuracy and the table of occurrences. The process described above was repeated using both feature d_g and h_p.

Table 2: Number of occurrences that each classifier got better accuracy than others using h_p and d_g

Classifier	KNN	SVM	DT	RF	NN	AB	NB
Ocurrences h_p	8	6	0	2	0	0	0
Ocurrences d_g	9	7	0	0	0	0	0

Table 3: Results of average accuracy using first the feature h_p and then the feature d_g

Classifier	KNN	SVM	DT	RF	NN	AB	NB
h_p(%)	94.45	94.42	94.04	94.30	93.94	93.03	93.59
d_g(%)	93.30	93.21	92.31	92.91	92.40	90.31	90.93

For cross-validation, it was considered range of k between 1 and 10; for the support vector machine algorithm a range of C between 0.01 and 100 and a range of gamma between 0.01 and 100; for the decision tree algorithm, a range of maximum depth between 1 and 10; for the Random Forest algorithm a range of maximum depth between 1 and 10 and a range of a number of estimators between 1 and 10; for the neural network algorithm a range of alpha between 10^{-5} and 10^{5} when cross-validation was performed; for the Adaboost algorithm network algorithm a range of a number of estimators between 1 and 10.

In the table 2 it is possible to see the number of occurrences that each classifier got better than the others by using the different features. In the table 3 it is possible to see the average accuracy for each classifier using the different features.

From this tables it is possible to see that Nearest neighbors had the higher number of occurrences (8 using h_p and 9 using d_g) and had the best accuracy (94.45% using h_p and 93.30% using d_g).

So from the tables 2,3 it was concluded that the nearest neighbors were the best classifier for this problem. The results are really similar by using one feature or another but as the average accuracy of the nearest neighbors when using the feature h_p was 94.45% and the average accuracy for the feature d_g was 93.30% the choice was to use the feature h_p for the final classifier.

It was then necessary to choose the number of neighbors k. To choose the k, it was performed cross-validation considering a range of k between one and twenty. The data was divided into ten folds, and the accuracy per k is the average accuracy of all folds. The best cross-validated accuracy was 92.2% achieved when the k was equals to eighteen. The final classifier was the nearest neighbor using $k = 18$ and using the feature h_p.

8.2 Gesture Detector training

The videos recorded in experience 1.2 were used to train the gesture detector. We used six different gestures: boxing, thumbs up,

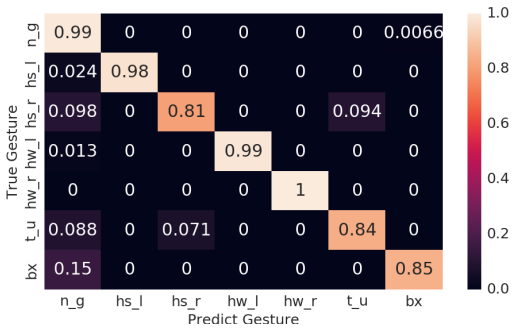

Figure 2: Confusion matrix of gesture detector normalized

handshake right, handshake left, hand wave right and hand wave left. Each gesture was trained considering all the videos of the persons performing that gesture, and fifty-five random videos of the persons performing the other gestures. It was ignored the lower body information, and it was used the hand's data information

To test the gesture classifier, it was performed a new experiment with ten persons; these ten persons were different from the ones used to train the gesture classifier. In this new test, it was asked to each person to perform each of the gestures to the robot during one minute; and to do not perform any gesture to the robot during one minute. In the end, each person performed six gestures (handshake right, handshake left, handwave right, handwave left, thumbs up and boxing gesture) and one no gesture, each during one minute.

In figure 2 it is possible to see the confusion matrix of the gesture detector.

By looking to figure 2, it is possible to verify that the matrix is close to an identity matrix. The prediction of the hand wave right gesture was perfect. This is important as this gesture had the highest number of occurrences to start an interaction. The hand wave right, hand wave left, the no gesture and the handshake left were almost perfectly predicted. The handshake right, the thumbs up and boxing gesture were not so well predicted but still achieved good results.

From table 4 it can be concluded that the no interaction gestures were recognized in 99.34% of the situations and predicted as an interaction gesture in 0.66% of the cases. So the gesture detector recognized with high accuracy a no interaction gesture. On the other hand, the interaction gestures were recognized correctly in 93.8% of the situations and predicted as no interaction gesture in 6.2% of the situations.

Table 4: Confusion matrix of interaction and no interaction gestures

True \ Predicted	no interact gesture	interact gesture
no interact gesture	99%	0.66%
interact gesture	6.2%	94%

9 FINAL END-USER VALIDATION

The objective of this experiment was to test the interaction intent detector developed and the final results. In this experiment there were thirty participants, different from the other forty one used in experience one. Were performed fourteen experiences, each one corresponding to a different situation, in each experience, it was recorded the RGB data, the output of the gesture classifier, the output of the gaze classifier and the head pose. Each situation was recorded during thirty seconds which corresponds to a total of seven minutes of data per person.

In the first six situations, the person tried to interact with the robot using the six gestures proposed, one at the time. All these situations were tagged with the Boolean value one that means that the persons wanted to interact with the robot. Then all these gestures were performed again but this time without looking at the robot. Finally, it was recorded a situation where the person was looking at the robot but not performing any gesture and a situation in which the person did nothing. All these last eight cases were tagged with the Boolean value zero meaning that the individual did not want to interact with the robot. In the total 14 experiences were made per person.

To evaluate the performance of the method were tested different different buffer lengths. For each frame and each length of the buffer if the number of zeros in that buffer is in strict majority then the output from the filter is zero. Otherwise, the output will be the number that occurs often. It was explored the response of different buffers from a range of length between one and ninety-nine.

9.1 Analysis of experience 2

In this experience, there were 12025 positive labeled frames and 11766 negative labeled frames. It was analyzed the number of True positives, True negatives, False negatives and False positives.

In the figure 3 it is possible to see the overall number of True positives, True negatives, False positives and false negatives detected per buffer size. The results show as expected the number true positives and False Positives decrease as the size of the buffer increases. This is because the buffer was initialized with zeros so it would take longer for a bigger buffer to initiate detections so it would detect less True Positives and less False Positives. For the same reason, the number of True negatives and the number of False negatives increases as the size of the buffer increases.

In general, a buffer with a small size would catch faster, would detect more true positives and would detect more false positives, but it would detect less true negatives and less false negatives. Also in general a buffer with a bigger size would catch slower, would detect less true positives, less false positives but it would detect more true negatives and more false negatives.

9.2 Error Analysis

When a false positive is detected there are three possible reasons. First, the person is, in fact, doing a gesture but not looking to the robot which reflects a failure in the gaze detector.

Second, the person is looking to the robot but is not performing any gesture which results from a failure in the gesture detector. Finally, if the person is not looking to the robot and not performing then there is a failure of both the gaze detector and gesture detector.

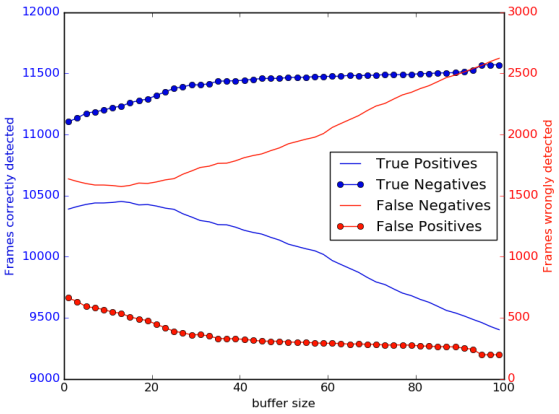

Figure 3: Overall Correctly frames detected

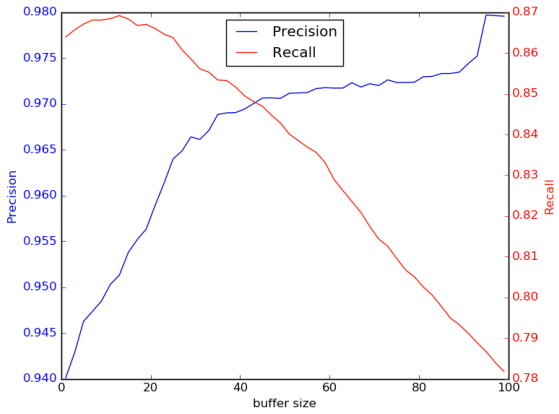

Figure 5: Overall precision and Overall recall

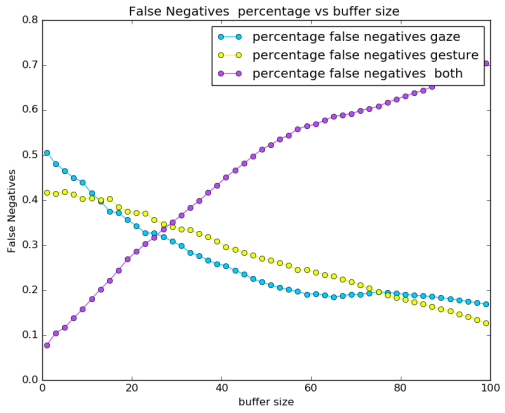

Figure 4: Distribution of false negatives

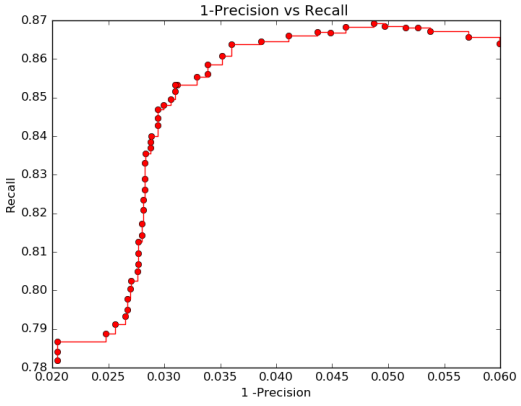

Figure 6: Overall 1-precision vs recall

In this experience, 0.0097053% of the times a false positive occurs is due to a failure of gesture detector, and 99.9902947% is due to a failure of the gaze detector.

In table 4 it is possible to see that the probability that an interaction gesture is detected when the person is not performing a gesture is 0.66%. This helps to explain why the false positives are due to failures of the gaze detector and not of the gesture detector.

When a false negative is detected it can be because the gaze detector fails or the gesture detector fails to detect, or it can be because both gaze and gesture detector fail.

In figure 4 it is possible to see the distribution of false negatives. A false negative occurred due to a failure with the gaze detector in average 26.31% of the situations and occurred due to a failure with the gesture detector in average 27.60% of the situations and finally occur due to a failure of both in 46.09% of the situations.

The fact that, for bigger sizes of buffer, the false negatives are detected as a failure of both is because all the buffers are initialized with the value zero, and that corresponds to a failure of gaze and gesture.

9.3 Evaluation

To understand the behavior of the interaction intent detector when the buffer size changes it was calculated the precision and recall curves. The precision-recall curve is helpful to understand the trade-off between precision and recall for different thresholds or in this case, size of the buffer.

In figure 5 it is possible to see the overall precision and recall per buffer size. The precision increases as the buffer size increases because the number of false positives decreases as the buffer size increases. The recall decreases as the buffer size increases because the number of false negatives increases as the buffer size increases.

Figure 9 shows the overall 1-precision vs. recall curve per buffer size. We can have a system with a small buffer that has a fast response and a high recall but with a low precision, or we can have a system with a big buffer that has a slow response and low recall but with a high precision.

Now it is important to evaluate what is the best value for the buffer size. According to [27] the F_1 score is a harmonic mean of precision and recall and is a good way to measure the trade-off between precision and recall.

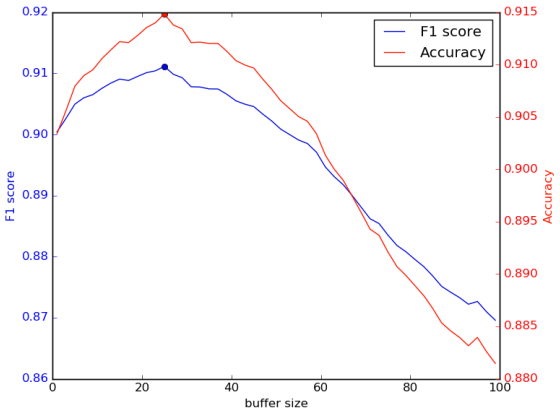

Figure 7: Overall Accuracy and F_1 score

Figure 8: Detection of interaction intent - hand wave left

In figure 7 we can see the overall F_1 score and the overall accuracy per buffer size. The length of the buffer that maximizes the overall F_1 score is 25 with a value of 0.9148. Also, the length of the buffer that maximizes the overall accuracy is again 25 with a value of 91.111%.. Recording that the system runs at 16 frames per second, this means that the system would take about 1.6 seconds to detect an interaction intent.

Figure 8 shows the output of the implemented system when detection of interaction intent occur Figure 9 shows the output when detection of interaction intent does not occur. In figure 8 The green thumbs up means that the person wants to interact, the eye in the middle of the image indicates that the person is looking to the robot and the Hand with an L on the right means that the person is performing an Hand Wave left. In figure 9 The red thumbs up means that the person does not want to interact, the eye in the middle with a cross indicates that the person is not looking to the robot and the Hand with an R on the right means that the person is performing an Hand Wave right, the gesture is not towards to the robot.

10 CONCLUSIONS AND FUTURE WORK

With this work, it was possible to develop an interaction intent detector that is capable of working in a real-time environment.

Figure 9: No detection of interaction intent when using hand wave right

The developed detector of interaction intent is fast(16 frames per second) and robust, showing an accuracy of 91.111% and an F1 score of 91.48% for a buffer size equals to 25 which represents 1.6 seconds to detect the interaction. The false positives were mostly due to failures with gaze detector while the false negatives were due to failures of gaze detector and gesture detector almost in the same percentage.

The environment where the tests were performed was controlled as well as the experiments. The environment and the assumptions made do not allow to have an 100% natural detector of interaction intent and it would be interesting to explore how to detect the interaction intent in a non controlled environment.

It would be interesting to explore other possibilities to do the interaction intent detector, for example using a solution like [6] ,the role of emotions and a interaction intent detector that uses verbal communication.

It was possible to confirm that the hand wave right is the gesture that was chosen as the most used gesture for interaction intent. This was the gesture with best detection rate in our system. The failures of the gesture detector where essentially due to a bad detection of handshake right, thumbs up and boxing gesture. It would be interesting to acquire more data of those gestures and train the system again to have a better detector.

The developed gaze detector was capable of understanding when the person was looking to the robot or not,using the k-nearest neighbors algorithm, achieving an accuracy of 92.2% when k was equals to 18. It would be interesting to explore other approaches like for example using a camera that has an analog zoom system incorporated that would allow to zoom in the face of the persons and extract features of the gaze vector with more precision.

The developed system was implemented in a real robot, and it was used and tested in different science fairs showing the importance of having an interaction intent detector with filtering and a tracking system associated.

REFERENCES

[1] Stylianos Asteriadis, Kostas Karpouzis, and Stefanos Kollias. 2009. Feature extraction and selection for inferring user engagement in an hci environment. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 5610 LNCS. 22–29.

[2] Tadas Baltrusaitis, Peter Robinson, and Louis Philippe Morency. 2016. OpenFace: An open source facial behavior analysis toolkit. In *2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016*.

[3] Wafa Benkaouar and Dominique Vaufreydaz. 2012. Multi-Sensors Engagement Detection with a Robot Companion in a Home Environment Multi-Sensors Engagement Detection with a Robot Companion in a Home Environment. Workshop on Assistance and Service robotics in a human environment at. (2012), 45–52.

[4] K K Biswas and S K Basu. [n. d.]. ([n. d.]).

[5] Cynthia Breazeal, Cory D. Kidd, Andrea Lockerd Thomaz, Guy Hoffman, and Matt Berlin. 2005. Effects of nonverbal communication on efficiency and robustness in human-robot teamwork. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. 383–388.

[6] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2016. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. (2016).

[7] Kristina Chodorow. 2013. *Mongo DB: The Definitive Guide*. 432 pages.

[8] Kerstin Dautenhahn. 2007. Methodology & themes of human-robot interaction: A growing research field. *International Journal of Advanced Robotic Systems* 4 (2007), 103–108.

[9] Salle Dhali. 2015. Vision based gesture recognition with Kinect sensor. (2015), 1–21.

[10] Dan Witzner Hansen and Qiang Ji. 2010. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (2010).

[11] David Klotz, Johannes Wienke, Julia Peltason, Britta Wrede, Sebastian Wrede, Vasil Khalidov, and Jean-Marc Odobez. 2011. Engagement-based Multi-party Dialog with a Humanoid Robot. *Proceedings of the SIGDIAL 2011 Conference* (2011), 341–343.

[12] Seongyong Koo and Dong Soo Kwon. 2009. Recognizing human intentional actions from the relative movements between human and robot. *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication* (2009), 939–944.

[13] Seongyong Koo and Dong Soo Kwon. 2009. Recognizing human intentional actions from the relative movements between human and robotKoo, S., & Kwon, D. S. (2009). Recognizing human intentional actions from the relative movements between human and robot. In Proceedings - IEEE International Workshop. In *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*. 939–944.

[14] Thi-Lan Lan Le, Minh-Quoc Quoc Nguyen, and Thi-Thanh-Mai Thanh Mai Nguyen. 2013. Human posture recognition using human skeleton provided by Kinect. *2013 International Conference on Computing, Management and Telecommunications (ComManTel)* (2013), 340–345.

[15] Giorgio Metta, Paul Fitzpatrick, and Lorenzo Natale. 2006. YARP: Yet another robot platform. (2006), 043–048 pages.

[16] Marek P. MP Michalowski, Selma Sabanovic, and Reid Simmons. 2006. A spatial model of engagement for a social robot. *International Workshop on Advanced Motion Control, AMC* 2006 (2006), 762–767.

[17] Christophe Mollaret, Alhayat Ali Mekonnen, I Ferran, J Pinquier, Christophe Mollaret, Alhayat Ali Mekonnen, and I Ferran. 2015. Perceiving user ' s intention-for-interaction : A probabilistic multimodal data fusion scheme . (2015).

[18] Plinio Moreno, Ricardo Nunes, Rui Figueiredo, Ricardo Ferreira, Alexandre Bernardino, José Santos-Victor, Ricardo Beira, Luís Vargas, Duarte Aragão, and Miguel Aragão. 2016. Vizzy: A humanoid on wheels for assistive robotics. *Advances in Intelligent Systems and Computing* 417 (2016), 17–28.

[19] James Munkres. 1957. Algorithms for the Assignment and Transportation Problems. *J. Soc. Indust. Appl. Math.* 5 (1957), 32–38.

[20] Bilge Mutlu, Fumitaka Yamaoka, Takayuki Kanda, Hiroshi Ishiguro, and Norihiro Hagita. 2009. Nonverbal Leakage in Robots : Communication of Intentions through Seemingly Unintentional Behavior. 2 (2009).

[21] Chrystopher L Nehaniv, Kerstin Dautenhahn, Jens Kubacki, Martin Haegele, and Christopher Parlitz. 2005. A Methodological Approach relating the Classification of Gesture to Identification of Human Intent in the Context of Human-Robot Interaction. (2005), 371–377.

[22] O Patsadu, C Nukoolkit, and B Watanapa. 2012. Human gesture recognition using Kinect camera. *Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on* (2012), 28–32.

[23] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Mg. 2009. ROS: an open-source Robot Operating System. *Icra* 3 (2009), 5.

[24] Julia Schwarz, Charles Claudius Marais, Tommer Leyvand, Scott E. Hudson, and Jennifer Mankoff. 2014. Combining body pose, gaze, and gesture to determine intention to interact in vision-based interfaces. *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14* (2014), 3443–3452.

[25] Brian A Smith, Q. Yin, Steven K Feiner, and Shree K Nayar. 2013. Gaze Locking: Passive Eye Contact Detection for Human-Object Interaction. *ACM Symposium on User Interface Software and Technology* (2013), 271–280.

[26] Bartosz Tworek, Alexandre Bernardino, and J Santos-Victor. 2008. Visual self-calibration of pan-tilt kinematic structures. *Robotica* (2008).

[27] Adam Yedidia. 2016. Against the F-score. (2016), 1–14.

[28] Xinshuang Zhao, Ahmed M. Naguib, and Sukhan Lee. 2014. Kinect based calling gesture recognition for taking order service of elderly care robot. In *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, Vol. 2014-Octob. 525–530.