A Naïve Bayes Classifier with Distance Weighting for Hand-Gesture Recognition

Pujan Ziaie, Thomas Müller, Mary Ellen Foster, and Alois Knoll

Technical University of Munich,
Dept. of Informatics VI, Robotics and Embedded Systems,
Boltzmannstr. 3, DE-85748 Garching, Germany,
{ziaie,muelleth,foster,knoll}@cs.tum.edu

Abstract. We present an effective and fast method for static hand gesture recognition. This method is based on classifying the different gestures according to geometric-based invariants which are obtained from image data after segmentation; thus, unlike many other recognition methods, this method is not dependent on skin color. Gestures are extracted from each frame of the video, with a static background. The segmentation is done by dynamic extraction of background pixels according to the histogram of each image. Gestures are classified using a weighted K-Nearest Neighbors Algorithm which is combined with a nave Bayes approach to estimate the probability of each gesture type.

Key words: Image Processing; Gesture Recognition; K-Nearest Neighbors; Nave Bayes; Classification; Human-robot interaction

1 Introduction

When humans interact with one another – and with artificial agents – they make extensive use of a range of non-verbal behavior in addition to communicating via speech. Processing and understanding the non-verbal parts of human communication are crucial to supporting smooth interaction between a human and a robot.

We concentrate on the task of *hand-gesture recognition*: recognizing and classifying the hand shapes and motions of a human user in the context of a cooperative human-robot assembly task. Hand gestures play an important role in this type of interaction, both as an accompaniment to speech and as a means of input in their own right. For example, if a user wants to tell a robot to pick up a certain object among many other objects, it can be difficult to indicate the desired object using only speech. However, if the user combines saying "Pick up that object." with a pointing gesture at the target object, this can be easier to process. Hand gestures can also themselves provide strong indications of the users intentions in the absence of speech: for example, users might move their hand near an object in preparation for picking it up, or may hold out their hand to indicate that they need the robot to hand over a particular object.

In this paper, we introduce and evaluate a method to recognize the following three types of gestures in a human-robot dialog system: pointing, grasping and holding out (Figure 1).

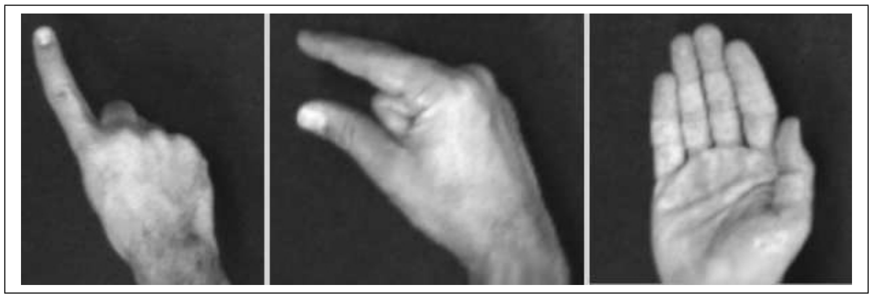

Fig. 1. Pointing, grasping, and holding-out gestures

The paper is organized as follows: we begin by discussing related work in the area of gesture recognition, particularly related approaches to the sub-tasks of image segmentation and classification. In the main part of the paper, we present a fast, robust and easy-to-implement gesture recognition algorithm which differentiates between three gesture classes mentioned above, and which can be extended to other similar applications. After the algorithm has been described in detail, we describe an experiment in which gestures from the JAST project were recognized and classified with an overall accuracy of 93%. At the end of the paper, we draw some conclusions and summarize results of this work.

2 Related Work

Many methods have been developed recently to perform successful gesture recognition. Most of these systems consist of two main steps: segmentation and extraction of invariants, and classification of gestures. In this section we discuss how similar applications perform these two steps.

2.1 Extracting Invariants

Invariants are shape descriptors extracted from an image that are independent of the viewpoint [1]. 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. 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 do the early segmentation process using skincolor histograms [2–5]. The problem with these methods is that they do not work 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. In the target JAST application, the background is static and can easily be eliminated (e.g. using methods described in [6]), so we concentrate instead on the geometric characteristics of the objects.

Zhou *et al.* [2] extracted invariants for gesture recognition using overlapping sub-windows, 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, but very time-consuming indeed.

Kuno and Shirai [7] used seven invariants to do handgesture 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 extracted inspired us for some future improvements.

2.2 Classification

Classification is a method to assign a class to a point (vector in spaces of more than two dimensions) in 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).

A naïve Bayes classifier assigns the most likely class to a given example given its feature vector, simplifying the task greatly by assuming that the features are independent given a class. Such classifiers are robust, simple to implement and computationally efficient – and, despite the often unrealistic assumption of independence, they are frequently very successful in practice. Many techniques have been developed to improve the performance of naïve Base classifiers; Zheng and Webb [8] provide an overview of efforts in this area.

K-nearest neighbors (KNN) classifiers have a good performance when the attributes of a system are linearly separable. It finds the K nearest (already classified) vectors in the space to the input. The class which has the most vectors in those K neighbors is chosen to be the class of the input vector. K-nearest neighbors with distance weighting (KNNDW) is an improvement which has been proved to outperform KNN in many cases [9]. In this method, the contribution of each neighbor to the overall classification is weighted by its distance from the point being classified.

The most relevant work to our method has been performed by Frank *et al.* [10] which introduces a locally weighted naïve Bayes (LWNB) classifier. Their evaluation on UCI dataset shows that LWNB outperforms KNN and KNNDW when K is big enough. Other refinements, like instance cloning local nave Bayes (ICLNB) [11], have also been introduced which manipulate the training data to get a better performance from the Bayes classifier.

In our implementation, we use a combination of KNNDW and LWNB to get a better performance without manipulating the training data or any complicated modification. The complete explanation can be found in Section 4.

3 Preoprocessing

The first step in the gesture-recognition process is to process the raw images from the overhead camera to extract the background and to identify *Regions of Interest* (ROI) for the gesture-recognition process. Once ROIs have been identified, the next step is to extract the geometric invariants from the binary image for use in classification. For each ROI we define the following invariants:

1. Number of points
2. Length of the outer contour
3. Change of gradient in x direction
4. Change of gradient in y direction

Since the user's hands enter the image from outside the camera view, we consider for gesture recognition only those ROIs which end at one of the four sides of the image (table).

For extracting the invariants, only a specific, predefined area of the end part of the ROI is processed. This way a completely stretched hand will be processed from the wrist to the finger tips. Next, the outer contour of the object is extracted, where the length of this contour is the second invariant.

Then we explore the contour to find the x-y gradient changes, which corresponds to the number of changes in direction. We refer to these points as *gradient points*. To avoid noise, we inspect only the changes in direction which last for a known number of steps (three steps in our application).

Supposing that we have m invariants, we have a vector with m (which is four in our application) dimensions.

$$Inv(m) = \{a_1, a_2, \ldots, a_m\} \tag{1}$$

During the training phase (Section 4.1), the resulting vector is added to the training pool; during the classification phase (Section 4.2), it is compared against the three gesture classes for identification.

4 Gesture Recognition

Before performing classification, a training pool is created based on a range of gestures produced by different users, where each training instance is labeled with its gesture class. The invariants from this pool are stored for use in the classification process. In Section 4.1, we describe the training process, while in Section 4.2 we describe how classification proceeds.

Note that we are classifying static gestures, while the user's hands could be in motion. We therefore wait for the system to reach a stable state before performing training or classification. A stable state is detected by tracking the coordinates of the ROIs and initiating gesture recognition only once the coordinates remain constant for several frames.

4.1 Training Phase

For the training phase, the user moves his or her hand in different positions and angles for each of the gestures, using both the left and right hands. As stated before, we have three classes of gestures:

$$C(m) = \{c_1, c_2, c_3\} \tag{2}$$

All the extracted invariants are saved in a simple text file. It is recommended that the training is done with a couple of users with different hand size and shape so that the classifier becomes more robust. In our application we used four users' hands. For each gesture around 150 samples is sufficient, so at the end of the training process we have a file consisting of 500 to 600 labeled gestures.

The vectors in the file have one more dimension in comparison with the invariant vector because of the class-id. If we assume that there are n vectors in the file (n samples) then each vector will be:

$$\begin{aligned} Tr_n(m) &= \{i_0, i_1, \ldots, i_m\} \\ i_0 &\subseteq C \end{aligned} \tag{3}$$

After constructing this pool of labeled invariant vectors, classification is able to proceed.

4.2 Classification Phase: Combining KNNDW and LWNB

To classify the extracted invariant, we first find the K nearest neighbors which are calculated based on the weighted distance of each training vector to the input invariant. Formally, we define the distance-weighting vector as:

$$W_{dist}(m) = \{wDist_1, wDist_2, \ldots, wDist_m\} \tag{4}$$

The distance from the extracted invariant to training vector n can then be computed in Euclidean space as follows:

$$dist_n(tr, Inv) = \sqrt{\sum_{i=1}^{m} \frac{(Tr_n(i) - Inv(i))^2}{wDist_i}} \tag{5}$$

We also normalize the distance so that all the values will be in $[0, 1]$. Next, we choose the K vectors from the training pool which have the shortest distance to the given invariant.

$$\begin{aligned} c(x) &= \{Tr_x(1), dist_x\} \\ x &= \{1, \ldots, K \end{aligned} \tag{6}$$

A normal naïve Bayes probability for the class of the given invariant will then be

$$p\left(C(j)|x\right) = \frac{\sum_{x=1}^{K} \delta\left(C(j), c(x)\right)}{K} \tag{7}$$

$$\delta(a,b) = \begin{cases} 1 \text{ if } a = b \\ 0 \text{ otherwise} \end{cases} \tag{8}$$

where j is the index of each class (type of gesture).

To improve the result, we add weights to the neighbors. This weight $wB(x) = f(d_x)$ is a function of the Euclidian distance of each vector d_x, which can be any monotonically decreasing function. In our application, we experimented with a functions like $f(d_x) = (1 - d_x)$ or $f(d_x) = (d_x)^{-p}$ for various p, but the function which produces the best classification performance was:

$$wB(x) = f(d_x) = \frac{1 - d_x}{1 + d_x} \tag{9}$$

Using these weights, we define our locally weighted naïve Bayes probability by weighting equation 4.2 as

$$p\left(C(j)|x\right) = \frac{\sum_{x=1}^{K} wB(x)\delta\left(C(j), c(x)\right)}{\sum_{y=1}^{K} wB(y)} \tag{10}$$

Then we can simply choose the class with the highest probability.

$$c(Inv) = \text{argmax}_{j=1,\ldots,3}\, p\left(C(j)|x\right) \tag{11}$$

The classification algorithm can be summarized as follows:

1. Find the k-nearest neighbors with weighted distance from the training pool.
2. Find the naïve Bayes probability of each, while weighted disproportional to their distance.
3. Choose the class of the vector with the highest probability.

5 Experimental Results

In order to test the recognition algorithm we constructed a training pool with less than 200 samples for each of the gestures, for a total of 580 samples in the training pool. These samples were made by three persons in different lighting conditions. Then we created a testing pool with about 40 samples for each gesture by a person other than those three whose gestures were represented in the training pool.

The highest overall performance without weighting the invariants shows 91.3% correct classifications with $K = 4$.

After trying many combinations of weights on the members of invariants, we found the best weighting vector $wDist$ to be $W_{dist}(m) = \{0.6, 0.6, 1.0, 1.0\}$. That is, the gradient changes are both weighted at 1.0, while the number of points and contour length are weighted at 0.6. This result is intuitively acceptable: the range of objectpoints and length of contour is much wider than the number of changes in gradients.

Testing the recognition system with distance weighting, we achieved the results shown in Figure 2. It is quite obvious that after applying distance weighting the performance increases and the fluctuation decreases. We observed that when $K \geq 7$ the performance starts to sink. The best performance seemed to be for $4 \leq K \leq 7$.

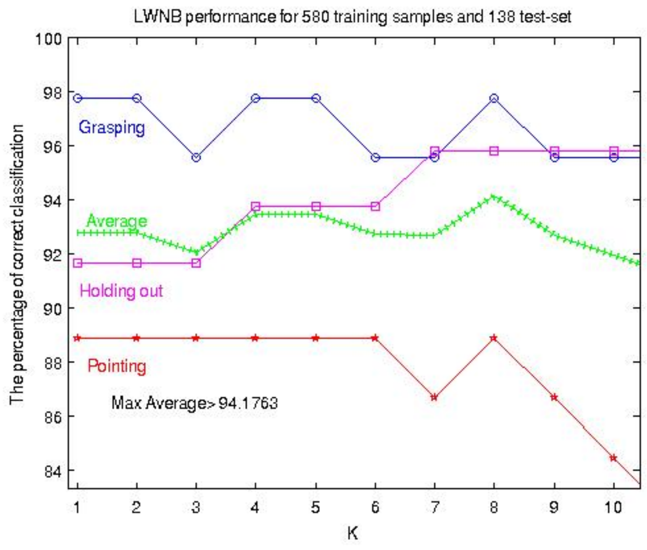

Fig. 2. Classification result with distance weighting

Running the full gesture-recognition process on a frame takes less than 50 msec on average. Of this time, segmentation takes 20–30 msec, while the recognition process takes 10–20 msec.

6 Conclusion

We have described a static gesture recognition method to distinguish between three gestures types: pointing, grasping and holding out. The process is based on classifying invariants of image blocks using a locally weighted naïve Bayes and K-nearest neighbors classifier.

The preprocessing is done by an adaptive method first extracting the background, which is considered to be unicolored (surface of the table), and segmenting the remaining pixels into regions of interest afterwards.

Four invariants of each ROI are then extracted. These invariants are: Number of pixels, length of the outercontour and changes in x and y gradients. The extracted invariants are then compared against the invariants in a pool of labeled examples created during a training phase for each type of the gestures. The best suitable type of gesture is then given by using a locally weighted nave Bayes classifier which is fed by the K-nearest neighbors of each invariant in the invariants pool.

After classification, an appropriate algorithm is applied in order to obtain symbolic information according to the type of gesture. This information is the finger-tip and its angle for pointing gestures, area of grasping for grasping gestures and the center of hand-pit for holding out type.

In an experiment, the whole process takes less than 50 msec in total and has an overall performance of about 93% at identifying the correct gesture type.

7 Acknowledgements

We thank the Cognitive Robotics Group at TUM for useful discussions and suggestions, particularly Thomas Rückstieß and Christian Osendorfer.

This research was supported by the EU project JAST (FP6-003747-IP).

References

1. Weiss, I.: Geometric invariants and object recognition. Int. J. Comput. Vision **10**(3) (1993) 207–231
2. Zhou, H., Lin, D.J., Huang, T.S.: Static hand gesture recognition based on local orientation histogram feature distribution model. In: Proc. IEEE CVPR Workshop. (2004) 161
3. Hongo, H., Ohya, M., Yasumoto, M., Yamamoto, K.: Face and hand gesture recognition for human-computer interaction. In: Proc. IEEE 15th Int. Conf. Pattern Recognition. Volume 2. (2000) 921–924
4. Wu, H., Shioyama, T., Kobayashi, H.: Spotting recognition of head gestures from color image series. In: Proc. ICPR. Volume 1., Washington, DC, USA, IEEE Computer Society (1998) 83
5. H. Boehme, e.a.: User localization for visually-based human-machine-interaction. In: Proceedings International Conference on Automatic Face- and Gesture Recognition, Washington, DC, USA, IEEE Computer Society (1998) 486–491
6. McIvor, A.: Background subtraction techniques. In: Proc. of Image and Vision Computing, Auckland, New Zealand (2000)
7. Kuno, K., Shirai, Y.: Manipulative hand gesture recognition using task knowledge for human computer interaction. In: Proc. of International Conference on Face & Gesture Recognition, Washington, DC, USA, IEEE Computer Society (1998) 468
8. Zheng, Z., Webb, G.I.: Lazy learning of bayesian rules. Mach. Learn. **41**(1) (2000) 53–84
9. Morin, R.L., Raeside, B.E.: 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) (1981) 241–243
10. Frank, E., Hall, M., Pfahringer, B.: Locally weighted naive bayes. In: Proc. of UAI-03, San Francisco, CA, Morgan Kaufmann (2003) 249–25
11. Jiang, L., Zhang, H., Su, J.: Learning k-nearest neighbor naive bayes for ranking. LNCS **3584** (2005) 175–185