A Novel Approach to Hand-Gesture Recognition in a Human-Robot Dialog System

Pujan Ziaie, Thomas Müller and Alois Knoll
Robotics and Embedded Systems Group
Department of Informatics
Technische Universität München
e-mail: {ziaie,muelleth,knoll}@cs.tum.edu

***Abstract*— In this paper, a reliable, fast and robust approach for static hand gesture recognition in the domain of a Human-Robot interaction system is presented. The method is based on computing the likelihood of different existing gesture-types and assigning a probability to every type by using Bayesian inference rules. For this purpose, two classes of geometrical invariants has been defined and the gesture likelihoods of both of the invariant-classes are estimated by means of a modified K-Nearest Neighbors classifier. One of the invariant-classes consists of the well-known Hu moments and the other one encompasses five defined geometrical attributes that are transformation, rotation and scale invariant, which are obtained from the outer-contour of a hand. Given the experimental results of this approach in the domain of the Joint-Action Science and Technology (JAST) project, it appears to have a very considerable performance of more than 95% correct classification results on average for three types of gestures (pointing, grasping and holding-out) under various lighting conditions and hand poses.**

***Keywords*— Image Processing, Gesture Recognition, Bayes Theory, K-Nearest-Neighbors, Hu Moments.**

I. Introduction

The interaction between human and robot is definitely one of the major issues in the 21st century. This is due to the fact that although nowadays many tasks are being performed merely by robots, however, there are many cases in which robots either need the supervision and direction of a human-being or they require collaboration with people to receive and process corresponding data to start a transaction or finish an assignment.

In some fields the interaction with humans is inevitable. In entertainment, for instance, a good understanding of what people want is important. Imagine a robot which is serving the people at a bar as bar-tender. This robot needs to communicate with people to see what their demands are and then carry out the corresponding task. Another example would be in bomb detection where the supervision of an expert is needed to reduce the risk. Trying to address these needs, new methods nave been sought to ease the process of communication. Not every customer at a bar or every specialist needs to know how to program a robot and insert the right instructions! Thus, a natural way of interaction should be constructed so that the robot can obtain the relevant data from the surrounding people. Human-Robot Interaction or HRI addresses this need.

Regarding a normal relationship between two (or more) people, they talk to each other, use gestures by means of parts and poses of their body, use the tone of their voice for stressing an issue or even make body-contacts like hand-shaking or patting on each other's back.

Gestures are, in fact, used for everything from pointing at a person to conveying specific information or implying a message. Researches indicate that gesturing does not only embellish spoken language, but is an essential part of the language generation process [9]. It happens very often that one cannot simply express his or her feelings or opinions without using additional gestures. Hand gestures, among other necessary domains in HRI, play an important role, both as an accompaniment to speech and as a means of input in their own right. This paper focuses on the task of static *hand-gesture*

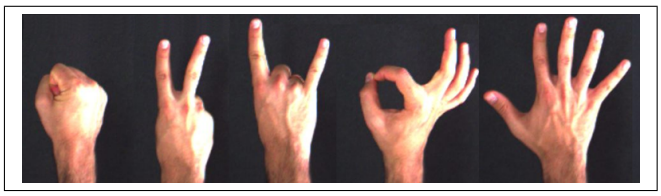

Fig. 1. Various static gestuers of a hand

recognition, viz., recognizing and classifying different hand shapes of a human user. A static gesture means that the style of the movement is of no value and all that matters is the shape and orientation. Some static gestures are depicted in figure 1. The whole process is in the context of a cooperative human-robot assembly task.
Considering this objective, an approach is introduced that operates based on classifying the gestures using two different classes of invariants and then select the most likely gesture-type by means of Bayesian inference rules.

In a nutshell, a Locally Weighted Naïve Bayes (LWNB) classifier, is used for two different feature vectors, one based on some defined geometrical invariants and the other on Hu moments. The output for each of these vectors is considered as an uncertain suggestion with an estimated likelihood resulting from the classifier. These priors are then combineds based on Bayes inference rules. The certainty-factor according each classification result is computed according to the performance of each of the invariant-classes. The approach is thoroughly expounded in IV-E.

In this paper, first the JAST project will be discussed which

helps us determine our application domain, restrain the expectations and understand the requirements. Then some related works in the area of gesture recognition and similar approaches to the sub-tasks of image segmentation and classification are listed. Afterwards, our gesture recognition approach for the JAST human-robot dialog system will be introduced in great length and finally, the experimental results and conclusion will be provided.

II. JAST

A. Project Definition

JAST stands for ***J**oint-**A**ction **S**cience and **T**echnology*. The projects aims to address the need for a robot acting in a cognitive environment, in which different modules collaborate coherently with each other to achieve a certain goal and successfully perform a specific task. The overall goal is to investigate the cognitive and communicative aspects of jointly-acting agents, both human and artificial. In JAST, an autonomous robot (see figure 2) which consists of different modules, communicates with a human to assemble a wooden construction. This means putting and installing different pieces of Lego®-like material to build up a wooden airplane for instance. For further information, one can visit JAST's official website[1].

Fig. 2. The JAST robot, environment and gestuers

To accomplish this task, the robot should be able to recognize (image processing), pick-up and handle objects (robotics) in different shapes and sizes and also communicate with a human being. To communicate in a natural way the robot needs to be capable of listening and talking (speech processing), recognizing gestures and position of its partner (gesture and face recognition) and expressing its feeling through facial mimics.

The input channels of the system consist of speech recognition, object and gesture recognition, robot sensors, and face tracking; synthesized speech, head movements and face gestures, and robot's arms actions are the system outputs. Based on these channel and the final goal, six input-output modules are defined, some acting as an autonomous agent and some as reasoning or coordinating components: speech recognition, object recognition, head tracking, robot (arms of the robot), iCat (head and face of the robot) and gesture recognition.

B. Recognizing Gestures

Object and gesture recognition in JAST are both performed on the output of a single camera which is installed directly above the table looking downward to take images of the scene. The output of this process is sent to a multi-modal fusion component [3], where it is combined with any spoken input from the user to produce combined hypotheses representing the user's requests. Three different gestures are required to be distinguished in the context of gesture recognition (as depicted in figure 2).

Pointing Gestures are those shapes of one's hand, which are used to point at an object. Obviously the index finger will be used by either the right or left hand. **Grasping Gestures** are used to demonstrate the action of taking something. This is usually combined with the dialog stating "I am going to pick up this object". The index finger along with the thumb are used for this purpose. **Holding-out Gestures** are a sign of asking for something, particularly an object.

III. Related Works

There exist numerous methods that have been developed recently to perform a successful gesture recognition. Most of these systems use model-based approaches, whereas some of them exploit invariant-classification methods. The invariant-based approaches consist of two main steps: Extraction of invariants and classification of gestures based on those invariants.

A. Segmentation & Extracting Invariants

Invariants are shape descriptors extracted from an image that are independent of the viewpoint [13]. Using invariants for recognition greatly simplifies the process of object recognition because it allows objects to be compared with reference models regardless of the orientation. Obviously, before extracting invariants, it is necessary to segment the recognized image to extract the relevant objects or regions of interest and to omit the irrelevant data.

For hand-gesture recognition, some researchers have tried to perform the early segmentation process using skin-color histograms [16], [5]. The problem with this approach is that

[1]Please visit http://www.euprojects-jast.net/

they do not operate well in cases when there are some other objects in the scene with the same color as skin color, or where the hand has other colors than the predefined one. In the target JAST application, the background is static and can easily be eliminated and therefore, the concentration can be mainly on the geometric characteristics of the objects.

Zhou *et al.* [16] used overlapping sub-windows to extract invariants for gesture recognition , and characterized them with a local orientation histogram feature description indicating the distance from the canonical orientation. This makes the process relatively robust to noise, however, much more time-consuming indeed. Kuno and Shirai [7] defined seven invariants to do hand gesture recognition, including the position of the fingertip. This is not practical when we have not only pointing gestures, but also several other gestures, like grasping. However, the invariants they considered inspired us for our defined invariants.

Normalized Zernike moments [15] of an image can also be used as effective invariants. In some similar approaches, the watermark of an image is generated by modifying the invariant-vector. For example, Lizhong Gu and Jianbo Su [4] tried to use Zernike moments along with a hierarchical classifier to classify hand-gestures. This method is not appropriate for the JAST project, since there is not a high degree of freedom for the hands due to the limited space for movements and actions.

B. Classification

Classification is a method to assign a class to a point (or vector in spaces of more than one dimensions) in an N-dimensional space. The classes may be predefined and learned beforehand (supervised learning), or may be extracted automatically based on a similarity metric (unsupervised learning).

K-nearest neighbors (KNN) classifiers has a good performance when the attributes of a system are linearly separable. It finds the K nearest (already classified) vectors to the input vector. The class which most vectors in those K neighbors belong to is chosen to be the right class of the input vector. K-nearest neighbors with distance weighting (KNNDW) is an improvement which has been proved to perform better than KNN in many cases [10]. In this method, the contribution of each neighbor to the overall classification is weighted by its distance from the point being classified. The classes are then assigned with a likelihood value based on a simple naïve Bayes approach.

The most relevant work to our classification approach addressed in this paper has been performed by Frank *et al.* [2] which introduces a Locally Weighted Naïve Bayes (LWNB) classifier. Their evaluation shows that LWNB outperforms KNN and KNNDW when K is big enough.

C. Bayesian Inference

Different classification results can be combined optimally based on Bayesian inference. Availing Bayesian theory in decision making has been used in many fields and applications like market prediction [1], motion detection [14] or advisory systems [11]. Making the final decision can be optimally performed, when there are several suggestion involved, all of which based on uncertain hypotheses. Oliver *et al.* [12] have shown how one feature vector (observation) can be fed into two different classifiers and in what way a Bayesian approach can be exploited to combine the results.

IV. Gesture Recognition Approach

A. Pre-processing

The background subtraction is carried out by applying an adaptive threshold in order to differentiate object-pixels from the background-pixels of the images delivered by the camera. A static or dynamic threshold can be used to perform this task, as described in [8]. In the domain of the JAST project, a minimum and a maximum threshold based on evaluation of a multi-dimensional color histogram are defined to extract the binary image (as demonstrated in Figure 3). For the second step, the already classified pixels are grouped into blobs, which are bounded segments that eventually turn into regions of interest (ROI).

The grouping procedure is performed by connecting neighboring pixels by means of a recursive algorithm. After having performed the grouping process, a bounding-box is assigned that defines the borders of each group (or segment) which constructs our ROI entities. These ROI entities are then sent to object- or gesture-recognition modules to be processed.

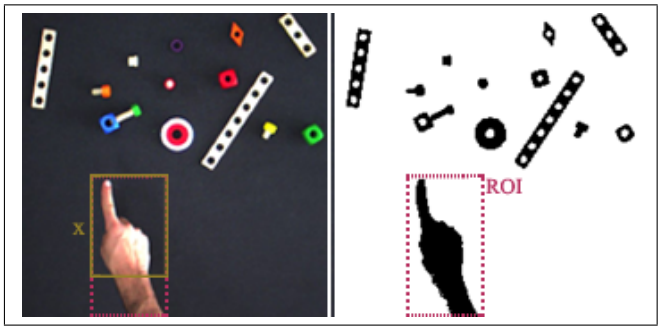

Fig. 3. An ROI circumscribing a gesture and the region to process

Since the user's hand is always entering the scene from the bottom part of the table, obviously merely those ROIs which end up at that particular position might contain a gesture. These eligible ROIs will be sent to the gesture recognition module.

B. Extracting Invariants

1) Definition: To increase the performance, two sets -or classes- of invariants are used in unison[2]. One set contains invariants that are specifically defined for the gesture recognition module of the JAST project and are called **defined**

[2]All the operations from now on are performed on the ROI-image received from the pre-processing module.

invariants from now on. The other set consists of the first six Hu moments.

2) Defined Invariants: Once the regions of interest (ROIs) have been identified as described in the preceding section, the next step is to extract some meaningful geometric invariants from the binary image to be used for the classification.

The set of invariants is defined as below. Note that all the attributes have been defined in a way that makes them transformation, rotation and scale invariant.

1) **Length of outer-contour**
   Normalized by division by the square root of the area.
2) **X-gradients**
   The number of direction-changes in x (first dimension) direction when exploring the outer-contour.
3) **Y-gradients**
   Same as above, for changes in y direction.
4) **Gradients-deviation**
   Normalized deviation of gradient points divided by the square root of the *area*.
5) **Furthest distance**
   Normalized distance of the furthest gradient point from the center of the gesture divided by the square root of the *area*.

Gradients are in fact semi-rotation invariant, since their value would be replaced by 90 degree rotation.

3) Hu Invariants: Hu moments [6], are scale, translation and rotation invariant. Hu derived these expressions from algebraic invariants applied to the moment generating function under a rotation transformation. In this work, only the first six moments are used, because the seventh moment, which is the skew-invariant one, appears to add no values to the recognition results.

4) Extraction Approach: Considering *defined invariants*, the gradients are first computed from the extracted outer-contour of the hand and their geometrical attributes (the last two invariants) are derived from their locations.
Since the shape and length of one's arm (viz., how deep it is in the scene) is immaterial, to extract the invariants, only a specific and predefined area of the upper part of the ROI (which is supposed to be the hand) is processed. The cropping distance, x is estimated due to the average size of a hand. This estimation can be availed, since the distance of the camera from the table is constant.This cropped area is depicted in Figure 3.

Supposing that there are M defined invariants and L Hu-moments, we have two vectors with M and L dimensions, which are five for the defined invariants and six for the Hu-moments in our experiment.

$$\begin{aligned} df\vec{I}nv &= \{a_1, a_2, \ldots, a_M\} \\ hu\vec{I}nv &= \{h_1, h_2, \ldots, h_L\} \\ L &\in \{1, \ldots, 7\} \end{aligned} \tag{1}$$

These invariant-vectors are added to the training pool together with their corresponding type of gesture, if the system is in its training phase (Section IV-C), or will be fed into the classifier to find the likelihood of each of the gesture-types (Section IV-D).

C. Training the Classifier(s)

Obviously, before performing the classification, a training pool should be created for each of the invariant-classes. It is recommended that the data is produced by different users under various lighting conditions in order to increase the robustness. Each training instance is labeled with its corresponding gesture type. The gesture types are defined as:

$$\vec{C} = \{c_1, c_2, \ldots, c_Z\} \tag{2}$$

where Z is the total number of gesture types which is *three* in this application. Extending the gestures can be simply done by adding the corresponding training data to the pools. Assuming there are N vectors in the training pool (meaning that we have N samples), each vector is defined as:

$$Inv_n^f(m) = \{df_0, df_1, \ldots, df_M\} \tag{3}$$

$$Inv_n^h(l) = \{hu_0, hu_1, \ldots, hu_L\} \tag{4}$$

$$\text{with:} \quad df_0, hu_0 \subseteq \vec{C},$$
$$m \in \{1 \ldots M\} \wedge l \in \{1 \ldots L\} \wedge n \in \{1 \ldots N\},$$

where Inv^f is a defined invariant-vector and Inv^h represents the vector of the Hu moments and the first element of each of them (df_0 and hu_0) represents the label of their class.

After constructing the two pools for labeled vectors, the classification can proceed.

D. Classification: LWNB

The classification algorithm is basically the same for both invariant-sets. Therefore, the general algorithm is discussed here, which will be applied to both *defined* and *Hu* invariant vectors and the recognition will result from both outputs based on Bayes inference rules (see IV-E).

In our application, the well-known K-nearest neighbors algorithm is used as our classifier, with two modifications. First, before performing the classification, the elements of the given vector (the invariants) are weighted based on their influence on the process. The proper weights have been extracted off-line based on empirical findings.

The second modification is performed after finding the K nearest neighbors. Instead of simply calculating the distance of each vector in the space and choose the number of the found vectors among the first K nearest ones, for each (training) vector (node) of a class, a weight is assigned to that node based on its distance to the input vector. The probability of a class is then based on the weights of that class in the first K neighbors.

According to (5), *defined* invariant-vector has M dimension and *Hu* invariant-vector has L dimensions respectively. We

have defined the distance-weighting vector for each of the invariant-classes as:

$$\vec{wdf} = \{wdf_1, wdf_2, \ldots, wdf_M\}$$
$$\vec{whu} = \{whu_1, whu_2, \ldots, whu_L\} \quad (5)$$

Consequently, the distance between the two input invariant-vectors in^f and in^h and the nth training-node of their corresponding training pool can be computed in Euclidean space as:

$$dist(Inv_n^f, in^f) = \sqrt{\sum_{m=1}^{M} \frac{\left(Inv_n^f(m) - in^f(m)\right)^2}{wdf_m}}$$
$$dist(Inv_n^h, in^h) = \sqrt{\sum_{l=1}^{L} \frac{\left(Inv_n^h(l) - in^h(l)\right)^2}{whu_l}} \quad (6)$$

The distance is indeed normalized, so that all the values be between 0 and 1.

In the next step, K^f defined invariant vectors and K^h Hu invariant vectors with the shortest distance from their respective input vectors are selected from the training pools ($\vec{Inv^f}_n$ and $\vec{Inv^h}_n$) for both invariant-classes. These selected vectors are called $\vec{sInv^f}_n$ and $\vec{sInv^h}_n$ respectively.

$$\vec{sInv^f}_{kf} = \{\vec{sInv^f}_1, \ldots, \vec{sInv^f}_{K^f}\} \quad (7)$$
$$\vec{sInv^f}_{kf} \in \{\vec{Inv^f}_1 \ldots \vec{Inv^f}_N\}$$
$$\vec{sInv^h}_{kh} = \{\vec{sInv^h}_1, \ldots, \vec{sInv^h}_{K^h}\} \quad (8)$$
$$\vec{sInv^h}_{kh} \in \{\vec{Inv^h}_1 \ldots \vec{Inv^h}_N\}$$
$$\text{with:} \quad kf \in \{1, \ldots, K^f\} \wedge kh \in \{1, \ldots, K^h\}$$

The first elements of each of these vectors, $sInv_{kf}^f(1)$ and $sInv_{kh}^h(1)$ are, in fact, the label of the class they belong to (according to (5)). Hence having c^f and c^h as variables corresponding to classes as:

$$c^f(kf) = \{sInv_{kf}^f(1)\} \quad (9)$$
$$c^h(kh) = \{sInv_{kh}^h(1)\} \quad (10)$$

At this level, the naïve Bayes probability (*likelihood*) of each class can simply be computed:

$$p(C^f(z)) = \frac{\sum_{kf=1}^{K^f} \delta(C^f(z), c^f(kf))}{\sum_{y=1}^{K^f}} \quad (11)$$
$$p(C^h(z)) = \frac{\sum_{kh=1}^{K^h} \delta(C^h(z), c^h(kf))}{\sum_{y=1}^{K^h}} \quad (12)$$
$$z \in \{1 \ldots Z\} \quad (13)$$

where δ is defined as

$$\delta(u, v) = \begin{cases} 1 \text{ if } u = v \\ 0 \text{ otherwise} \end{cases} \quad (14)$$

and z represents the index of each class (implying the type of gestures).

The simple interpretation of this formula is that the likelihood of each class is the number of vectors which belong to that class among the K selected vectors, divided by the total number of K, viz., K^f or K^h depending on the invariant-class (*defined* or *Hu*). This likelihood, however, does not take the different distances of each node into account. This means that the number of class-nodes found is used for the probability, regardless of what the distance of each node from the input vector was.

To take advantage of the effect of distances and improve the results, we add weights to the selected nodes (neighbors). Considering each node as t, this weight $wB(t) = f(ds_t)$ is a function of the already computed Euclidean distance ds_x of each node and can be any monotonically decreasing function.

In this application, functions like $f(ds_t) = 1 - d_t$ or $f(ds_t) = (ds_t)^{-p}$ for various p were examined and the best function appeared to be:

$$wB(t) = f(ds_t) = \frac{1 - ds_t}{1 + ds_t} \quad (15)$$

Using these weights, a locally weighted naïve Bayes probability can be defined by weighting equations 12 and 13 as

$$p(C^f(z)) = \frac{\sum_{kf=1}^{K^f} wB(x_{kf})\delta(C^f(z), c^f(kf))}{\sum_{y=1}^{K^d} wB(y)} \quad (16)$$
$$p(C^h(z)) = \frac{\sum_{kh=1}^{K^h} wB(x_{kh})\delta(C^h(z), c^h(kf))}{\sum_{y=1}^{K^h} wB(y)} \quad (17)$$
$$z \in \{1 \ldots Z\}$$

The likelihood of each class is now available for both *defined* and *Hu* invariant-vectors. One possibility is to choose one of the methods and make a decision based on the result of its pertinent classifier.

$$c(\vec{Inv}) = \text{argmax}_{z=1,\ldots,Z} p(C(z)) \quad (18)$$
$$\vec{Inv} \subset \vec{huInv}, \vec{dfInv}$$

According to the experimental results, *Hu* invariant-vectors resulted in a maximum of 93 % of correct classification. This rate was 91 % for the defined invariants approach.

E. Recognition based on Bayes Decision Theory

1) Combination Possiblity: So far the likelihood of each class, based on the given vector for both *defined* and *Hu* invariants (moments), has been estimated.

To find the best solution, first, the correctness-rate of both approaches (*defined* and *Hu*) are extracted. Table IV-E.1 shows the results for two invariant-classes. The weighting vectors of the invariants have been chosen randomly and the results of expectation-rate do not vary much given different weights: According to Table IV-E.1, there is a low probability for each gesture to be falsely recognized by both classifiers. Therefore, we can take advantage of Bayes theory to increase the expectation values.

Expectation Value	Pointing	Grasping	Holding
Both Correct	0.7650	0.9600	0.9150
$DfInv$ Correct	0.0300	0.0600	0.3000
$HuInv$ Correct	0.5550	0.8250	0.0
Both False	0.2250	0.3150	0.0

Table 1. Classification results for both *Defined* and *Hu* approaches

2) Applying Bayes Inference to Classification Results: Bayes inference rules are of great avail for making the best decision regarding the type of the given gesture.

The certainty factors, or in other words, the *likelihood parameters* of each invariant-class, considering one sample gesture (*Pointing*), are shown in Table IV-E.2 (Table IV-E.1 shows the extracted values considering all gestures).

In this table, $p(df)$ is the likelihood of the gesture being correctly recognized using the defined invariants as the input vector to the classifier. The same is `true` for $p(hu)$. gs_x is a gesture with type x, which encompasses all z classes, with z being 3. The three different types of gestures are symbolized as gs_p, gs_g, gs_h, standing for pointing, grasping and holding-out gestures. In this table, however, only pointing gesture (gs_p) is addressed and the same thing applies to the other two class, viz., gs_g and gs_h.

Based on the premises addressed above, $p(df|gs_x)$ implies the likelihood of gesture x, when the given defined invariant-vector is classified as gesture x. Given one of the two methods,

Likelihood Measures	Pointing
Both Correct $p(df)p(hu)$	$p(df\|gs_p)p(hu\|gs_p)$
$DfInv$ Correct $p(df)p(\bar{hu})$	$p(df\|gs_p)p(\bar{hu}\|gs_p)$
$HuInv$ Correct $p(\bar{df})p(hu)$	$p(\bar{df}\|gs_p)p(hu\|gs_p)$
Both False $p(\bar{df})p(\bar{hu})$	$p(\bar{df}\|gs_p)p(\bar{hu}\|gs_p)$

Table 2. Likelihood parameters regarding pointing gestures gs_p

to compute the total likelihood of gestures (e.g. Hu invariants), the basic probability formula can be used, which in this case is

$$\begin{aligned} p(hu) &= p(hu|df) + p(hu|\bar{df}) \\ &= p(hu)\cdot(df) + p(hu)\cdot(\bar{df}), \end{aligned} \tag{19}$$

knowing that the result of classifier for *defined* invariants is independent of *Hu* invariants and vice versa. The likelihood of a gesture can then be approximated by:

$$\begin{aligned} &p(gs_x) = p(hu|gs_x)p(hu) + p(df|gs_x)p(df) \\ &\text{with: } x \in p, g, h \end{aligned} \tag{20}$$

This likelihood value is the final decision factor. Eventually, the gesture with the highest (maximum) likelihood will be adjudged the winner, viz, the most likely gesture.

$$\begin{aligned} &c = \text{argmax}_{x=1,\ldots,Z} p(C(x)) \\ &\text{in this application } Z = 3 \text{ hence } x \subset gs_p, gs_g, gs_h \end{aligned} \tag{21}$$

In sum, the steps toward recognition can be listed as follows:

1) Extract the *defined* and *Hu* invariants.
   Invariants are extracted and grouped into *defined* and *Hu* vectors: $\vec{dfInv}$ and $\vec{huInv}$.
2) Pinpoint the K nearest neighbors.
   The K nearest nodes in the training pool for both vectors are selected and weights are assigned to them disproportional to their distance.
3) Find the likelihood of each gesture for each vector.
   The likelihood of each gesture is computed based on the weights of the K selected nodes. The calculation is done by using a simple naive Bayes approach.
4) Select the most likely gesture via a Bayesian inference approach. The likelihoods of both vectors are fed into a Bayesian inference system and the most likely gesture is selected according to the certainty factor of each vector (invariant-class).

V. Experimental Results

Availing Bayesian inference rules to combine the results of our classifier for both invariant-classes always leads to a better performance, as shown in Table 3. The table shows the performance of each method for some random weighting vectors. Understandably, the overall performance increases

Def.	K^f	Hu	K^h	Comb.	K^c
87.62%	4	92.94%	6	94.00%	8
89.14%	13	92.94 %	5	95.01 %	7
92.54%	5	92.94 %	5	95.01 %	7
92.06%	5	93.46 %	5	95.01 %	7
91.04%	5	92.98 %	5	95.43 %	3
92.54%	7	93.46 %	5	95.43 %	2
90.50%	2	92.98 %	5	95.45 %	6
88.09%	2	92.98 %	5	95.45 %	2

Table 3. Performance results of *Defined*, *Hu* and combined invariants for different K

when the decision is made based on the Bayesian inference rules. It should also be noted that the results are more influenced by the likelihood table (refer to IV-E.2) than the performance of the classifier. By following the above mentioned approach, 95.45 % correct recognition occures on average. The performances of all three methods are depicted in diagrams of Figure 4 and Figure 5.

Presumably, one can increase the overall performance even more, by manipulating the likelihood values. This can be performed by means of algorithms like maximum likelihood estimation.

Running the full gesture-recognition process on a frame takes less than 25 msec on average (usually between 24-26 msec). Of this time, segmentation takes about 20 msec,

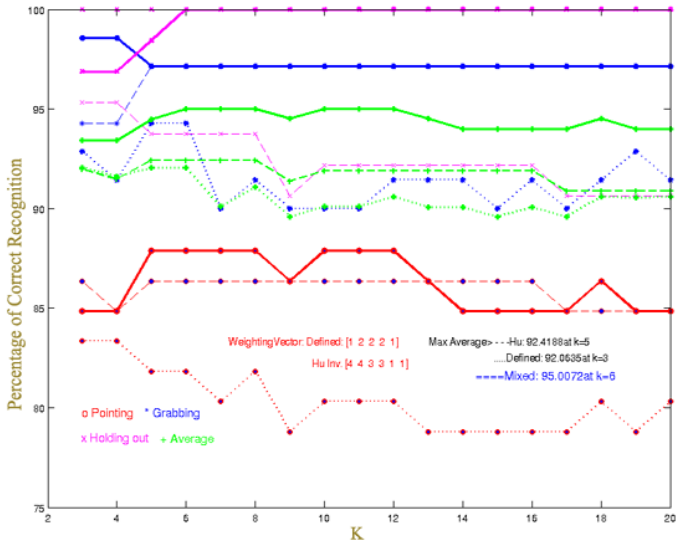

Fig. 4. Recognition Performance (95%)

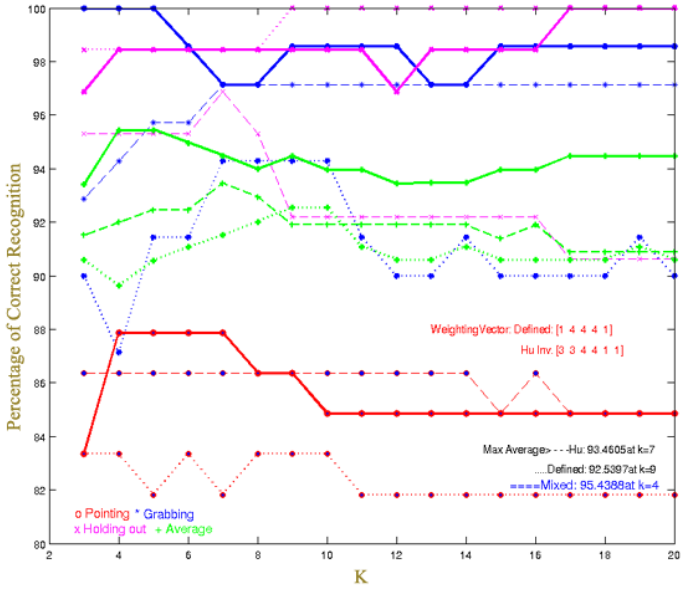

Fig. 5. Recognition Performance (95.45%)

while the (gesture) recognition process together with the autonomous segmentation module take approximately 4-6 msec.

VI. Conclusion

Using LWNB as classifier and combining its results for two invariants-class, viz, *defined* and *Hu* invariants, together with apt values for the parameters of the system, results in a performance of more than 95.0% of correct recogintion for three gestures.

To achieve these results, a testing pool with about 200 samples was constructed for all of the gestures (roughly 70 each), for a total of 500 samples in the training pool. The training data were made by five persons (three boys and two girls) in different lighting conditions. The testing data were created by two people other than those, whose gesture were represented in the training pool.

The results of each classifier given different weighting vectors as well as their combination by means of a Bayesian inference system were demonstrated for different values of Ks (K^d & K^h). The X axis of these graphs represents the value of the corresponding K for the K-nearest neighbor selection, while the Y axis shows the percentage of gesture instances of each type that were correctly recognized.

After trying various combinations of weights, LWNB classifier provides a maximum of 91.58% correct results with $wdf = [12211]$ as the weighting vector. This result is intuitively acceptable, as the range of invariants like contour length or deviation is much wider than the number of changes in gradients, hence, requiring higher weights for gradients in both directions.

Selecting Hu-moments as an input vector, a performance of 93.46% correct classification was achieved, with a weighting vector of $whu = [443411]$ which is again intuitively reasonable, putting a higher weight on the first four moments.

After constructing a recognition system based on the combination of results by means of a Bayesian decision making system, a performance of 95.45% correct classification was achieved at $K^c = 6$ with $wdf = [14423]$ and $whu = [223311]$. The likelihood values were calculated under $K^d = 2$ and $K^h = 5$ respectively. Other combinations would also lead to the same performance according to the values provided in table 3. The results can be viewed in figure 5. It should be noted that to obtain the likelihood values, first both invariant-classes are given to the classifer and the values are extracted based on the performance at this pre-recognition stage. According to the graphical demonstration of the performance, not only the correct recognition rate increased compared to single classifications, but also there were less fluctuations in the results.

It is essential to keep in mind that the likelihood values availed during the Bayesian inference are of high importance, and presumably, an even better performance can be attained by manipulating likelihood values by using methods like Maximum Likelihood Estimation. This will be further investigated in future works. Extending the application for recognizing more gestures as well as modifying the process to operate in a 3D environment are other plans of us for the future.

VII. Acknowledgement

This work was supported by the EU FP 6 Cognitive Systems Integrating Project *Joint-Action Science and Technology* JAST (FP6-003747-IP) – www.europrojects-jast.net.

References

[1] Yiling Chen, Chao-Hsien Chu, Tracy Mullen, and David M. Pennock. Information markets vs. opinion pools: an empirical comparison. In *EC '05: Proceedings of the 6th ACM conference on Electronic commerce*, pages 58–67, New York, NY, USA, 2005. ACM.

[2] Eibe Frank, Mark Hall, and Bernhard Pfahringer. Locally weighted naive bayes. In *Proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 249–25, San Francisco, CA, 2003. Morgan Kaufmann.

[3] Manuel Giuliani and Alois Knoll. Integrating multimodal cues using grammar based models. In Constantine Stephanidis, editor, *HCI (6)*, volume 4555 of *Lecture Notes in Computer Science*, pages 858–867. Springer, 2007.

[4] Lizhong Gu and Jianbo Su. Natural hand posture recognition based on zernike moments and hierarchical classifier. In *IEEE International Conference on Robotics and Automation*, pages 3088–3093, Pasadena, CA, USA, May 2008.

[5] H. Hongo, M. Ohya, M. Yasumoto, and K. Yamamoto. Face and hand gesture recognition for human-computer interaction. In *Proc. IEEE 15th Int. Conf. Pattern Recognition*, volume 2, pages 921–924, 2000.

[6] M. K. Hu. Visual pattern recognition by moment invariants. *IRE Trans. Information Theory*, 8:179–187, 1962.

[7] K. Kuno and Y. Shirai. Manipulative hand gesture recognition using task knowledge for human computer interaction. In *FG '98: Proceedings of the 3rd. International Conference on Face & Gesture Recognition*, page 468, Washington, DC, USA, 1998. IEEE Computer Society.

[8] A. McIvor. Background subtraction techniques. In *Proc. of Image and Vision Computing*, Auckland, New Zealand, 2000.

[9] D. McNeill and E. Levy. *Conceptual Representations in Language Activity and Gesture*. John Wiley and Sons Ltd, thirteenth edition, 1982.

[10] R. L. Morin and B. E. Raeside. A reappraisal of distance-weighted k-nearest neighbor classification for pattern recognition with missing data. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-11(3):241–243, 1981.

[11] Keung-Chi Ng and Bruce Abramson. Consensus in a multi-expert system. In *CSC '90: Proceedings of the 1990 ACM annual conference on Cooperation*, pages 351–357, New York, NY, USA, 1990. ACM.

[12] Nuria M. Oliver, Barbara Rosario, and Alex P. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):831–843, 2000.

[13] Isaac Weiss. Geometric invariants and object recognition. *Int. J. Comput. Vision*, 10(3):207–231, 1993.

[14] Yair Weiss and Edward H. Adelson. Slow and smooth: A bayesian theory for the combination of local motion signals in human vision. Technical Report AIM-1624, Massachusetts Institute of Technology, 1998.

[15] F. Zernike. Beugungstheorie des schneidenverfahrens und seiner verbesserten form, der phasenkontrastmethode. *Physica*, 1:689–704, 1934.

[16] H. Zhou, D. J. Lin, and T. S. Huang. Static hand gesture recognition based on local orientation histogram feature distribution model. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshop*, page 161, 2004.