Gestural Human-Robot Interaction

Insaf Ajili, Malik Mallem, Jean-Yves Didier

Gestural Human-Robot Interaction

Ajili Insaf, Mallem Malik, Jean-Yves Didier
Laboratoire d'Informatique, Biologie Intégrative et Systèmes Complexes (IBISC)
e-mail: (Insaf.Ajili, Malik.Mallem, Jean-Yves.Didier)@ufrst.univ-evry.fr

Abstract— **Interactive robotics is a vast and expanding research field. Interactions must be sufficiently natural, with robots having socially acceptable behavior for humans, adaptable to user expectations, thus allowing easy integration in our daily lives in various fields (science, industry, domestic, health ...). In this context, we will achieve a system that involves the interaction between the human and the NAO robot. This system is based on gesture recognition via Kinect sensor. We choose the Hidden Markov Model (HMM) to recognize four gestures (move forward, move back, turn, and stop) in order to teleoperate the NAO robot. To improve recognition rate, data are extracted with Kinect depth camera under ROS, which provides a node that tracks human skeleton. We tried to choose a feature vector as relevant as possible to be the input of the HMM. We performed 3 different experiments with two types of features extracted from human skeleton. Experimental results indicates that the average recognition accuracy is near 100%.**

Keywords— **Human-Robot Interaction, Gesture Recognition, Skeleton Tracking, Kinect, HMM, ROS.**

I. Introduction

New human-robot interaction techniques are regularly proposed in the scientific literature in order to improve it and make it more effective and more natural, especially gestural interaction, which is one of the key components of a true interaction. Indeed 65% of the information in communication acts is non-verbal and this type of communication makes the interaction more robust. A gesture is a movement of body parts having a particular meaning. For this, gesture recognition presents one of the many challenges in human-robot interactions field. Dynamic gesture recognition systems typically includes three stages, detection, tracking and finally recognition. For the first phase "detection", there are methods based on the features, the best known is the skin color [1]. However, this method is not robust enough to deal with dynamic conditions. In fact it requires a uniform lighting and background. Other methods based on matching have been developed by [2]. They have some drawbacks since they are very sensitive to rotation and scaling, hence are not reliable for dynamic characteristics. Therefore, recent studies tend to use a new information that is the depth. This information is relevant in any interaction. During the recent years this field gained from the growing use of 3D sensors to improve the performance of gesture detection and tracking. Among these sensors, there is the stereo camera used in [3] to track two hands. However these stereoscopic systems work only for textured scenes, require contrast and visual texture in the scene for stereo matching. They can fail segmentation in several occasions such as when an object and a background share the same texture. After that, a new 3D camera model appeared, Swiss Ranger SR-2. It is an active sensor that provides depth frames in real time. In contrast to stereo camera which was subject to geometric constraints imposing a minimum size, these Time-Of-Flight (TOF) 3D cameras can be designed with compact size. Therefore some researchers used this camera for detecting and tracking gestures [4]. But this type of camera also has some limitations when using it such as interference due to lighting conditions. In such cases depth information will be noisier in the exposed areas to direct light so this drastically limits its use. Then, the Kinect sensor appeared with a high VGA resolution 640×480 pixels as opposed to TOF camera which still have a very limited resolution of 200×200 pixels also provides a good depth range and frame-rate at 30 fps. Several studies have been based on this sensor, to track hand gestures [5]. Some researchers combined color and depth information for segmentation and tracking body members as in [6] for hand tracking. Recently the founding company of the sensor has released a development kit for body tracking. It contains a skeleton tracking algorithm to track the human body through the Kinect sensor. Then many researchers used this algorithm which consists in projecting a skeleton on the human body image so that each joint of the body will be related to a joint of the skeleton and labelled with identifier which will be subsequently used as information for the gesture recognition system. Now for the final step of gesture recognition, the hidden Markov model (HMM) has been widely used in the community for modeling dynamic movements. This approach was widely in use for speech recognition and has already been exploited in gestures recognition. In order to achieve the highest recognition rate one should find the most appropriate features to describe a set of gestures. Some works have used HMM and choose the orientation between two successive hand frames as a feature vector to recognize geometric shapes [7]. In [8] they kept the same feature vector to recognize a set of gestures such as (up, down, left, right) and three characters. Another way is to combine orientation with location and velocity features in order to recognize alphanumeric characters [9]. Others took the advantage of the Kinect in skeleton tracking, for example in [10], the feature vector that they choose is composed of distance feature from the left/right elbow/hand to the spine. In [11] they used HMM to recognize four command gestures to control a robot and they choose as feature vector the central hand detected in 3D by the Kinect and projected in 2D space. In [12] they recognized ten gestures chosen from the army visual signals to control a mobile robot. Gesture recognition was performed by neural network (NN) classifiers, they extracted two types of features (quaternions and angles) and compared between them to obtain a high recognition rate. In [13] they used weighted Dynamic Time Warping method to recognize two gestures. Two features had been extracted for wave gesture (euclidean distance between the hand and the neck joints and the angle in the elbow joint) and three features for the point at one gesture (angle in the elbow joint, distance between the hip and the hand joints, and the position of the hand joint), in order to interact with Nao robot. In [14] they integrated the Euler angle to recognize left arm gestures to control the pionner robot with five gestures (come, go, wave, rise up and sit down). Their features are four joint angles (left elbow yaw and roll, left shoulder yaw and pitch).

In our work, we use four gestures (go forward, go back, turn around and stop) to command Nao robot. Gesture recognition is performed with HMM approach, then we tried to find relevant features that adapt to different persons. So we selected joint orientations features as [12] to be independent of person characteristics and then allow any person to control the robot without the need of normalization. We tried to improve the recognition result with combining two types of orientation features (quaternions and angles). Features were provided by skeleton tracking algorithm using Microsoft Kinect under ROS, to be the input to HMM to classify gestures. The rest of the paper is organized as follows: in section II, we define our recognition system as well as its various stages. In section III we present the experiment result, and we will finally conclude with some perspectives.

II. PROPOSED PROCESS

Our proposed system consists of three stages. The first is the data acquisition and feature extraction that will be made by the Kinect camera. Here we seek the most relevant feature vector as possible to have an important recognition rate. The second step is the tracking performed under ROS (Robot Operating System) which was created to facilitate writing software for robotic task. This one will set up a human tracking node created by the OpenNI driver of Kinect. Finally, for gesture recognition step, we will use the HMM formalism and therefore we will implement the Baum-welch algorithm for training phase and Forward-Backward algorithm for gesture classification.

1) Data acquisition: The first step in our recognition system is the data acquisition by the Kinect that records gesture motion in the space under ROS. Kinect camera is a device that works as a webcam (Fig.1). To use this sensor we have installed three drivers, the sensor Kinect module, the NITE middleware, and the OpenNI driver [15]. The Microsoft Kinect is integrated into the ROS environment, an operating system for robotic service. The basic principle of ROS is to run in parallel a large number of executables (called nodes) that need to exchange information synchronously via topics or asynchronously via services. An OpenNI package under ROS contains launch files allowing to use the Kinect sensor in ROS system. This package provides a node called "openni_tracker" which requires user calibration (Fig.2) in order to achieve full body tracking. Once the calibrating step is performed we launch the openni_tracker node for Skeleton tracking and thus publish the OpenNI skeleton frames in a specific topic called "tf"(transformation) (Fig.3).

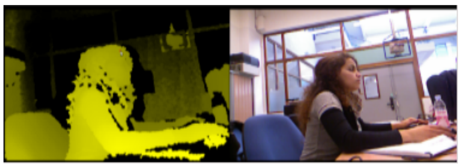
Fig. 1. Depth-RGB camera Kinect.

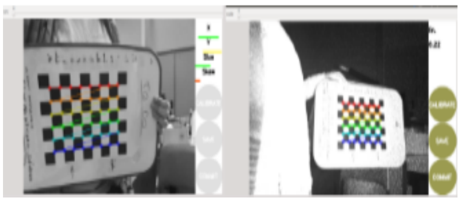
Fig. 2. Kinect calibration.

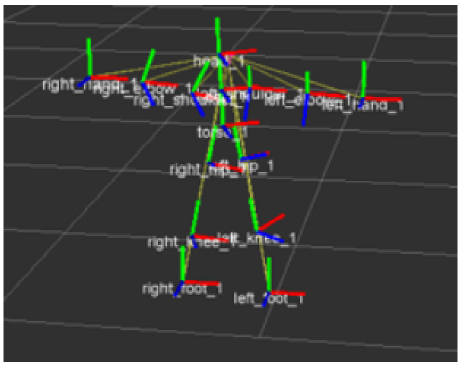

Fig. 3. Skeleton frames.

2) Feature extraction: In our work it is desired to select a set of gestures to teleoperate robot, so we need the following gestures: move forward, move back, turn around and stop. Commands are performed with the left arm. For the choice of the feature vector we tried to find the most robust features without integrating normalization. So features were independent of the height and position of the person. For this reason, we choose rotation features. We compared two types of orientation quaternion and angle. Under ROS, there is an "openni_tracker" node which presents a rigid transformation stored as a rotation quaternion and a 3d translation of each joint skeleton. For the rotation part we were interested by the quaternions of the left elbow and shoulder joints to describe 3D rotation of the left arm. A quaternion q is presented as follows: $q = a + bi + cj + dk$ where a, b, c, d are real numbers and i, j, k imaginary units. It presents a rotation through an angle α around the axis given by a unit vector (x, y, z). With $\alpha = 2\arccos(a) = 2\arcsin\sqrt{b^2 + c^2 + d^2}$ and the axis $(x, y, z) = \frac{1}{sin(\alpha/2)}(b, c, d)$. So the input of HMM model will be the descriptor vector which composed by 8 features (four quaternions of shoulder joint (*sh*) and elbow joint (*el*)) for each frame: $(a_{sh}, b_{sh}, c_{sh}, d_{sh}, a_{el}, b_{el}, c_{el}, d_{el})$.

After that we will choose another feature vector extracted from the translation part, this descriptor vector present three joint angles $(\theta_1, \theta_2, \theta_3)$ of the left arm (Fig.4).
$\theta_1 = \widehat{L_h L_e L_s}$; $\theta_2 = \widehat{L_e L_s R_s}$; $\theta_3 = \widehat{L_e L_s L_{hi}}$

Where L_h: Left hand joint, L_e: Left elbow joint, L_s: Left shoulder joint, R_s: Right shoulder joint and L_{hi}: Left hip joint.

Then we compare between the two vectors, and we try to create a feature that provides the highest recognition rate. The result is described in the experimental part.

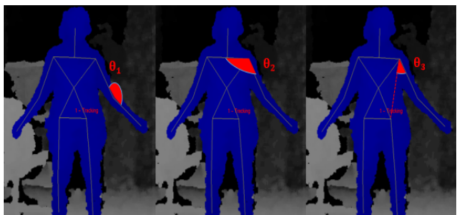

Fig. 4. Skeleton tracking, second feature vector $(\theta_1, \theta_2, \theta_3)$.

3) Discretization: Once we turned raw data into relevant features, we should sample these data to make sequences of frames with the same length. For this, we implemented a C++ program that consists in sampling data. We suppose that a feature vector is composed by N features $(X_1, X_2, \ldots, X_N)$. And d is the distance between two points (two successive feature vectors): $d = \sqrt{\sum_{i=1}^{N}((X_t)^i - (X_{t+1})^i)^2}$.

$(X_t)^i$ is the i^th feature of the current frame.

$(X_{t+1})^i$: is the i^th feature of the next frame.

N is the size of feature vector (the number of features in a descriptor vector).

The discretization algorithm is implemented as follows:

1) Delete similar points. If d the distance between two successive points p_1 and p_2 is less or equal than ϵ delete p_2 (the second feature vector).
2) Set length gesture (number of frames T).
3) Compute the average distance: $d'= \frac{N'}{T}$, N' is number of points (feature vectors) after removing noises.
4) Discretize points with average distance. Get points after every d' points $p_{i+1} = p_i + d'$.

4) Vector quantization: Once we sampled data, we pass to the quantization step to translate feature vectors into finite symbols, to be subsequently accepted by HMMs as input. For this quantization we use K-means [16] as a clustering algorithm which is based on minimum distance between the center of each cluster and the feature point. So given an integer K, we separate a set of points in K clusters. This task is implemented for both training phase to partition training and testing data into k clusters.

5) Hidden Markov Model: Hidden Markov model is a statistical Markov model which defines two properties. First it supposes that each observation was issued by a hidden state. Second it supposes that given the value of s_{t-1}, the current state s_t is independent of all the states prior to $t-1$. A hidden Markov model consists of the following elements:

- $\{s_1, \ldots, s_N\}$ a finite set of states, N is the number of states.
- $\{v_1, \ldots, v_M\}$ a finite observation symbols, M is the number of symbols.
- A= $\{a_{ij}\}$ an $N \times N$ matrix which represents the state transition probabilities, also called transition matrix, where a_{ij} represents the transition probability from the state s_i to the states s_j. $a_{ij} = P(q_{t+1} = s_j | q_t = s_i)$ $1 \le i, j \le N$, q_t indicates the state at time t.
- B= $\{b_{jk}\}$ an $N \times M$ matrix represents the observation probability distribution. $b_j(k) = P(o_t = v_k | q_t = s_j)$ $1 \le j \le N$ and $1 \le k \le M$, is the probability of generating the k^{th} observation symbol at time t in the state s_j.
- $\pi_i = \{\pi_i\}$ initial state distribution (when $t = 1$), $\pi_i = P(q_1 = s_i)$ is the probability that the state s_i is the initial state.

Therefore we can use the compact notation $\lambda = (A, B, \pi)$ to define an HMM.

6) Initializing HMM parameters: In the initialization step we should first choose HMM topology. In fact there are three variations of HMMs: the first variation is the ergodic model which has more transitions than the others. Any state in this model can be reached from any other state. It is a fully connected model. The second topology is the left-right model where the states proceed from left to right (each state can transition to itself or to the following states). The third model is the left-right banded model (Fig.5), where each state can go back to itself or to the next state. We choose this type which restricts transition between states. It is the simplest model which make easier matching of the data to the model, and fits well our gestures that evolve over time, thus do not need backward transition.

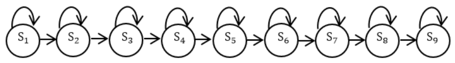

Fig. 5. Left-Right Banded model with nine states.

We propose four gestures for robot teleoperation, so we need four HMMs. The first step to do in the HMM formalism is the initialization. This phase requires a good choice of state number. So we compared the results found in each gesture at each change of state number. After many tests by changing the number of states from 4 to 11, it was concluded that the best state number for recognition gesture was 9. So for an HMM with 9 states we need to define the initial vector $\pi =$

$(1\,0\,0\,0\,0\,0\,0\,0\,0)^T$. Second, we define the transition matrix as follows:

$$\begin{bmatrix} a_{ii} & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_{ii} & r & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_{ii} & r & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_{ii} & r & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{ii} & r & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & a_{ii} & r & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{ii} & r & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{ii} & r \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Such that, $a_{ii} = 1 - \frac{N}{T}$, $r = 1 - a_{ii}$, N is the number of states (9 states) and T the length of gesture path, we choose 60 frames. And finally the emission matrix $B = \{b_{im}\} = \frac{1}{M}$, M being the number of symbols, in our case we choose 12 symbols.

7) Training phase: We have used an iterative algorithm [17] to train each HMM through training data (Fig.6). It adjusts each initial model parameters in such a way that they achieve the maximum likelihood $P(O|\lambda)$ for the given training data. So given a hidden Markov model with fixed architecture (initial probability vector, transition probability matrix and emission probability matrix), we must iteratively reestimate model parameters to have a maximum probability of generating all training sequences. To compute these new parameters, the Baum-Welch algorithm uses two new probability matrices:

$\epsilon_t(i,j) = P(q_t = s_i, q_{t+1} = s_j|O,\lambda)$

$\gamma_t(i) = \sum_{j=1}^{N} \epsilon_t(i,j)$.

The coefficient $\epsilon_t(i,j)$ represents the probability to switch from state s_i at time t to the state s_j at time $t+1$ given the model and the observation sequence. The second coefficient $\gamma_t(i)$ represents the probability to be in the state s_i at time t given the model and the observation sequence. Then reestimation parameters will be denoted as:

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T-1} \epsilon_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \;;\; \overline{b}_j(k) = \frac{\sum_{t=1,O_t=v_k}^{T-1} \gamma_t(j)}{\sum_{t=1}^{T-1} \gamma_t(i)};\; \overline{\pi} = \gamma_1(i)$$

After training we get the new parameters $\overline{\lambda} = (\overline{\pi}, \overline{A}, \overline{B})$ for each gesture. This provides $P(O|\overline{\lambda}) \geq P(O|\lambda)$. These new parameters will be the inputs of the Forward-Backward algorithm for tests.

8) Recognition phase: We have used Forward-Backward algorithm [17] to classify input gestures with the trained models (Fig.7). It is the most efficient algorithm to address this problem. We note that the observation can be done in two stages. First the emission from the beginning of observation $O(1:T)$ leading to the state q_i at time t then the emission from the end of the observation $O(t+1:T)$ starting from q_i at time t. So the probability of the observation is equal to: $P(O|\lambda) = \sum_{i=1}^{n} \alpha_T(i)\beta_T(i)$, with $\alpha_T(i) = P(o_1, o_2, \ldots, o_t, q_t = s_i|\lambda)$ is the probability of transmitting the observation sequence $O(1:T)$ and end up to q_t at time t and $\beta_t(i) = P(o_{t+1}, o_{t+2}, \ldots, o_T|q_t = s_i, \lambda)$ is the probability of transmitting the observations from time $t+1$ up

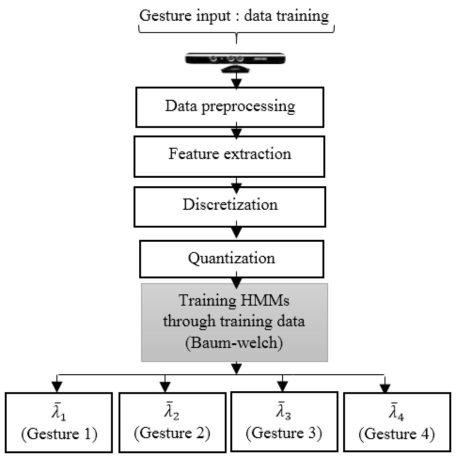
Gesture input : data training
Data preprocessing
Feature extraction
Discretization
Quantization
Training HMMs through training data (Baum-welch)
$\bar{\lambda}_1$ (Gesture 1)
$\bar{\lambda}_2$ (Gesture 2)
$\bar{\lambda}_3$ (Gesture 3)
$\bar{\lambda}_4$ (Gesture 4)

Fig. 6. Training phase.

to T knowing that we part from q_t at time t. After evaluating the probability of generating the sequence observation by each HMM, we can determine the model that gives the highest probability $\lambda^* = \underset{all\lambda}{argmax} P(O|\lambda)$.

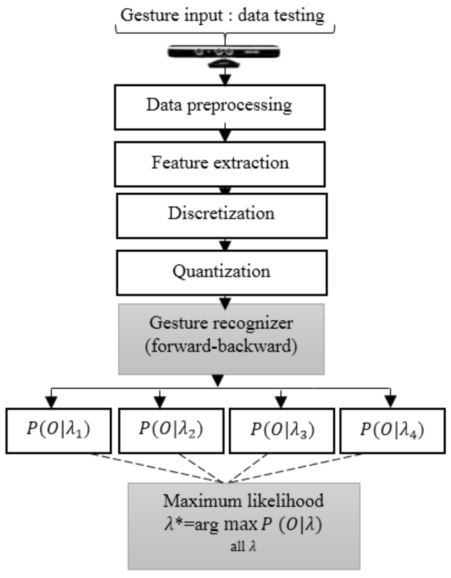
Gesture input : data testing
Data preprocessing
Feature extraction
Discretization
Quantization
Gesture recognizer (forward-backward)
$P(O|\lambda_1)$
$P(O|\lambda_2)$
$P(O|\lambda_3)$
$P(O|\lambda_4)$
Maximum likelihood λ^*=arg max $P(O|\lambda)$ all λ

Fig. 7. Recognition phase.

III. EXPERIMENTAL RESULTS

Our proposed gesture recognition system was applied to a database composed of 400 sequences from ten subjects. Each subject was asked to make each gesture ten times (move forward, move back, turn around, stop). After that we divided them into two sets, 200 samples for training set and 200 for testing set. Discretization algorithm was applied to fix the number of frames for each gesture repetition, we choose 60 as the length of gesture. Once we defined this vector, we quantify it into discrete values (we choose 12 as the number of symbols) to be the input to HMM model for the training and classification phases. We used the Forward-Backward algorithm to classify every input gesture, then we consider probabilities of belonging to each model. The highest one is considered as the recognition result. Recognition rate was defined as the ratio of the number of recognized gestures over the number of input gestures.

$$Recognition\,rate(\%) = \frac{number\,of\,recognized\,gestures}{number\,of\,input\,gestures} \times 100.$$

We conducted three experiments:

In the first experiment every gesture is defined by an HMM model, for the feature vector we choose quaternions of left elbow and shoulder joints for each frame $F1 = [a_{sh1}, b_{sh1}, c_{sh1}, d_{sh1}, a_{el1}, b_{el1}, c_{el1}, d_{el1}, \ldots, a_{shT}, b_{shT}, c_{shT}, d_{shT}, a_{elT}, b_{elT}, c_{elT}, d_{elT}]$, T presents the number of frames. We choose a duration of 2 seconds between two successive frames. *el* means elbow joint and *sh* shoulder joint. We found a recognition rate which reached 100% for move back gesture, 98% for turn around gesture. Then three unrecognized gestures for stop gesture and two for move forward gesture (Tab.1).

For the second test we used joint angles as a feature vector $F2 = (\theta_{1,1}, \theta_{2,1}, \theta_{3,1}, \theta_{1,2}, \theta_{2,2}, \theta_{3,2}, \ldots, \theta_{1,T}, \theta_{2,T}, \theta_{3,T})$. $\theta_{i,j}$ means the angle i at frame j. We have displayed these angles to show their variation for each gesture. It was noted that with the first gesture “move forward“ the most variating angle is θ_1 ranging from 160 to 20 degree (Fig.8). For the “move back“ gesture the same angle varies the most in the opposite direction on average from 20 to 160 degree (Fig.9). For the “turn around“ gesture, θ_2 is the angle that varies the most from 90 to 160 degree (Fig.10). And finally in “stop“ gesture, θ_3 changes in a remarkable way from 15 to 100 degree (Fig.11). With this second choice of descriptor vector, we found the same result as quaternions feature for move back and turn around gestures, but a recognition rate of 92% for move forward gesture and 98% for stop gesture (Tab.2).

In the final test we combined the two results. Figure 12 describes the idea of the final experiment. Each gesture was presented with two HMMs, the first with the joint angles descriptor (λ_i^1 ; $1 \leq i \leq 4$) and the second with the quaternions features (λ_i^2 ; $1 \leq i \leq 4$). In the training phase, we trained 8 HMMs (2 HMMs by gesture) and in the recognition phase, we computed the maximum likelihood of each sequence given the two types of feature vectors of each HMM model. In this case we obtain four optimal HMMs $(\lambda_1^*, \lambda_2^*, \lambda_3^*, \lambda_4^*)$. We can determine the gesture class from the maximum likelihood between this four models. By applying this algorithm, we found the same result with the two types of feature vector for turn around (98%) and move back (100%) gestures. For stop gesture we found three errors with the quaternions features and one error with the joint angles features. The combination of the two results let us to have a recognition rate of 100% (Fig.13). For the move forward gesture we found two errors with the first feature vector and four errors with the second vector. The fusion of two results returned one error (Fig.14), which leads up to a recognition rate of 98%. So, this makes us to reach very important recognition rates for all gestures [Tab.3].

- Application under ROS to control Nao robot:

Now we want to test our application with Nao robot to evaluate its performance. We have implemented four nodes under ROS: “openni_tracker“ node that publishes coordinate joints skeleton. A second node “sequence_observation“ which subscribe to the first node to have coordinate joints informations and generates the sequence observation of each test gesture. It performs data discretization and quantization. The third node “recognition_node“ subscribes to the second and computes the probability of generating observation sequence by each HMM, and then returns the model that corresponds to the test gesture. Finally we launch the “teleoperation_node“, this node was extracted from the idea of [18] to teleoperate Nao robot (Fig.15).

Table 1. Gesture Recognition rates with quaternions feature vector.

Gesture	Move forward	Move back	turn around	stop
Move forward	48	0	0	0
Move back	0	50	0	0
Turn	0	0	49	0
Stop	0	0	0	47
Recognition rate	96%	100%	98%	94%

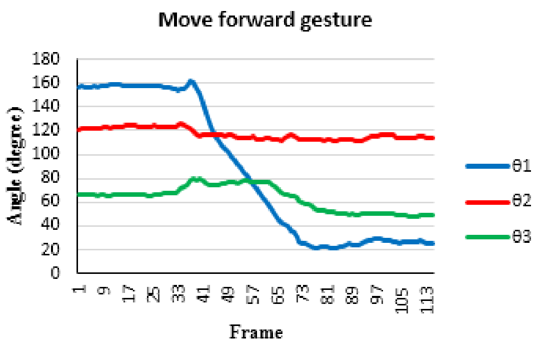

Fig. 8. Angles variation in move forward gesture.

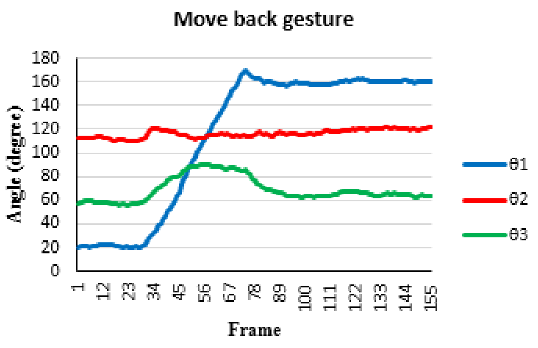

Fig. 9. Angles variation in move back gesture.

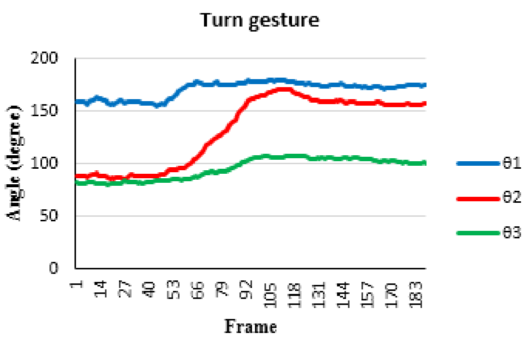

Fig. 10. Angles variation in turn around gesture.

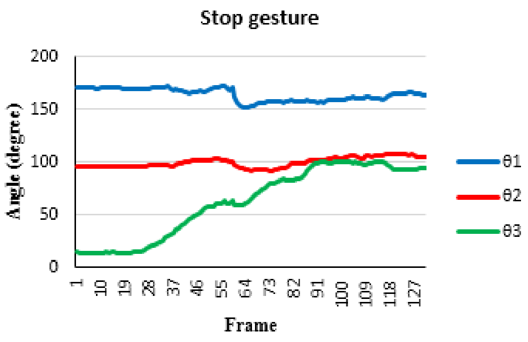

Fig. 11. Angles variation in stop gesture.

Table 2. Gesture Recognition rates with joints angles feature vector.

Gesture	Move forward	Move back	turn around	stop
Move forward	46	0	0	0
Move back	0	50	0	0
Turn	0	0	49	0
Stop	0	0	0	49
Recognition rate	92%	100%	98%	98%

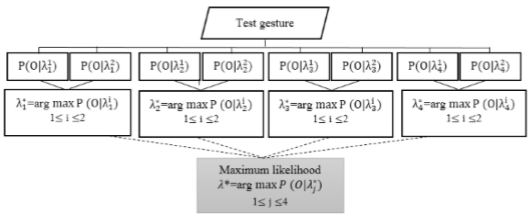
Test gesture
P(O|λ$_1^1$)
P(O|λ$_1^2$)
P(O|λ$_2^1$)
P(O|λ$_2^2$)
P(O|λ$_3^1$)
P(O|λ$_3^2$)
P(O|λ$_4^1$)
P(O|λ$_4^2$)
λ$_1^*$=arg max P (O|λ$_1^i$) 1≤ i ≤2
λ$_2^*$=arg max P (O|λ$_2^i$) 1≤ i ≤2
λ$_3^*$=arg max P (O|λ$_3^i$) 1≤ i ≤2
λ$_4^*$=arg max P (O|λ$_4^i$) 1≤ i ≤2
Maximum likelihood
λ*=arg max P (O|λ$_j^*$) 1≤ j ≤4

Fig. 12. Recognition phase with combining two feature vectors.

Table 3. Gesture Recognition rates with combining two feature vectors.

Gesture	Move forward	Move back	turn around	stop
Move forward	49	0	0	0
Move back	0	50	0	0
Turn	0	0	49	0
Stop	0	0	0	50
Recognition rate	98%	100%	98%	100%

IV. Conclusion

In this paper we implemented a real-time gesture recognition system using Kinect sensor under ROS based on the

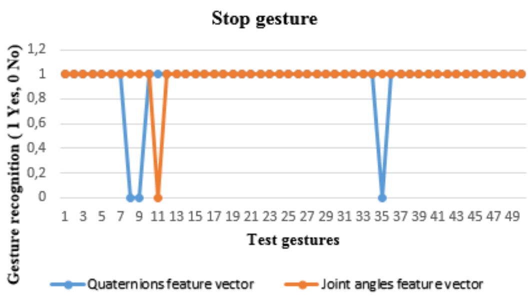

Fig. 13. Result of the fusion of two feature vectors for stop gesture.

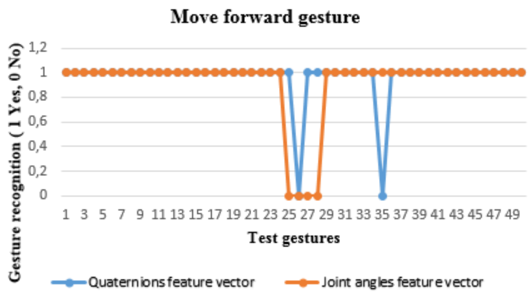

Fig. 14. Result of the fusion of two feature vectors for move forward gesture.

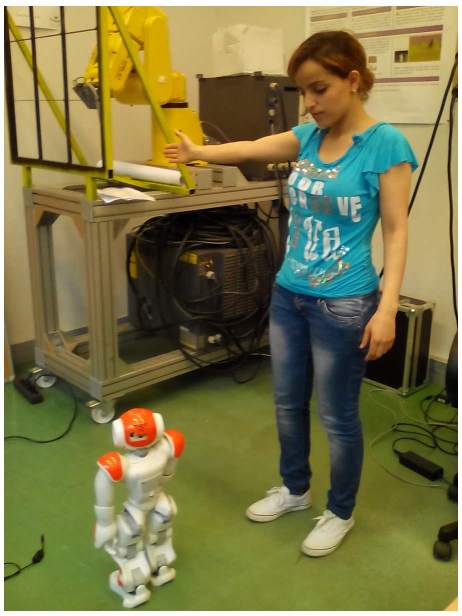

Fig. 15. Human-Nao interaction with gestures.

hidden Markov model. Our data base was trained with different subjects, and the idea of merging both results provided by two types of orientation feature vectors led us to achieve recognition rates near 100%. At the same time we implemented a system that was independent from many factors (size and position of people, gesture speed and duration). After this work we will continue on the same issue for gesture recognition, we will add other gestures for robot control in order to vary activities for Nao.

REFERENCES

[1] Reza Azad and Fatemeh Davami. A robust and adaptable method for face detection based on Color Probabilistic Estimation Technique. International Journal of Research in Computer Science A Unit of White Globe Publications, 2014.

[2] M. Pierobon and M. Marcon and A. Sarti and S. Tubaro. 3-D Body Posture Tracking For Human Action Template Matching, Acoustics. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), II-II, 2, 2006.

[3] Rustam Rakhimov Igorevich and Pusik Park and Jongchan Choi and Dugki Min. Two Hand Gesture Recognition Using Stereo Camera. International Journal of Computer & Electrical Engineering, 5(1), 2013.

[4] D. Droeschel and J. Stückler and S. Behnke. Learning to interpret pointing gestures with a time-of-flight camera. 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 481-488, 2011.

[5] Qin, Shuxin and Zhu, Xiaoyang and Yang, Yiping and Jiang, Yongshi. Real-time Hand Gesture Recognition from Depth Images Using Convex Shape Decomposition Method. Journal of Signal Processing Systems, 74(1), 47–58, 2014.

[6] Oikonomidis, Iason and Kyriazis, Nikolaos and Argyros, Antonis A. Efficient model-based 3D tracking of hand articulations using Kinect. Proceedings of the 22nd British Machine Vision Conference, BMVC'2011, University of Dundee, UK, Aug. 29-Sep. 1, 2011.

[7] Kshitish Milind Deo, Avanti Yashwant Kulkarni and Tirtha Suresh Girolkar. Hand Gesture Recognition using Colour Based Segmentation and Hidden Markov Model. International Journal of Computer Applications Technology and Research, 4(5), 386–389, 2015.

[8] Y. Wang and C. Yang and X. Wu and S. Xu and H. Li. Kinect Based Dynamic Hand Gesture Recognition Algorithm Research. 4th International Conference on Intelligent Human-Machine Systems and Cybernetics, 274–279, 2012.

[9] Ayoub Al-Hamadi, Mahmoud Elmezain, and Bernd Michaelis. Hand Gesture Recognition Based on Combined Features Extraction. International Journal of Information and Mathematical Sciences, 6(1), 2010.

[10] K. Lai and J. Konrad and P. Ishwar. A gesture-driven computer interface using Kinect. IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI), 185–188, 2012.

[11] Kun Qian and Jie Niu and Hong Yang. Developing a Gesture Based Remote Human-Robot Interaction System Using Kinect. International Journal of Smart Home, 7(4), 2013.

[12] Grazia Cicirelli, Carmela Attolico, Cataldo Guaragnella and Tiziana D'Orazio. A Kinect-based Gesture Recognition Approach for a Natural Human Robot Interface. International Journal of Advanced Robotic Systems, 2015.

[13] G. Canal and C. Angulo and S. Escalera. Gesture based human multi-robot interaction. International Joint Conference on Neural Networks (IJCNN), 2015.

[14] Y. Gu and H. Do and Y. Ou and W. Sheng. Human gesture recognition through a Kinect sensor. IEEE International Conference on Robotics and Biomimetics (ROBIO), 1379–1384, 2012.

[15] http://structure.io/openni.

[16] J. B. MacQueen. Some Methods for classification and Analysis of Multivariate Observations. Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, 281-297, 1967.

[17] Baum, L. E. . An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. In Shisha, O. (Ed.), Inequalities III: Proceedings of the 3rd Symposium on Inequalities, s, University of California, Los Angeles, pp. 1–8, 1972.

[18] I. Almetwally and M. Mallem. Real-time teleoperation and tele-walking of humanoid robot Nao using Kinect depth camera. In Proc. of 10th IEEE International Conference on Networking, Sensing and Control (ICNSC), 2013.