VIEW INVARIANT GESTURE RECOGNITION USING 3D MOTION PRIMITIVES

M.B. Holte and T.B. Moeslund

Computer Vision and Media Technology Lab, Aalborg University, Denmark. E-mail: tbm@cvmt.dk

ABSTRACT

This paper presents a method for automatic recognition of human gestures. The method works with 3D image data from a range camera to achieve invariance to viewpoint. The recognition is based solely on motion from characteristic instances of the gestures. These instances are denoted 3D motion primitives. The method extracts 3D motion from range images and represent the motion from each input frame in a view invariant manner using harmonic shape context. The harmonic shape context is classified as a 3D motion primitive. A sequence of input frames results in a set of primitives that are classified as a gesture using a probabilistic edit distance method. The system has been trained on frontal images (0° camera rotation) and tested on 240 video sequences from 0° and 45°. An overall recognition rate of 82.9% is achieved. The recognition rate is independent of the viewpoint which shows that the method is indeed view invariant.

Index Terms— Machine vision, stereo vision, gesture recognition, view invariant, 3D motion primitives

1. INTRODUCTION

In human communication gestures are widely used to convey or emphasise information and the automatic recognition of gestures has therefore rechieved much attention in many areas of computer vision research. Reconstruction of humans and their exact pose has been a widely used approach but a current trend is to do recognition directly on image data, e.g. silhouette data [1, 2] or spatio-temporal features [3, 4].

All systems relying on information extracted from 2D images are faced with the common problem that the 2D image is a projection of the 3D gestures. Some overcome this by merely addressing gestures carried out in a plane parallel to the camera-plane. To handle different view-points a new class can be learned for each gesture for each view-point (given some resolution). This leads to even more tedious training *and* the risk that too many classes might lead to overlap in the feature-space, which again results in reduced recognition rates. Furthermore, some gestures might be ambiguous in such a systems. E.g., a "point right" gesture seen from the front is similar to a "point straight ahead" gesture seen from a side view. To overcome such problems recent methods have investigated the use of 3D data as opposed to 2D image data [1, 5].

We are interested in the general case of recognizing 3D gestures from different viewpoints and therefore apply 3D data. We want to avoid the possible problems inherent to classical stereo approaches (the correspondence problem, careful camera placement and calibration) as used in [1, 5] and instead apply a 3D range camera.

In this paper we aim at view invariant gesture recognition directly on range data. Recognizing gestures directly on range data can be based on data from all frames constituting that gesture, e.g. a trajectory through some state-space. A more flexible approach is to define and recognize gestures by a set of primitives [2]. We represent gestures as an ordered sequence of 3D motion primitives (temporal instances). We focus on arm gestures and therefore only segment the arms (when they move) and hereby suppress the rest of the (irrelevant) body information. Concretely we use 3D double difference images to extract the moving arms and represent this data by their Shape Context. We make the primitives invariant to rotation around the vertical axis by re-representing the Shape Context using Spherical Harmonic basis functions, yielding a Harmonic Shape Context representation.

In each frame the primitive, if any, that best explains the observed data is identified leading to a discrete recognition problem since a video sequence of range data will be converted into a string containing a sequence of symbols, each representing a primitive. After pruning the string a probabilistic Edit Distance classifier is applied to identify which gesture best describes the pruned string. Our approach is illustrated in figure 1.

2. SEGMENTATION

2.1. Data Acquisition

We capture 3D data using a CSEM SwissRanger SR-2 range camera (see figure 1) [6]. The camera is based on the Time-Of-Flight principle and emits radio-frequency modulated light in the near-infrared spectrum, which is backscattered by the scene and detected by a CMOS CCD. The resolution is 160×124 pixels with an active range of 7.5 m. The depth accuracy is typically in the order of a few centimeters, depending of the distance range and illumination. Figure 2(left) shows a range image of one time instant of a point right gesture.

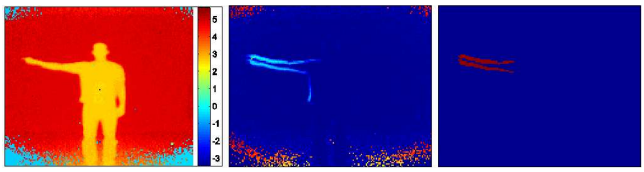

Fig. 2. *Left: A range image, where the pixel values correspond to a distance. Middle: The difference range image used for motion detection. Right: The resulting motion detected in 2D after hysteresis bandpass filtering and creation of a double difference image.*

2.2. 3D Motion Detection

We detect movements (of the arms) using a 3D version of 2D double differencing [7]. This is done by subtracting the depth values pixelwise in two pairs of depth images (see figure 2(middle)), thresholding and finally ANDing the two binary images. The moving arm (and its shadow) is visible in the binary image, but so is a large

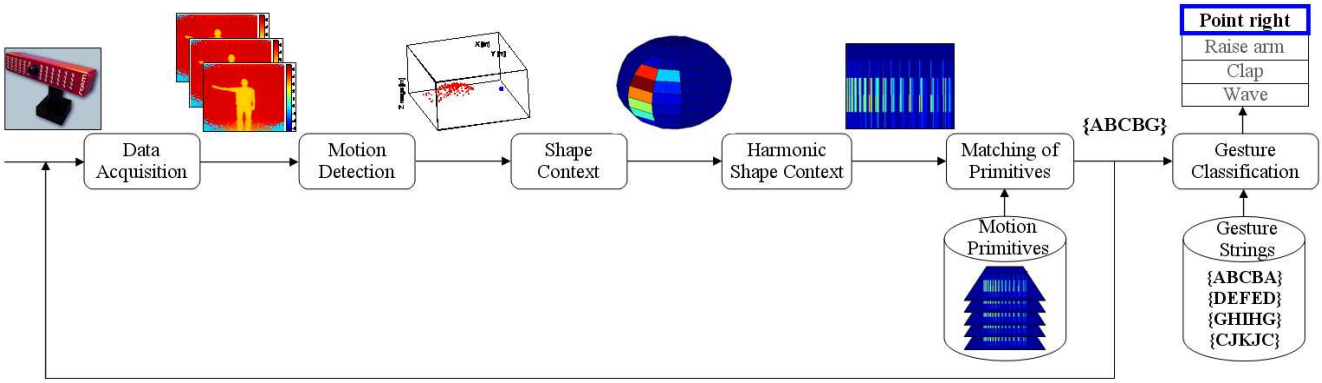

Fig. 1. *An overview of the range based gesture recognition system. Note that the feedback loop illustrates that a number of frames are processed before recognition of gestures commences.*

amount of noise due to erroneous depth values often produced by the SwissRanger camera. To handle these noise effects, each of the two 3D difference images is filtered with a hysteresis bandpass filter before they are ANDed together (see figure 2(right)). This filter operates in 2D and uses four threshold values T_1, T_2, T_3 and T_4. The 3D difference values that fall within the motion range $[T2, T3]$ are most likely to originate from arm movements. Pixels in the range $[T1, T2] \bigcup [T3, T4]$ are also classified as belonging to the arm if and only if they are connected with pixels from $[T2, T3]$. This hysteresis principle yields less fragmented motion regions while excluding noisy image regions. Too small motion regions caused by noise or unwanted motion along the body are filtered by a size criterion.

3. MOTION PRIMITIVES

3.1. Shape Context

After detecting the motion we are left with a point cloud in 3D (see figure 1). We represents this data efficiently using shape context [8]. A shape context is based on a spherical histogram. This histogram is centered in a reference point (center of gravity of the human body) and divided linearly into S = 12 azimuthal (east-west) bins and T = 12 colatitudinal (north-south) bins, while the radial direction is divided into U = 5 bins. The radial division is made in steps of 30 cm. The value of a bin is given by the number of 3D points falling within that particular bin. This results in an n ($S \times T \times U = 12 \times 12 \times 5 = 720$) dimensional feature vector for each frame. Figure 3 gives an example of the shape context descriptor.

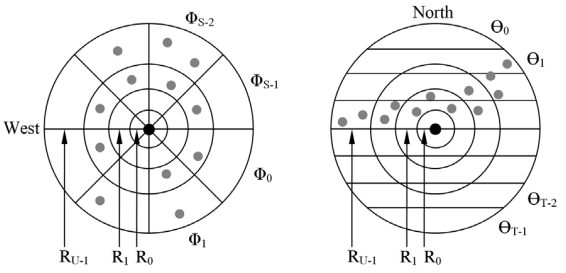

Fig. 3. *A horizontal and a vertical cross-section of a Shape context descriptor.*

3.2. View Invariant Representation: Harmonic Shape Context

By introducing spherical harmonics we can eliminate one of the two rotational parameters in a shape context descriptor. We eliminate the rotation around the vertical axis, see figure 3, and hereby make our representation invariant to variations in this parameter.

Any given spherical function, i.e. a function $f(\theta, \phi)$ defined on the surface of a sphere parameterized by the colatitudinal and azimuthal variables θ and ϕ, can be decomposed into a weighted sum of spherical harmonics as given by equation 1.

$$f(\theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} A_l^m \, Y_l^m(\theta, \phi) \tag{1}$$

The term A_l^m are the weighing coefficient of *degree* m and *order* l, while the complex functions $Y_l^m(\cdot)$ are the actual spherical harmonic functions of *degree* m and *order* l. The following states the key advantages of the mathematical transform based on the family of orthogonal basis functions in the form of spherical harmonics. The complex function $Y_l^m(\cdot)$ is given by equation 2.

$$Y_l^m(\theta, \phi) = K_l^m \, P_l^{|m|}(\cos\theta) \, e^{jm\phi} \tag{2}$$

The term K_l^m is a normalization constant, while the function $P_l^{|m|}(\cdot)$ is the *associated Legendre Polynomial*. The key feature to note from equation 2 is the encoding of the azimuthal variable ϕ. The azimuthal variable solely inflects the *phase* of the spherical harmonic function and has no effect on the *magnitude*. This effectively means that $||A_l^m||$, i.e. the norm of the decomposition coefficients of equation 1 is invariant to parametrization in the variable ϕ.

The actual determination of the spherical harmonic coefficients is based on an inverse summation as given by equation 3, where N is the number of samples ($S \times T$). The normalization constant $4\pi/N$ originates from the fact, that equation 3 is a discretization of a continuous double integral in spherical coordinates, i.e. $4\pi/N$ is the surface area of each sample on the unit sphere.

$$(A_l^m)_{f_u} = \frac{4\pi}{N} \sum_{\phi=0}^{2\pi} \sum_{\theta=0}^{\pi} f_u(\theta, \phi) \; Y_l^m(\theta, \phi) \tag{3}$$

In a practical application it is not necessary (or possible, as there are infinitely many) to keep all coefficient A_l^m. Contrary, it is assumed the functions f_u (f_u are the spherical functions for $u \in [0; U-1]$) are band-limited why it is only necessary to keep coefficient up to some bandwidth. Concretely we use 136 coefficients (see figure 1).

4. CLASSIFICATION

The classification is devided into two main tasks: recognition of motion primitives by use of the harmonic shape context descriptors, and recognition of the actual gestures using an ordered sequence of primitives (see figure 1).

4.1. Recognition of Primitives: Correlation

A motion primitive is recognized by matching the current harmonic shape context with a known set, one for each possible primitive. The actual comparison of two harmonic shape contexts is done by the normalized correlation coefficient. To this end each harmonic shape context is represented as a vector of length n containing the (stacked) spherical harmonic coefficients for the specific surface region.

The system is trained by generating a representative set of descriptors for each primitive. A reference descriptor is then estimated as the average of all these descriptors for each class (primitive).

After processing a sequence the output will be a string with the same length as the sequence. An example is illustrated in equation 4. Each letter corresponds to a recognized primitive and $\emptyset$ corresponds to time instances where no primitives are detected. The string is pruned by first removing '$\emptyset$'s, isolated instances, and then all repeated letters, see equation 5. A weight is generated to reflect the number of repeated letters (this is used below).

$$\begin{aligned} \text{String} &= \{\emptyset, \emptyset, B, B, B, B, B, E, A, A, F, F, F, F, \emptyset, D, \\ &\quad D, G, G, G, G, \emptyset\} \quad (4) \\ \text{String} &= \{B, A, F, D, G\} \quad (5) \\ \text{Weights} &= \{5, 2, 4, 2, 4\} \quad (6) \end{aligned}$$

4.2. Recognition of Gestures: Probabilistic Edit Distance

The result of recognizing the primitives is a string of letters referring to the known primitives. During a training phase a string representation of each gesture to be recognized is learned. The task is now to compare each of the learned gestures (strings) with the detected string. Since the learned strings and the detected strings (possibly including errors!) will in general not have the same length, the standard pattern recognition methods will not suffice. We therefore apply the Edit Distance method [9], which can handle matching of strings of different lengths.

The edit distance is a well known method for comparing words or text strings, e.g., for spell-checking and plagiarism detection. It operates by measuring the distance between two strings in terms of the number of operations needed in order to transform one to the other. There are three possible operations: *insert* a letter from the other string, *delete* a letter, and *exchange* a letter by one from the other string. Whenever one of these operations is required in order to make the strings more similar, the score or distance is increased by one. The algorithm is illustrated in figure 4 where the strings *motions* and *octane* are compared.

The first step is initialization. The two strings are placed along the sides of the matrix, and increasing numbers are place along the borders beside the strings. Hereafter the matrix is filled cell by cell by traversing one column at a time. Each cell is given the smallest value of the following four operations:

Insert: The value of the cell above + 1

Delete: The value of the cell to the left + 1

Exchange: The value of the cell up-left + 1

No change: The value of the cell up-left + 0. This is the case when the letters in question in the two stings are the same.

Using these rules the matrix is filled and the value found at the bottom right corner is the edit distance required in order to map one string into the other, i.e., the distance between the two strings. The actual sequence of operations can be found by back-tracing the matrix. Note that often more paths are possible.

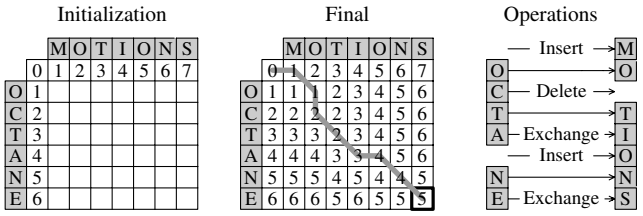

Fig. 4. *Measuring the distance between two strings using edit distance.*

When the strings representing the gestures are of different lengths, the method tends to favor the shorter strings. Say we have detected the string $\{B, C, D\}$ and want to classify it as being one of the two gestures: $\#1 = \{J, C, G\}$ and $\#2 = \{A, B, C, D, H\}$. The edit distance from the detected string to the gesture-strings will be two in both cases. However, it seems more likely that the correct interpretation is that the detected string comes from gesture #2 in a situation where the start and end has been corrupted by noise. In fact, 2 out of 3 of the primitives have to be changed for gesture #1 whereas only 2 out of 5 have to be changed for gesture #2. We therefore normalize the edit distance by dividing the output by the length of the gesture-string, yielding 0.67 for gesture #1 and 0.2 for gesture #2, i.e., gesture #2 is recognized.

The edit distance is a deterministic method but by changing the cost of each of the three operations with respect to likelihoods it becomes a probabilistic method[1]. Concretely we apply the weights described above, see equation 6. These to some extent represent the likelihood of a certain primitive being correct. The higher the weight the more likely a primitive will be. We incorporate the weights into the edit distance method by increasing the score by the weight multiplied by β (a scaling factor) whenever a primitive is *deleted* or *exchanged*. The cost of *inserting* remains 1.

The above principle works for situations where the input sequence only contains one gesture (possibly corrupted by noise). In a real scenario, however, we will have sequences which are potentially much longer than an gesture and which might include more gestures after each other. The gesture recognition problem is therefore formulated as for each gesture to find the substring in the detected string, which has the minimum edit distance. The recognized gesture will then be the one of the substrings with the minimum distance. Denoting the start point and length of the substring, s and l, respectively, we recognize the gesture present in the detected string as:

$$\text{Gesture} = \arg\min_{k,s,l} PED(\Lambda, k, s, l) \quad (7)$$

where k index the different gestures, Λ is the detected string, and $PED(\cdot)$ is the probabilistic edit distance.

[1] This is related to the Weighted Edit Distance method, which however has fixed weights.

5. TEST AND RESULTS

For testing purpose we use a vocabulary consisting of 11 primitives. This is illustrated in figure 5. The criteria for finding the primitives are 1) that they represent characteristic and representative 3D configurations, 2) that their configurations contain a certain amount of motion, and 3) that the primitives are used in the description of as many gestures as possible, i.e., fewer primitives are required. By use of this vocabulary of primitives we describe 4 one- and two-arms gestures: "Point right", "raise arm", "clap" and "wave".

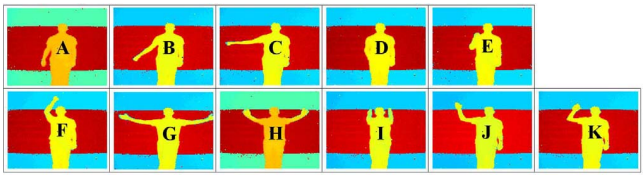

Fig. 5. *The vocabulary consisting of 11 primitives. The primitves are illustrated by range images of the arm configurations, which are color coded. The color can vary slightly due to error pixels and normalization.*

We test the system on range data recorded of 10 test subjects, each performing the four gestures 3 times from a $0°$ and $45°$ viewpoint with respect to the camera. A total of 240 video sequences have been recorded. Figure 6 shows an example of the visual differences that occur when a gesture is performed from these two viewpoints.

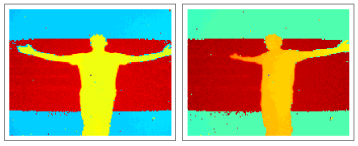

Fig. 6. *Range data examples of a time instance from a video sequence including a person carrying out a "clap" gesture shown from a $0°$ and $45°$ camera viewpoint.*

To evaluate the view invariance of the system, the data which is used to train the motion primitives is only from the $0°$ viewpoint. The overall matching rate is 82.9% and the error distribution can be seen in the confusion matrix in figure 7. In comparison, when only testing on sequences from $0°$ we obtain a recognition rate of 84.2%.

	1	2	3	4
1. Point right	81.7	0.0	8.3	10.0
2. Raise arm	8.3	88.3	0.0	3.4
3. Clap	11.7	8.3	78,3	1.7
4. Wave	6.7	1.7	8.3	83.3

Fig. 7. *Test results (given in percentages) for the 4 gestures recorded from a $0°$ and $45°$ viewpoint with respect to the camera.*

No significant increase in error can be observed when training and testing on sequences from different viewpoints, i.e., the approach is indeed view invariant. The errors observed in both tests are mainly due to personal variations when performing gestures like "point right" and "raise arm". I.e., some tend to raise their arm above the shoulder while pointing while some do not stretch their arm fully when raising their arm. Another example is in the case of a "clap" gesture, where one of the arms might not be visible or segmented properly due to a too extreme viewpoint when the individual performs this gesture. Hence a "clap" gesture might be classified to be more likely a "point right" gesture. Furthermore, the points included in the motion cloud are not perfectly located at the arms but stretches backwards. These miss-located points can cause impact on the primitive descriptors, and hereby lead to errors.

6. CONCLUSION

In this paper we have presented a method for view invariant gesture recognition. Our approach does not rely on information from an entire video sequence but perform gesture recognition using temporal instances that only represent a subset of the original sequence. We have applied a range camera and segment 3D primitives based on motion. These are represented compactly and view invariant using harmonic shape context. A probabilistic Edit Distance classifier is applied to identify which gesture best describes a string of primitives. We have evaluated the method on 240 video sequences recorded from a $0°$ and $45°$ viewpoint with respect to the camera. The results indicate a valid approach of recognizing 82.9% of the gestures correctly. It should be noted that the classifier is trained on gestures from only one view ($0°$) and tested on gestures from a very different view ($45°$). With this test scenario we have shown that the method is indeed view invariant.

Acknowledgements: This work is funded by the MoPrim project (Danish National Research Councils - FTP) and the HERMES project (FP6 IST-027110). The authors would like to thank Professor Thomas Bak, AAU, for providing access to the CSEM SwissRanger Camera.

7. REFERENCES

[1] D. Weinland, R. Ronfard, and E. Boyer, "Free Viewpoint Action Recognition using Motion History Volumes," *Computer Vision and Image Understanding*, vol. 104, no. 2, 2006.

[2] A. Yilmaz and M. Shah, "Actions Sketch: A Novel Action Representation," in *Computer Vision and Pattern Recognition, vol. 1*, 2005.

[3] A. Bobick and J. Davis, "The Recognition of Human Movement Using Temporal Templates," *Pattern Analysis and Machine Intelligence*, vol. 23, no. 3, 2001.

[4] Alonso Patron-Perez and I. Reid, "A Probabilistic Framework for Recognizing Similar Actions using Spatio-Temporal Features," in *British Machine Vision Conference*, UK, Sep. 2007.

[5] I. Cohen and H. Li, "Inference of Human Postures by Classification of 3D Human Body Shape," in *Workshop on Analysis and Modeling of Faces and Gestures*, Nice, France, Oct. 2003.

[6] T. Oggier, M. Stamm, M. Schweizer, and J. Pedersen, "User manual swissranger 2 rev. b," vol. Version 1.02, March 2005.

[7] Y. Kameda, M. Minoh, and K. Ikeda, "Three Dimensional Motion Estimation of a Human Body Using a Difference Image Sequence," in *Asian Conference on Computer Vision*, Singapore, Dec. 1995.

[8] S. Belongie, J. Malik, and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts," *Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, 2002.

[9] V.I. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Doklady Akademii Nauk SSSR*, vol. 163, no. 4, 1965.