Gesture recognition for robot teleoperation

Insaf Ajili, Malik Mallem, Jean-Yves Didier

Upper body gesture recognition

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 by humans, adaptable to user expectations, thus allowing easy integration in our daily lives in various fields (science, industry, domestic, health ...). To make such interaction we choose gestures as a way of communication. Human gestures are certainly natural and flexible. In this context we developed a robust upper body gesture recognition system in order to teleoperate in the future a humanoid robot. Gestures are performed by Kinect camera for skeleton detection and tracking. A robust descriptor vector is chosen to describe gestures, named BSM feature vector which can be represent three important aspects, The connexion between different Body parts, the Shape changing during gesture and describe gesture Motion in the space. Three best-known and successful learning methods used for training and gesture classification, Random forest classification, Support vector machine and Multi layer Perceptron. The proposed method has been evaluated on two public benchmarks, the Microsoft Research Cambrige (MSRC-12), and MSR Action3D datasets. The results obtained showed that the proposed recognition system is more relevant than the traditional methods.**

***Keywords*— Gesture Recognition, BSM feature, Kinect, Support Vector Machine, Multi-layer Perceptron, Random Forest.**

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 was 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 step, tracking step and finally recognition step. For the first phase "detection", there are methods based on the features, the best known is the skin color [1], other on matching method [15]. But they know many drawbacks since they are very sensitive to many factors such as rotation and scaling, dynamic characteristics,... Therefore, recent studies tend to use a new information that is the depth. This information is relevant in any interaction. For this,in recent years this field has an important presence of 3D sensors to improve the performance of gesture detection and tracking such as stereo camera [10], Time-Of-Flight sensor [7] and finally the most powerful 3D kinect camera with a high VGA resolution 640×480 pixels which provides a good depth range and frame-rate at 30 fps. Several studies have been based on this sensor, to track hand gestures [13]. 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. Many researchers have 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 proposed an associating and identifier which will be subsequently used as information for the gesture recognition system.

In the final step, recognition process affected with suitable classification algorithms (RF, MLP and SVM). In order to achieve the highest recognition rate we have to find the most appropriate features to describe gestures. As [6] they choose the orientation between two successive hand frames as a feature vector to recognize geometric shapes.To recognize a set of characters present in 2D videos [11] histograms of gradient (HOG) are selected to generate a local motion signature. In [9] a covariance matrix for skeleton joint locations over time was considered as a discriminative descriptor for a sequence. Sift and MBH descriptors were choosen for action recognition in [14]. In [16] they choose the center of the hand detected in 3D by the Kinect and projected in 2D space to describe gestures. In [4] 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 [18] a moving pose descriptor is used as a concatenation of the body joints pose and ist first and second order derivatives for action recognition. In this work, upper body gesture recognition algorithm exploiting the skeleton provided by Kinect is proposed. The algorithm starts with the feature extraction to properly describes actions. Learning methods are used for recognition step and finally evaluation has been on two publicly available datasets.

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 our experimental evaluation, and we will finally conclude with some perspectives.

II. PROPOSED PROCESS

Our proposed system consists in recognizing upped body gestures, it is constituted by three fundamental steps:Prepocessing data, Feature extraction and recognition step. For the last stage there are two phases, the first one is the training phase and the second is the classification step.

1) Preprocessing data: The first step in our proposed system is the preprocessing step which is applied to the raw 3D skeleton data to normalize data in order to adapt different user's height and position in relation to kinect. Skeleton data contain 3D position data for human skeletons. Each joint position in the skeleton space is represented as (x, y, z).

This preprocessing stage consists on the following steps:

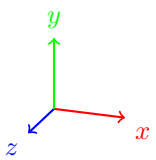
(1) Translation: to define the same origin of coordinates system for all frames, the selected one corresponds to the hip center of the human skeleton.
(2) Normalization: we normalized joint coordinates over the sequence to range from 0 to 1. (the height of the subject is calculated; then all skeleton 3D coordinates are normalized according to the calculated value)
(3) Rotation: we aligned skeleton at initial frameaccording to Anatomical Planes (transversal, frontal and sagittal), in such a way that the user stands straight in front of the kinect in order to guarantee that the gesture is always observed from the same point of view, without regard to the initial pose of the user relating the sensor. So we consider (xOy), (xOz) and (yOz) are respectively the frontal, transversal and sagittal body planes. And we make the following transformations:
   - A rotation around y-axis $R_y(\alpha)$ to have the segment related left shoulder and right shoulder parallel to frontal plane
   - A rotation around x-axis consists $R_x(\beta)$ of aligning shoulder and hip centers with frontal plane.
   - A rotation around z-axis $R_z(\gamma)$ in order to make left and right shoulder parallel with transversal plane.

After calculated the new initial joints positions, we apply these transformations for each frame.

Let $P(x_i, y_i, z_i)$ be the initial position of the joint with index i and $C(c_x, c_y, c_z)$ is the hip center coordinates.

- In translation step:
  $x_i^t = x_i - cx$; $y_i^t = y_i - cy$; $z_i^t = z_i - cz$.
- In normalisation step:
  For a matrix M where each row corresponds to 3D joints coordinates (x_i^t, y_i^t, z_i^t) with $i = 1, \ldots, 20$ is the joint index and columns represents frames.
  $x_i^n = \frac{x_{ij}^t - maxM(:,j)}{maxM(:,j) - minM(:,j)}$; $y_i^n = \frac{y_{ij}^t - maxM(:,j)}{maxM(:,j) - minM(:,j)}$; $z_i^n = \frac{z_{ij}^t - maxM(:,j)}{maxM(:,j) - minM(:,j)}$; where x_{ij}^t is x_i^t at frame j.
- In rotation step:
  $x_i^f = x_i^n * Ry(\alpha) * Rx(\beta) * Rz(\gamma)$; $y_i^f = y_i^n * Ry(\alpha) * Rx(\beta) * Rz(\gamma)$; $z_i^f = z_i^n * Ry(\alpha) * Rx(\beta) * Rz(\gamma)$.

So the preprocessing step results a relevant joint position $P_f(x_i^f, y_i^f, z_i^f)$, $i = 1 \ldots 20$ which is independant from users'height, position and point of view.



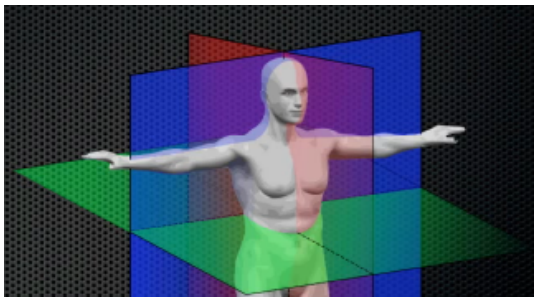


Fig. 1. user.

2) Feature extraction: For a faithful representation of the skeleton in gesture leading to successful recognition, we extract three important types of features.

Body: The first defines the relationship between the different members of the upper body part, here we will present the different angles between the different joints.

Shape: The second descriptor is the change of the upper body shape during the gesture, two characteristics have been defined the convex volume of the upper body part and the area of the triangle formed by three articulations (shoulder center, left hand and right hand) in order to better have Information about the extension of both hands.

Motion: The third feature describe the motion of both arms and head to konw the their displacement in 3D space. It gives us idea about retreat, advancement, horizontal (left/right) and vertical(top/bottom) displacement, with respect to the center hip since we have made a translation of the skeleton to have the center hip as the original reference.

a) Body feature: The first component describes the connexion between body parts, which parts are influenced by others, what is the sequence of the movement between the body parts. We choose seven joints angles as features (Fig. 2) to describe the relation between upper body's articulations which are:

$$\theta_1^{l/r} = h^{l/r}\widehat{el^{l/r}sh^{l/r}}; \theta_2^{l/r} = el^{l/r}\widehat{sh^{l/r}sh^{r/l}};$$
$$\theta_3^{l/r} = el^{l/r}\widehat{sh^{l/r}hi^{l/r}}; \theta_4^{l/r} = he\,\widehat{hc\,kn^{l/r}}.$$

Where $h^{l/r}$: Left/right hand joint, $el^{l/r}$: Left/right elbow joint, $sh^{l/r}$: Left/right shoulder joint, $kn^{l/r}$: Left/right knee joint, he: Head and hc: Hip center.

b) Shape feature: This component analyses the changing forms that the body makes during movement. For this feature we compute the volum of the convex hull of a set points which are (head ,left elbow, right elbow, left hand, right hand and hip center) (Fig. 3). It is based on Quickhull algorithm which uses two geometric operations: oriented hyperplane through a set of points and signed distance to hyperplane. The hyperplane defines a halfspace of points that have negative distances from the hyperplane. If the distance is positive, the point is above the hyperplane. The principle of this algorithm is to find a convex polygon using a set of points S, for which all points

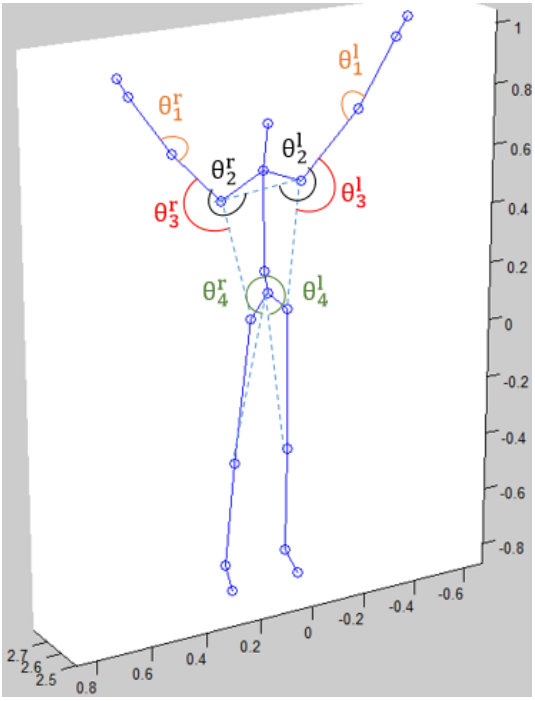


Fig. 2. Joint angles feature vector.

lies inside it.

By calculating convex volume of the upper part one will have an idea on the change of its form during a gesture. On the other hand in some gestures we can be not suffisientto make difference between gestures for that we added a second shape descriptor which gives more information about the extension of two arms which is the area of triangle formed by both hands and shoulder center (Fig. 4).

Let $P_1P_2P_3$ be the triangle formed by respectively shoulder center, left hand, and right hand joints.

First we compute the orthogonal vector of two vectors $P_1\vec{P}_2$ $P_1\vec{P}_3$ and by taking their cross product:

$\vec{u} = P_1\vec{P}_2 \wedge P_1\vec{P}_3$

Area = $\frac{|\vec{u}|}{2}$.

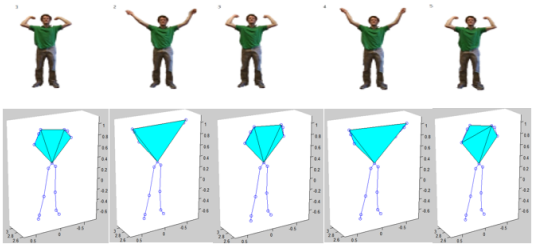


Fig. 3. Upper body Convex hull.

c) Motion feature: For motion feature we describe upper body part deplacement in 3D space, specially both hands, shoulders and the head joints. So a each frame we extract (x, y, z) coordinates of each joint. Then we describe the gesture paths according to the horizontal plane (extensio/flexion), the vertical plane (elevation/abaissement), and the sagittal plane (advancement/retreat) (Fig. 5). Therefore we have a descriptor vector of size 15 features.

Algorithm 1 Quickhull

procedure QHULL(S)
  $S \leftarrow n$ points
  Select farthest left point p_1
  Select the furthest right point p_2
  $S = G_1 \bigcup G_2$ ▷ $[p_1p_2]$ *divides the remaining* $(n-2)$ *points into two groups* G_1 *and* G_2. G_1 *are points in* S *that are in the right side of the oriented line from* p_1 *to* p_2 *and* G_2 *are points in* S *that are in the right side of the oriented line from* p_2 *to* p_1.
  FindHull (G_1, p_1, p_2)
  FindHull (G_2, p_1, p_2)
  procedure FINDHULL(G, a, b)
    $G \leftarrow$ set of points

    if $G = \emptyset$ **then** return
    else
      Find farthest point, say c from $[ab]$
      Add point C to convex hull at the location between a and b
      $G = s_0 \bigcup s_1 \bigcup s_2$
      $s_0 \subset \triangle p_1Cp_2$
      $s_1 \subset$ right side of the oriented line from a to C
      $s_2 \subset$ right side of the oriented line from c to C
      FindHull (s_1, a, C)
      FindHull (s_2, C, b)
    end if
  end procedure
end procedure

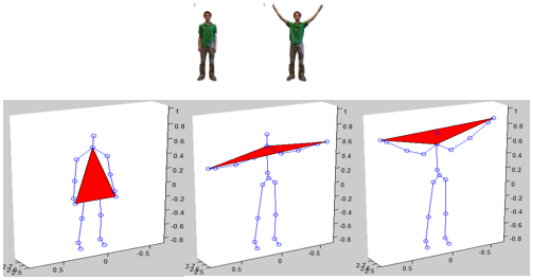


Fig. 4. Triangle area formed by (both hands and shoulder center joints).

In total at each frame we derive a feature vector composed of 25 features.

3) Sampling: 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++

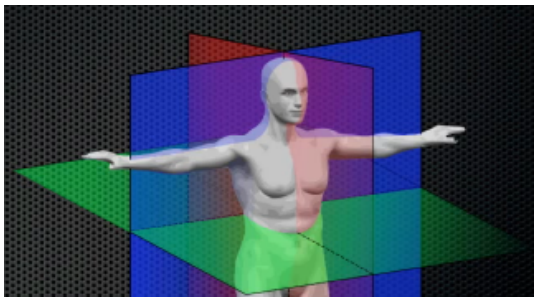


Fig. 5. Motion feature.

program that consists in sampling data .We suppose that f is a set of feature vectors composed by N descriptor vectors $(f_1, f_2, \ldots, f_N)$, where N is the initial number of frames. This algorithm consists in sampling f into a fixed frames number T.

Algorithm 2 Sampling algorithm

procedure SAMPLING(f, T)
  $f \leftarrow (f_1, f_2, \ldots, f_N)$
  $d = \sqrt{\sum_{i=1}^{N}(f_i - f_{i+1})^2}$ ▷ *d is the distance between two successive feature vectors*
  $f_i \leftarrow$ the feature vector of the current frame
  $f_{i+1} \leftarrow$ the feature vector of the next frame.
  $T \leftarrow$ number of frames
  if $d \leq \epsilon$ **then**
    Delete f_{i+1}
  else
    Compute the average distance: d'= $\frac{N'}{T}$ ▷ *N' is the number of featurevectors after removing noises.*
    Get points after every d' feature vectors: $f_{i+1} = f_i + d'$.
  end if
end procedure

We sample our data into ,35 frames. We use 8 body features , 15 motion features, and 2 shape features for a 25-dimensional feature vector at each frame. We use 35 frames, obtaining $d = 875$.

4) Recognition phase:

a) Random Forest Classifier: The random forest classifier consists of an ensemble of tree classifiers where each tree is grown randomly. So instead of using all features, RF randomly selects a subset of features to split at each node when growing a tree.

1) Training step:
Bagging phase: For a standard training set S of size m, bagging generates k new training data sets s_i each of size m', with $m' \leq m$ by sampling from S uniformely and with replacement which will learn k trees.
Growing phase: The learning phase of a decision tree consists in constructing its hierarchical structure. At each stage of the construction it is a question of adding new nodes to the structure, and thus has grown the decision tree. Thus, from the root the "nodes" are subdivided progressively into child nodes. The separation in two of the distribution of the samples for a node is made according to a comparison between a Component of the descriptor vector and an associated cut value determined randomly over a range of variation between Minimum and maximum values taken by the component on the whole corpus. To decide to split a node in two or more sub-nodes, it is first important to choose an appropriate node splitting criterion so as to maximise homogeneity of the offspring nodes. In the experimental result, two criterion were used to measure the quality of a split, the information gain based on the concept of entropy from information theory and the Gini impurity [3]. And we found a better classification result with the gini criterion.

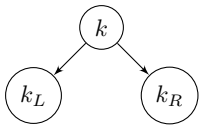


Fig. 6. Left-Right Banded model with nine states.

This Figure 6 shows a hypothetical situation that node k is split into two nodes k_l and k_r according to an attribute X. In the figure $n(k)$, $n(k_l)$ and $n(k_r)$ respectively denote the numbers of learning samples in nodes k, k_l, and k_r.
Gini index: $I(k) = 1 - \sum_{j=1}^{l} \frac{n_j(k)^2}{n(k)^2}$
$n_j(k)$ denotes the number of learning samples which belong to class j at node k.
The amount of homogeneity gain achieved by splitting node k can be then evaluated by:
$G(k) = I(k) - [\frac{n(k_l)}{n} I(k_l) + \frac{n(k_r)}{n} I(k_r)]$

So a low Gini coefficient indicates more equal distributions, a high Gini coefficient shows unequal distribution.

2) Classification step:
To classify a new feature vector, we put it down each of the trees in the forest was fed into the decision tree, it follows a track through the tree determined by the values of its components, and the nodes'cut values, until it will end up in a leave node. Each tree votes for that class and the classification having the most votes over all the trees in the forest is chosen.

b) Multi-layer Perceptron: Multi-layer Percpetron is the most known and most frequently used type of neural network , they are able to cope with non-linearly separable problems. An MLP is composed of a large number of highly interconnected neurones working in unison to solve specific problem. The architecture of MLP consists of three or more layers. Input layer is the first layer which its number of neurons is equal to the number of selected specific features. Output layer is the last layer which determines the desired classes. And

hidden layers to increase the ability of the network. Unlike the input and output layers, we have no prior knowledge of the number of neurons needed in the hidden layer. So, determining their numbers is one of the most critical tasks in a neural network design. Each neuron has a set of input with random weights and bias input. All inputs are multiplied by the associated weights and summed then passed through an activation function to the output layer, yielding

$(o^l)_j = f_j^l((b_j^l + \sum_{i=1}^{n_{l-1}} w_{j,i}^l (o^{l-1})_i)$,

where o^l is the output vector, $(o^{l-1})_i$ are n_{l-1} inputs, $w_{j,i}^l$ is the weight of the i^{th} input in the j^{th} neuron, f_j^l is the activation function and b_j^l the bias of the j^{th} neuron. Once the network weights and biases have been initialized, the network is ready for training. The training process requires a set of network inputs and target outputs. During training the weights and biases of the network are iteratively adjusted to minimize the error between the network outputs and the target outputs using descent which is determined using Backpropagation [2]. For classification, it minimizes the Cross-Entropy loss function [17], giving a vector of probability estimates $P(y|x)$ per sample x where y is the target and x is a set of features.

c) SVM classifier: SVM is basically a two-class classifier, based upon the idea of constructing a hyperplan which seperates a set of examples with a maximum margin. Given a training data set of the form (x_i, y_i), where $x_i \in R^n$ is the i^th example and $y_i \in 1, \ldots, K$ is the i^th class label. There exists a hyperplane of the form $w^T x + b = 0$ separating the positive from the negative training examples, where w is the normal to the hyperplane and b is the perpendicular distance of the hyperplane to the origin. The optimal hyperplan is constructed by solving an optimization problem:

$$Min\ \frac{1}{2} w^T w subjectto\ y_i(w^T x_i + b) \geq 1 \tag{1}$$

whose dual problem is:

$$Max\ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{j,k=1}^{n} \alpha_j \alpha_k y_j y_k (x_j^T x_k^T)\ subjectto\ \sum_{i=1}^{n} \alpha_i y_i = 0, \alpha_i \geq 0, i = 1 \ldots n \tag{2}$$

For the linearly non-separable case, the minimization problem needs to be modified to allow the misclassified data points. Therefore a relaxation variable $\epsilon_i \geq 0$ is introduced with a mapping function $\phi(\Delta)$ which projects the training data x_i into a higher dimensional space so as to allow for nonlinear decision surfaces. The optimization problem can be presented as:

$$Min \frac{1}{2} w^T w + C \sum_{i=1}^{n} \epsilon_i subjecttoy_i(w^T \phi(x_i) + b) \geq 1 - \epsilon_i i = 1, \ldots, n \tag{3}$$

whose dual problem using the Lagrange multipliers is:

$$Max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{j,k=1}^{n} \alpha_j \alpha_k y_j y_k (\phi(x_j)^T \phi(x_k)) subjectto \sum_{i=1}^{n} \alpha_i y_i = 0, 0 \leq \alpha_i \leq C, i = 1 \ldots n \tag{4}$$

$(\phi(x_j)^T \phi(x_k)) = K(x_j, x_k)$ is a kernel function associated with this mapping.

The performance of an SVM classifier is dependent on the choice of this function. Different kernel functions have been employed for different classification tasks. In our case we employed four kernels functions (sigmoid, radial basis function, Mahalanobis and polynomial) for gesture classification and we compared their performance, We obtained the most recognition rate with the polynomial kernel

$K(x_j, x_k) = (\gamma * x_j^T x_k + 1 + c_0)^d$,Degree (d) is the main parameter here, but you can also vary γ and the coefficient c_0 to make the kernel non-symmetric.

In the classification step, the classifier producing the maximum ouput is considered the winner, and this class label is assigned to that example.

d) Multi class SVM classifier: For multi-class classification, two extended method are proposed to classify our gestures. The first one is the one against all method which was introduced by [5]. It constructs K separate binary classifiers for k class classification where The k^th SVM is trained with positive examples belonging to class k and negative examples belonging to the the remaining $K-1$ classes. During test, the class label is determined by the classifier that produces the maximum output.

The second method is the one against one-against-one, this method constructs $K(K-1)/2$ binary classifiers where each one is trained on data from two classes. Applying each classifier to a test example will performed a voting among the classifiers and the class with the most votes wins.

III. EXPERIMENTAL RESULTS

a) MSRC-12 Dataset: The MSRC-12 gesture dataset [8] was recorded using a Kinect sensor at $30Hz$. The dataset comprises 12 gesture classes, divided into two categories iconic gesture category and metaphoric gesture category. The dataset has 594 sequences, collected from 30 people performing 12 gestures. For every sequence, one subject performs an action several times. In every frame, 3D locations of twenty body joints are estimated. different types of instructions are given to persons such as (text, image, video and their combinations). In our approach, we are interested in the recognition of the upper body part, so we work only with on the metaphoric gestures since in the latter only the upper part of the skeleton moves. So we will take into consideration the six following gestures: start music / raise volume, navigate to next menu, wind up the music, take a bow to end music session, protest the music, and move up the tempo of the song).

b) Proctocol evaluation: We have evaluated our system with the first data base MSRC-12, we follow the same strategy as [[8]]. we sampled our data into 35 frames, so we obtain a feature vector with dimension $35 * 25 = 875$. In the five-fold cross-validation, the dataset is randomly divided into five uniform subsets.The cross-validation process is then repeated 5 times, at each time four subsets are used for training and

one subset for testing. Finally the average performance is calculated using F-score defined as:

$F - score = 2 * \frac{precision * recall}{precision + recall}$.

$precision = \frac{T_p}{T_p + F_p}$; $recall = \frac{T_p}{T_p + F_n}$;

T_p is the number of true positives and F_n is the number of false negatives. For each learning methods we tried to adjust its parameters to obtain best results:

c) Random Forest Classifier: We started with the first method the Random forest Classifier, the differents parameters which need to be adjusted are:

- Number of trees: $10^1, 10^2, 10^3$.
- Number of features among n descriptor values to consider for best split: n, $\sqrt{n}$, $\log_2 n$.
- Criterion: gini, entropy In table 1 we found the best parameters that make the highest average F-score.

Table 1. Best parameters combinations for Random Forest Classifier method

Parameters	Number of trees	max features	Criterion
Values	100	$\sqrt{n}$	Gini

d) Multi-layer perceptron: For the Multi-layer perceptron four important parameters to adjust, the number of hidden layer, the activation function and two parameters for bakcpropagation, learning rate and momentum (Tab. 2).

- Hidden layer size (S): $10^1, 10^2, 10^3$.
- Activation function:
  logistic sigmoid function $f(x) = \frac{1}{1+\exp(x)}$
  hyperbolic tan function $f(x) = tanh(x)$
  Rectified linear unit function $f(x) = \max(0, x)$
- Learning rate (ϵ):It controls the step-size in updating the weights, it should be bewteen 0 and 1.
- Momentum(m): To speed convergence network while avoiding instability, it should be bewteen 0 and 1.

Table 2. Best parameters combinations for Multi-layer Perceptron method

Parameters	S	Activation function	ϵ	m
Values	100	Relu	0.001	0.9

e) Multi class SVM classifier: In the SVM method we choose the kernel function and its parameters and the penality parameter (Tab. 3)

- kernel function: linear, polynomial, rbf, sigmoid.
- parameters kernel function(polynomial): Degree (d), γ and Kernel coefficient a_0.
- Penalty parameter C.

Table 3. Best parameters combinations for SVM method

Parameters	kernel function	d	γ	a_0	C
Values	polynomial	3	1	1.0	1

A comparaison between four classifier (Random Forest(RF), Multi-layer Perceptron (MLP), One against All (OAO), One Againt One (OAO)) is presented in Table. Here we measure the intramodality generalization performance: training and testing using the same instruction modality. We obtained a higher average F-score in almost all methods. Under the same conditions as the [8], means we use number of frames 35, and a random forest as a classifier we can confirm our successeful approach with the comparaison in table .

Table 4. Mean and standard deviation F_score using all instructions (T for Text, I for Image, V for Video) for the metaphoric gestures

Gesture	RF	MLP	OAA	OAO
T	0.93 (+/- 0.05)	0.82 (+/- 0.25)	0.92 (+/- 0.11)	0.93 (+/- 0.08)
I	0.98 (+/- 0.06)	0.92 (+/- 0.09)	0.99 (+/- 0.03)	0.99 (+/- 0.02)
V	0.99 (+/- 0.02)	0.90 (+/- 0.13)	0.97 (+/- 0.03)	0.99 (+/- 0.02)
IT	0.98 (+/- 0.02)	0.94 (+/- 0.14)	0.98 (+/- 0.04)	1.00 (+/- 0.01)
VT	0.99 (+/- 0.01)	0.98 (+/- 0.02)	0.99 (+/- 0.01)	0.99 (+/- 0.01) heigh

Table 5. Comparison with MSRC-12 metaphoric gestures with the five modalities

Gesture Video+Text	Text	Image	Video	I
Our approach 0.99 (+/- 0.01)	0.93 (+/- 0.05)	0.98 (+/- 0.06)	0.99 (+/- 0.02)	0.9
Baseline [8] 0.698 (+/- 0.092)	0.621 (+/- 0.059)	0.441 (+/- 0.069)	0.552 (+/- 0.025)	0.57

f) MSRAction3D Dataset: MSRAction3D [12] is a public dataset which has been collected using a a depth sensor similar to the Kinect device at 15 fps. It includes 20 actions performed by 10 persons Each action is performed three or two times.This dataset is divided into three action, we intereted in ten actions since our subject concerns the recognition of the upper body part so we evaluated our approach with the following gestures: High arm wave, Hand catch, Draw cross, Draw Tick, Draw circle, Two-hand wave, Forward punch, Horizontal arm wave, Hammer and Hand clap.

g) Florence3D dataset[?]:: It includes 9 different activities: wave, drink from a bottle, answer phone, clap, tight lace, sit down, stand up, read watch, and bow.These activities were performed by 10 different subjects, for 2 or 3 times, resulting in a total number of 215 sequences. This is a challenging dataset, since the same action is performed with both hands and because of the presence of very similar actions such as drink from a bottle and answer phone. The activities were recorded in different environments, and only RGB videos and 15 skeleton joints are available.

IV. Conclusion

In this paper, we have proposed an effective approach to recognizing upper body human gestures is proposed started by precprocessing step to normalize and sample skeleton data . After that we choose a robust descriptor vector to present gestures using a combination of three principal components.

The first component used to define the relation between differents body parts, the second to describe changing shape during gesture and the final component is used to make motion directions of gestures in 3D space. For gesture recognition a comparaison between three known methods used for training and gestures classfication. The results Experiments on two databases MSRC-12 and MSRAction3D show that our method performs better than many existing skeletal representations of the state of the art in most cases. As a perspective, we intend to create our own database specific to teleoperate humanoid robot (NAO) with upper body gestures after having successfully selected a robust feature vector .

REFERENCES

[1] R. Azad and F. Davami. A robust and adaptable method for face detection based on color probabilistic estimation technique. *CoRR*, abs/1407.6318, 2014.

[2] C. M. Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[3] L. Breiman. *Classification and regression trees*. Wadsworth International Group, Belmont, Calif, 1984.

[4] G. Canal, C. Angulo, and S. Escalera. Gesture based human multi-robot interaction. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2015.

[5] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[6] K. M. Deo, A. Y. Kulkarni, and T. S. 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.

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

[8] S. Fothergill, H. Mentis, P. Kohli, and S. Nowozin. Instructing people for training gestural interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1737–1746, New York, NY, USA, 2012. ACM.

[9] M. E. Hussein, M. Torki, M. A. Gowayyed, and M. El-Saban. Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, pages 2466–2472. AAAI Press, 2013.

[10] R. R. Igorevich, P. Park, J. Choi, and D. Min. Two hand gesture recognition using stereo camera. *International Journal of Computer & Electrical Engineering*, 5(1), February 2013.

[11] M. B. Kaâniche and F. Brémond. Recognizing gestures by learning local motion signatures of hog descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2247–2258, Nov 2012.

[12] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, pages 9–14, June 2010.

[13] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *British Machnine Vision Conference (BMVC 2011)*, volume 1. BMVA Press, 2011.

[14] D. Oneata, J. Verbeek, and C. Schmid. Action and Event Recognition with Fisher Vectors on a Compact Feature Set. In *ICCV 2013 - IEEE International Conference on Computer Vision*, pages 1817–1824, Sydney, Australia, Dec. 2013. IEEE.

[15] M. Pierobon, M. Marcon, A. Sarti, and S. Tubaro. 3-d body posture tracking for human action template matching. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 2, pages II–II, May 2006.

[16] K. Qian, J. Niu, and H. Yang. Developing a gesture based remote human-robot interaction system using kinect.

[17] M. D. Richard and R. P. Lippmann. Neural network classifiers estimate bayesian a posteriori probabilities. *Neural computation*, 3(4):461–483, 1991.

[18] M. Zanfir, M. Leordeanu, and C. Sminchisescu. The moving pose: An efficient 3d kinematics descriptor for low-latency action recognition and detection. In *2013 IEEE International Conference on Computer Vision*, pages 2752–2759, Dec 2013.