6D MOTION GESTURE RECOGNITION USING SPATIO-TEMPORAL FEATURES

Mingyu Chen, Ghassan AlRegib, and Biing-Hwang Juang

School of Electrical and Computer Engineering, Georgia Institute of Technology
Atlanta, Georgia 30332, U.S.A.
{mingyu, alregib, juang}@gatech.edu

ABSTRACT

Depending on the tracking technology in use, a 6D motion gesture can be tracked and represented explicitly by the position and orientation or implicitly by the acceleration and angular speed. In this work, we first present the reasoning for the definition and recognition of motion gestures. Five basic feature vectors are then derived from the 6D motion data. Our main contribution is to investigate the relative effectiveness of various feature dimensions for motion gesture recognition in both user dependent and user independent cases. We also propose a feature normalization procedure and prove its effectiveness in achieving "scale" invariance especially in the user independent case. Our study gives an insight into the attainable recognition rate with different tracking devices.

Index Terms— Motion Gesture, Gesture Recognition, Spatio-Temporal Features

1. INTRODUCTION

The control motion of conventional pointing devices, such as a mouse, trackpad, and touchscreen, is limited to trajectories on a plane. The 2D motions also form the basis of gestures on the traditional devices. With the development of tracking technologies, it is possible to track the body motion in 3D space at an affordable price and with good accuracy. Motion-based control is gaining popularity, and motion gestures provide a natural and complementary modality in human-computer interactions. Motion information beyond a 2D trajectory, such as depth and orientation, expands the definition of motion gestures and may improve the accuracy and robustness of gesture recognition. In this work, we focus on motion gestures performed by a hand or a handheld device. Tracking an object in space actually requires six dimensions: three for translation and three for rotation. Therefore, a 6D motion gesture is represented by a 3D spatial trajectory and augmented by another three dimensions of orientation.

The tracking technology in use affects the sampling rate, latency, resolution, and accuracy of the spatio-temporal data captured from a motion gesture. Nowadays, the most popular tracking technologies are optical sensing and inertial sensing. The former usually tracks the explicit 6D motion, i.e., the position and orientation in a global reference frame. The inertial sensing actually measures the accelerations and angular speeds in the device-wise coordinates, which depict the implicit 6D motion. Theoretically, it is possible to reconstruct the spatial trajectory from the implicit 6D data with the inertial navigation algorithm. The drifting issue and error propagation deteriorate the reconstruction over time and make it less accurate than the explicit 6D motion from optical tracking. However, the implicit motion signals still contain enough information to define and recognize the motion gestures.

A motion gesture can be viewed as a spatio-temporal pattern of different dimensions, which depend on the tracking technology in use. The spatial trajectory can be either 2D or 3D. The tracking results can be either implicit or explicit, with or without the orientation information. Gesture recognition is commonly done with hidden Markov models [1, 2, 3]. Other approaches for gesture recognition include dynamic time warping [4], data-driven template matching [5, 6], and feature-based statistical classifiers [7, 8, 9]. In general, the reported recognition rates are above 90%. Since these results are obtained with different datasets and various experimental settings, a direct comparison of performance is not meaningful.

In our previous work [9], we presented a 6D motion gesture database (6DMG) which contains both explicit and implicit 6D motion data in a "vocabulary" of 20 gestures. Given a motion gesture of various tracking signals, we extracted the corresponding fixed-length feature set for classification. The statistical features in use are either geometric or algebraic, and they barely contain any temporal information. We proposed a temporal extension to the feature set to describe the motion gesture at a very coarse scale in time, e.g., the mean values of the first half gesture. However, the overall feature extraction process still treats the gesture more like a static pattern. Benchmark recognition results were obtained by using a simple linear classifier on the extracted features.

In this work, we attempt to recognize 6DMG with a totally different approach by looking at the time series nature of the motion gesture. Derived from different tracking signals, we represent the motion gesture as a sequence of feature vectors (observations) and use HMMs to model it. It is essential to understand what features help to describe the motion gesture. We can then decide the proper HMM structure with reasonable physical meaning. Our main contribution is to investigate the relative effectiveness of various feature dimensions in motion gesture recognition. Both user dependent and user independent cases are addressed (in Section 4). We also show that the normalization process is essential for robust user independent recognition. The experimental settings are identical to our previous work [9] so that the comparison of the recognition accuracy is fair.

We introduce the tracking devices and the motion gesture set in the following section. In Section 3, we discuss the processes of feature extraction and normalization. The experiments and results are given in Section 4, and Section 5 concludes this paper.

2. 6DMG: 6D MOTION GESTURE DATABASE

We use a hybrid framework for 6D motion tracking: WorldViz PPT-X4 for optical tracking of the position and Wii Remote Plus (Wiimote) for the inertial measurement of the acceleration and angular speed. The orientation is derived from the fusion of the acceleration

Table 1. The gesture list of 6DMG

Gesture Name	Sample # avg. (std.)	max/min norm. ratio P	O	V	W	A
SwipeRight	51.9 (20.7)	8.7	5.9	14.5	12.4	24.1
SwipeLeft	51.6 (20.4)	6.4	5.0	30.3	10.8	27.4
SwipeUp	44.6 (15.5)	5.0	5.2	27.0	11.6	19.2
SwipeDown	47.2 (16.7)	4.2	7.2	46.1	21.1	29.9
SwipeUpright	45.2 (16.9)	5.4	5.6	17.7	16.6	31.7
SwipeUpleft	44.9 (17.5)	5.5	3.9	14.6	17.0	36.2
SwipeDnright	46.5 (18.8)	6.1	8.1	18.8	15.2	20.9
SwipeDnleft	47.5 (19.0)	7.2	5.2	26.2	37.8	40.3
PokeRight	70.9 (23.0)	4.2	4.5	15.4	11.2	16.5
PokeLeft	74.5 (25.1)	4.4	4.7	16.0	18.4	19.3
PokeUp	72.3 (23.4)	4.9	4.4	23.5	13.0	11.1
PokeDown	71.0 (24.9)	5.0	5.5	18.1	17.4	20.4
Vshape	71.6 (23.7)	4.4	4.9	9.7	16.0	11.4
Xshape	99.3 (28.0)	4.0	4.2	9.1	11.7	13.8
CirHorClk	104.3 (27.0)	3.5	4.2	9.0	7.3	9.1
CirHorCclk	103.1 (30.0)	3.6	4.0	9.0	10.6	7.2
CirVerClk	108.3 (33.0)	3.8	5.8	13.4	8.9	19.2
CirVerCclk	102.4 (32.0)	3.8	6.5	8.4	8.6	13.6
TwistClk	63.2 (18.9)	8.8	3.3	8.6	4.2	6.5
TwistCclk	64.5 (18.9)	11.4	2.7	6.7	4.0	10.0

and angular speed. The recorded motion data contain both explicit and implicit spatio-temporal information sampled at 60 Hz. For gesture recording, we use a push-to-gesture scheme, which allows the user to demarcate the uni-stroke motion gesture. We consider the imprecise segmentation as part of the variation of the gesture data.

We define a total of 20 gestures, including swiping motions in eight directions, poke gestures that swipe rapidly forth and back in four directions, v-shape, x-shape, clockwise or counter clockwise circles in the vertical or horizontal plane, and wrist twisting (roll). The names and duration statistics of the 20 gestures are listed in Table 1. There are no "mirror" gestures, which means the direction and rotation are the same for both right- and left-handed users.

We recruited 28 participants (21 right-handed and 7 left-handed, 22 male and 6 female) for recording. Every subject recorded each distinct gesture 10 times, and the 6DMG database has 5600 gesture samples in total. When recording, we advised the subject to perform the gesture in a consistent way, but we did not constrain his or her gripping posture, the gesture articulation style and speed. Variations of the same gesture between individuals are expected, which make the user independent recognition challenging. Space limitations preclude the implementation and recording details of 6DMG. The interested reader is referred to our technical report[1].

3. FEATURE EXTRACTION

3.1. What Defines a Motion Gesture?

Before we do the recognition, it is important to understand what defines a motion gesture. In most cases, it is the spatial trajectory that defines the motion gesture. This basically holds true not only for our gesture set but also for other existing gestures with 3D spatial or gaming interactions. Exception exists when the spatial trajectory carries little or no deterministic information. For example, the wrist twisting gesture is better defined by the change in orientation than the spatial trajectory.

The spatial trajectory can be obtained directly from the position or inferred from the fusion of acceleration and angular speed. The velocity of the trajectory can also be meaningful. The orientation or angular speed may contain side information to better describe the motion gesture. From the position data, we can derive feature vectors P_o and V_o, where $P_o = [p_x, p_y, p_z]$ denotes the positions offset by the starting position, and $V_o = [\Delta p_x, \Delta p_y, \Delta p_z]$ denotes the rate of change in the position. In 6DMG, we represent the orientation with quaternion. Although it is easier to interpret and visualize Euler angles, an Euler representation suffers from discontinuity when the angle wraps around and may cause problems when modeling. We then define another feature vector $O_o = [q_w, q_x, q_y, q_z]$, which represents the absolute orientation. The implicit 6D motion data provide the device-wise acceleration and angular speed, and they form the feature vectors $A_o = [a_x, a_y, a_z]$ and $W_o = [w_{yaw}, w_{pitch}, w_{roll}]$ respectively. Depending on the tracking signals in use, we can derive a corresponding set of features (observations) with kinematic meanings for the HMMs.

With these features, we transform the motion gesture into a spatio-temporal pattern. Each HMM state has the ability to capture a subset of that pattern, i.e., a segment of the motion. Because the motion gesture is an order-constrained time-evolving signal, the left-right HMM topology is more suitable. Each segment of the motion may have different time span, but it is unlikely to skip a segment of continuous motion when rendering a gesture. Thus, we do not consider skip transitions in the HMM topology.

3.2. Normalization

In general, people recognize a motion gesture by the path spanned by the motion regardless of its scale or speed. Therefore, the recognizer should not be affected by the scale and speed unless fast/slow or big/small motions have different meanings in the gesture set. This is very unlikely to happen especially in user independent systems because the definition of fast/slow or large/small motions can be vague and different among users. To make the recognizer scale and speed invariant, proper normalization of features is very important.

Let the upper case letters without subscripts denote the normalized features. Normalization of P_o, V_o, and W_o is straight forward with linear scaling, e.g., $P = s_{pos} P_o$. However, the scaling factor is determined differently according to the physical meaning of the normalization target. After normalization, the largest magnitude among $[p_x, p_y, p_z]$, the largest norm of $[\Delta p_x, \Delta p_y, \Delta p_z]$, and the largest magnitude among $[w_{yaw}, w_{pitch}, w_{roll}]$ are scaled to unit length.

We cannot rescale A_o directly because the gravity mixes up with the motion acceleration. In [3], gravitational acceleration is compensated by subtracting the mean of A_o based on the assumption that the sensor heading is constant over the time of one recording. Apparently this does not work in our case, and we have to use extra information, i.e., the orientation, to remove the gravitational acceleration. Given O_o, we first convert the device-wise acceleration to the global coordinates and subtract the constant gravity. A is then obtained by scaling the largest norm of the extracted motion acceleration to unit length.

The normalization of O_o may be the most tricky one because the quaternion cannot be "scaled" directly. First, we need to offset (rotate) O_o by the starting orientation so that the first orientation becomes unit quaternion. We then convert the quaternion into the

[1]The 6DMG database, technical paper, gesture viewer, loader, and exporter are available at `http://www.ece.gatech.edu/6DMG`

axis-angle representation as

$$[q_w, q_x, q_y, q_z] = \left[\cos\frac{\alpha}{2}, r_x \sin\frac{\alpha}{2}, r_y \sin\frac{\alpha}{2}, r_z \sin\frac{\alpha}{2}\right], \quad (1)$$

where α is the angle rotated about the axis $[r_x, r_y, r_z]$ by the right-hand rule. The concept here is to normalize α and keep the axis untouched, i.e., scale the rotation amount without changing the rotation direction. Note that there is an important limitation of the orientation representation: the rotation direction (or angle) is not unique. For example, rotating 1.1π and -0.9π around the same axis results in identical orientation, so we don't know the true rotation amount to normalize. The ambiguity cannot be resolved unless we keep track of the evolving orientation. Fortunately, the rotation angle in 6DMG rarely exceeds π, so we choose to wrap α to $(-\pi, \pi]$. The rotation angle is then normalized by scaling the maximum of $|\alpha|$ to π. When the true rotation angle exceeds π, there must be an intermediate orientation whose rotation angle crosses π. In such a case, the resulting scaling factor of α is very close to 1, which means no rescaling. Finally, we can convert the normalized rotation angle and the original rotation axis back to the quaternion representation of O.

The ratio of the maximum over minimum scaling factor of the normalized features can be a good indicator for "scale" variations. Table 1 lists the maximum to minimum ratio of each normalized features set for every gesture in the user independent case. In general, the ratios of P and O are much smaller than those of the time-derivative features A, V, and W. The only exception is the ratio of P for the twisting gestures, in which we already know the spatial trajectory is not deterministic. In the user dependent case, the ratios for all features basically fall under 3. Therefore, the normalization process should be more helpful for the user independent case due to its huge in-class variations as shown in Table 1. The reader has to bear in mind that normalization is no elixir. The concept of scale invariance reduces the in-class variations, but it may backfire if the variations between classes are reduced too much at the same time.

4. EXPERIMENT SETUP AND RESULTS

In this section, we would like to verify the normalization process and evaluate the recognition performance with different combinations of feature sets in both user dependent and user independent cases. Although different topologies can be specified per gesture according to its complexity, we use the same topology for all gestures for generality. The experiments are done with HMMs of 4, 6, and 8 states with single Gaussian mixture per state. We use the Hidden Markov Model Toolkit (HTK) for HMM modeling, training, and testing.

For the user dependent recognition experiment, we train the HMMs with 5 tokens randomly drawn from each gesture of a single user, and use the remaining 5 tokens for testing. For cross validation, we repeat the experiment 50 times for each of the 21 right-handed users. For the user independent case, the training set is formed from the gesture data of randomly selected 5 right-handed users. We then test with the gestures of the remaining 16 right-handed users and 7 left-handed users respectively. This is equivalent to training the recognizer in advance with only right-handers and having new right- or left-handed users simply come in and use the system. The experiment is repeated 200 times to calculate the average recognition rate. We use the same initial seed to randomize the combination of selected training samples so that the results are reproducible and comparable across different settings.

4.1. Evaluation of Normalization

First, we investigate how much the normalization of these basic features can help in both user dependent and user independent cases. The average recognition rates are listed in Table 2. We only show the results from HMMs of 4 states 1 Gaussian mixture and 8 states 1 Gaussian mixture. The normalization only improves slightly (and sometimes hurts) the performance in the user dependent case, but it has significant impact in the user independent case. This actually confirms our expectation in Section 3.2 that the normalization helps more when there are higher in-class variations. In the user independent case, the time-derivative features V, W, and A benefit more from the normalization process, which also agrees with the variation ratios in Table 1. Note that the ratio of O only takes into account the "scaling" of rotation angles and doesn't include the variation of the absolute orientation. Therefore, the large improvement of O is partially contributed by the orientation offset (+15.0%) and the rotation angle normalization (+5.9%) with 4 states 1 Gaussian mixture HMMs. Increasing the number of states also gradually improves the performance, but the gain is less prominent for the features that already achieve high accuracy.

4.2. Evaluation of the Combined Feature Sets

Second, we evaluate the recognition performance with different combinations of the basic features, including the explicit spatial 3D (PV), implicit 6D (AW), explicit 6D (PO and PVO), and complete 6D ($PVOW$). These combinations correspond to the possible available tracking signals. The average recognition rates are plotted in Figure 1. We also list the numbers of AW and NPO in Table 2. In the user dependent case, over 99% accuracy is attainable with either implicit or explicit 6D feature sets. When we combine feature sets of different kinematic meanings together, we tie more "constraints" on each HMM state and make it more discriminative. However, the improvement becomes marginal at certain level, so we stop at the feature set $PVOW$, which contains up to the first order time derivative information.

In the user independent case, it's quite interesting that the accuracy of the left-handed testing set is higher than that of the right-handed testing set even though the recognizer is trained with right-handed users. This may be resulted from the unbalanced size of training sets (right-handed: 16; left-handed: 7). Once we enlarge the left-handed training set, the performance is expected to get closer to the right-handed case. However, based on our current results, the recognizer trained by right-handers is applicable to left-handers. We may further assume that the recognition of motion gestures is regardless of handedness, which is true for our gesture set.

4.3. Comparison with Our Previous Work

The sequences of random combinations drawn for training and testing sets are actually identical to our previous work [9] so that we can make direct comparison. In [9], the recognition results are obtained by using a linear classifier with extracted statistical features of either implicit or explicit 6D motion data, which are comparable to the feature set AW and PVO in the HMM case. Table 3 shows the performance comparison between the linear classifier and the HMM-based recognizer of 8 states 1 Gaussian mixture. In the user dependent case, the performance is almost the same. In the user independent case, our new approach outperforms by at least 6.6% for implicit 6D and 2.8% for explicit 6D in the absolute recognition rate. The best attainable accuracy of $PVOW$ is even slightly higher than PVO.

Table 2. The recognition rates with and without normalization (UD: user dependent case, UI.R: user independent case on right-handed users)

		P_o	P	O_o	O	V_o	V	W_o	W	A_o	A	AW	PVO
UD	(4 states)	95.88	96.23	97.45	96.76	97.84	97.88	96.78	97.26	97.58	97.03	98.98	99.17
UD	(8 states)	97.57	97.83	98.54	97.97	98.20	98.54	97.71	98.10	98.54	98.40	99.08	99.55
UI.R	(4 states)	85.13	87.38	64.42	85.32	73.64	87.65	63.75	75.40	62.15	80.33	87.30	95.12
UI.R	(8 states)	88.72	88.62	72.55	88.53	82.05	91.31	69.83	80.79	71.51	88.58	91.86	96.34

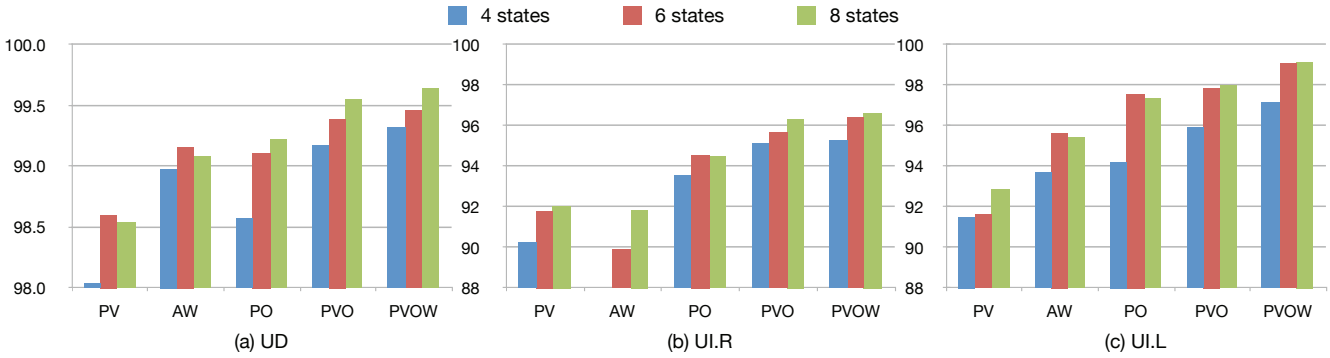

Fig. 1. The average recognition rates of combined feature sets. (a) UD: user dependent case. (b) UI.R: user independent case on right-handed users. (c) UI.L: user independent case on left-handed users.

Table 3. The performance comparison

	Implicit 6D		Explicit 6D	
	Linear	HMM	Linear	HMM
UD	98.80	99.08	99.59	99.55
UI.R	85.24	91.86	93.51	96.34
UI.L	78.58	95.43	96.99	98.00

5. CONCLUSION

In this work, we attempt to recognize motion gestures of both explicit and implicit 6D motion information, which includes the position and orientation in the global frame, and the acceleration and angular speeds in the device-wise coordinates. Both user dependent and user independent cases are addressed. In general, motion gestures are defined by the spatial trajectory. This also holds true in our gesture set with two exceptions that should be defined by the change in orientation. From the motion data, we derive five basic feature vectors: P_o, V_o, O_o, A_o, and W_o for HMMs. Each feature vector has its own kinematic meaning and contains the clue for gesture recognition. We also propose the normalization process to make the feature set "scale" invariant to overcome the huge in-class variations in the user independent recognition. Finally, we put together different basic feature vectors and investigate the discriminating power of the combined feature set.

Based on the results, all the five basic feature vectors are discriminative to certain level. In the user dependent case, the combined feature sets of either implicit or explicit 6D motion achieve accuracy higher than 99%. We also show that time-derivative features have higher variations among users and hence benefit more from the normalization. With complete 6D information, the recognition rate is above 96% in the user independent case. In the future, we hope to replace the push-to-gesture scheme with automatic gesture spotting.

6. REFERENCES

[1] Hyeon-Kyu Lee and Jin H. Kim, "An hmm-based threshold model approach for gesture recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, pp. 961–973, Oct. 1999.

[2] Jani Mäntyjärvi, Juha Kela, Panu Korpipää, and Sanna Kallio, "Enabling fast and effortless customisation in accelerometer based gesture interaction," in *Proc. of the 3rd Int'l Conf. on Mobile and ubiquitous multimedia*, 2004, MUM '04, pp. 25–31.

[3] Christoph Amma, Dirk Gehrig, and Tanja Schultz, "Airwriting recognition using wearable motion sensors," in *Proc. of the 1st Augmented Human Intl. Conf.*, 2010, AH '10, pp. 10:1–10:8.

[4] Jiayang Liu, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan, "uwave: Accelerometer-based personalized gesture recognition and its applications," *Pervasive and Mobile Computing*, vol. 5, no. 6, pp. 657 – 675, 2009, PerCom 2009.

[5] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li, "Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes," in *Proc. of UIST '07*, 2007, pp. 159–168.

[6] Sven Kratz and Michael Rohs, "Protractor3d: a closed-form solution to rotation-invariant 3d gestures," in *Proc. of the 16th Int'l Conf. on Intelligent user interfaces*, 2011, IUI '11, pp. 371–374.

[7] Dean Rubine, "Specifying gestures by example," *SIGGRAPH Comput. Graph.*, vol. 25, pp. 329–337, Jul. 1991.

[8] M. Hoffman, P. Varcholik, and J.J. LaViola, "Breaking the status quo: Improving 3d gesture recognition with spatially convenient input devices," in *Virtual Reality Conference (VR10)*, Mar. 2010, pp. 59 –66.

[9] Mingyu Chen, Ghassan AlRegib, and Biing-Hwang Juang, "A new 6d motion gesture database and the benchmark results of feature-based statistical recognition," in *IEEE Int'l Conf. on Emergin Signal Processing Applications*, Jan. 2012.