A Handwritten Character Recognition System Based on Acceleration

Chun Yuan* Shiqi Zhang* Zhao Wang**
* Information Science and Technology Division,Graduate School at Shenzhen, Tsinghua University,Shenzhen, China
** Institute of Software, School of Electronics Engineering and Computer Science, Peking University, Beijing, China;
* yuanc@sz.tsinghua.edu.cn

Abstract
In this paper, an handwritten character recognition system based on acceleration is presented. The character recognition system using a 3-dimensional (3D) accelerometer, includes three procedures: original signal detection, signal processing (preprocessing and quantization) and recognition/classification. In quantization procedure, Trajectory Orientation (TO) and Curve Feature (CF) are adopted and compared. In recognition procedure, Fully-connected Hidden Markov Model (HMM) and Left-Right HMM are both implemented and compared. The system, in the recognition of 10 Arabic numerals, achieves the Correct Rate(CR) of 99.05% and the Total Correct Rate (TCR) of 94.76% .

Keywords:
Character Recognition; Accelerometer; Quantization; Curve Feature;

1. Introduction
Pattern recognition techniques based on inertial devices have been thriving for years. In the last century, inertial devices including accelerometers and gyroscopes were mainly used in the army and some luxurious products such as the car safety airbag system. However, in recent years, inertial devices have been much more widely used in civil products because of the continuously decreasing price and the higher performance. Like "miCOACH" from Adidas [1] and "nike+" from Nike [2], many products, especially sports goods, have been developed using inertial devices to help people improve the standard of living.

Traditional Human Computer Interaction (HCI) mainly focuses on keyboards, which are efficient enough when both the operations and the tasks are simple. As the pace of life is increasing rapidly, the traditional HCI can hardly satisfy users' requirements in some situations. Instead, ubiquitous computing devices are needed, which must be tiny, effortlessly portable and constantly available. Consequently, new technologies on HCI were flourishing at the end of the last century, such as speech recognition [3], vision-based gesture recognition [4, 5] and tablet types of devices. These technologies can help users get rid of keyboards and all of them are much more effective and efficient than the traditional approaches [6]. Compared to these technologies, pattern recognition based on inertial devices is relatively new, but there are some inherent advantages of this technology. For example, to control a computer or a mobile phone by an inertial devices, what we need is to shake a remote controller or even the computing device directly, provided that its size is small enough. By doing so, we can avoid the annoying disturbance in public places where speech recognition or gesture recognition systems served. Additionally, in the field of medicine, pattern recognition based on inertial devices can assist doctors by correcting abnormal habits like human abnormal gait [7], and can even make real time monitoring of daily activities and physiological parameter feasible in practice [8].

Many algorithms for gestures or characters recognition were studied[9], such as Hidden Markov Model (HMM), Bayesian Networks (BN) and Dynamic Time Warping (DTW). Since the object to be analyzed could be either discrete or continuous, there exist two kinds of HMM, which are Discrete HMM (DHMM) and Continuous HMM (CHMM). Most researchers use DHMM to find the most probable activity state [10,11,8], which is a lightweight one for math calculation. In [12], CHMM and DTW are adopted in the character recognition system and the HMM Tool Kit (HTK) is used to perform the recognizing. However, both CHMM and DTW can increase the computational burden compared to DHMM, which would be infeasible for the major potential application field of lightweight embedded systems. Moreover, because all the sampled signals from accelerometer are discrete, it would be straightforward to use them without transforming them into the continuous domain. So we choose DHMM instead of CHMM in despite of its great utility.

The rest part of this paper is organized as follows. Firstly, the hardware system is introduced to present a tangible impression. Secondly, the signal processing part is presented, including both preprocessing and feature extraction, which is the highlight of this paper and three different kinds of feature extraction methods are presented respectively. Thirdly, the training and the testing process of handwritten characters using HMM is described. Experimental results are presented in Section 4 and this paper is concluded in Section 5 with some directions for future work.

2. Hardware System
The 3D accelerometer MMA7260T we use is made by Freescale Semiconductor, which has selectable measuring range of 1.5g/2g/4g/6g, a high sensitivity of 800mV/g at 1.5g and a small Quad Flat Package (QFP) of 6mm × 1.45mm. MC9S08QG8 is an 8-bit MCU made by Freescale Semiconductor too. It has many excellent features, such as 20-MHz working frequency, 8 Kbytes Flash, 512 bytes RAM, 8-channel 10-bit A/D converter and a serial communication interface module. The 3D accelerometer is fixed on 3D Acc. Module and then this module is connected with the Evaluation Card (CT298) as a pluggable device as Fig. 1

shows.

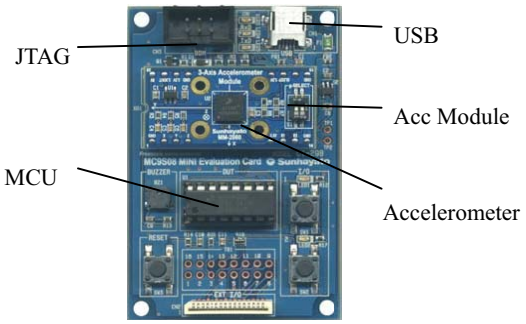


Figure 1. Mini evaluation card with 3D accelerometer module

The USB/COM connector is used to convert the port from COM to USB physically with the signals still recognized by PC as COM. Although the mini evaluation card has a size of only 47mm × 72mm, which may still too big to be embedded into pen-like devices, mobile phones or other kinds of handheld devices. Needless to worry about that, because once the accelerometer and MCU together are used in commercial products, the components like JTAG, serial port and all the buttons would be discarded or laid inside the PCB, thus the size would be reduced in great deal. The data from evaluation card are transmitted through a serial port line to PC terminal, where we use the tool of Matlab on Windows OS to finish the analysis as Fig. 2 shows.

3. Algorithm

In this section, we would introduce the process of character recognition based on accelerometer, which includes effective section extraction, zero bias compensation, filtering, feature extraction and classification. The flow chart of this system is show as Fig.3. The first three parts aim to prepare data or usable information for the last two parts, belong to preprocessing. Considering that no matter in which direction user draws the characters, they must be drawn in a 2-demensional plane, so theoretically we could discard the acceleration in z-axis. If so, the heavy burden of calculation could be released, because there exist much sine and cosine calculation in 3D space transformation. In this case, the sensor's center of gravity could be moved in one plane vertical by z-axis. If not, the output data of accelerometer would be jumbled by coordinate transition. In this way, all the operations are simplified from 3D to 2D.

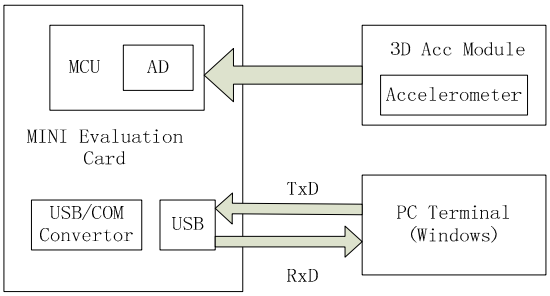


Figure 2. Skeleton of the hardware system

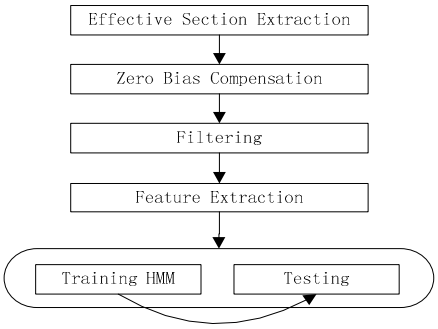


Figure 3. Flow chart of algorithm

3.1 Preprocessing

The acceleration signals are transmitted as data stream from mini evaluation card to PC terminal by serial port line. Preprocessing is the first step to divide the stream into many sections and make sure that every section contains and only contains the information of one character. To extract an effective section, the only task is to find out the beginning and the end of a character as precisely as possible.

When the accelerometer stays motionless, the accelerations in both x and y coordinate would be stable. To detect the beginning of a handwritten character, we set a threshold acceleration value according to experience. If the acceleration change in either x-axis or y-axis surpasses this threshold, it would represent a beginning of a handwritten character. This threshold should be carefully determined; otherwise it might be activated by unexpected small or big disturbance. Similarly, the end of a character is defined that acceleration keeps invariable in both x-axis and y-axis for a short time. Following equations present the method to detect the beginning and the end points.

$$\begin{cases} \left|a_x(T+1)-a_x(T)\right|<\theta \quad and \quad \left|a_y(T+1)-a_y(T)\right|<\theta \quad T<T_0 \\ \left|a_x(T+1)-a_x(T)\right|>\theta \quad or \quad \left|a_y(T+1)-a_y(T)\right|>\theta \quad T=T_0 \end{cases} \tag{1}$$

$$\begin{cases} \left|a_x(T_{end}+1)-a_x(T_{end})\right|>\phi \quad or \quad \left|a_y(T_{end}+1)-a_y(T_{end})\right|>\phi \\ \left|a_x(T+1)-a_x(T)\right|<\phi \quad and \quad \left|a_y(T+1)-a_y(T)\right|<\phi \end{cases} \quad for \quad T=T_{end},T_{end}+1,\cdots,T_{end}+N \tag{2}$$

where, T denotes time or the sequence number of acceleration vector, T_0 is the sequence number of beginning and could be selected by continuously "filtering" acceleration values, T_{end} is the sequence number of the end, π is the acceleration threshold at the beginning, θ is the acceleration threshold of the end and N denotes the holding time of the end.

In this way, a matrix A could be constructed by the 2 × 1 acceleration vectors from T_0 to T_{end}, and it has and only has all the acceleration data of a handwritten character in both x-axis and

y-axis.

$$A=\begin{bmatrix} a_x(T_0) & a_x(T_0+1) & \cdots & a_x(T_{end}) \\ a_y(T_0) & a_y(T_0+1) & \cdots & a_y(T_{end}) \end{bmatrix} \quad 3$$

The manufacturing technology of Micro-electromechanical System (MEMS) has been improved greatly in these years, but there still exist many technical obstacles which are very hard to overcome like random drift and noise. The output of an accelerometer is a constant voltage called zero bias when it is stationary in every direction. The constant is a summation of the gravity acceleration and a drift. There are many unexpected environmental factors make drift keep changing all the time, like temperature and electric field. The drift could be regarded as a random function of time and constant within a short and certain period of time. The module of zero bias compensation is used to discard the zero bias of acceleration data. In following equations, a0 is the average acceleration during the time from T_0 to T_{end}, and the values in x-axis and y-axis are not distinguished here for their similarity.

$$a_0=\frac{1}{T_{end}-T_0}\sum_{i=T_0}^{T_{end}-1}a_r(i) \quad 4$$

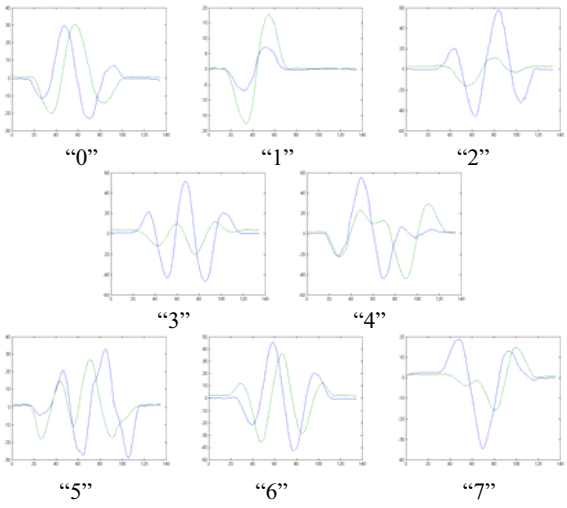
$$a(i)=a_r(i)-a_0 \quad i=T_0,\cdots,T_{end} \quad 5$$

where, a_r(i) is the original acceleration value in either x-axis or y-axis and a(i) is the value after zero bias compensation.

The accelerometer output is a serial of discrete acceleration values. Due to accelerometer structures and random vibrations, there exists some noise in the output. Average Filtering is one of the simplest methods of filter algorithm.

$$a(j)=\frac{1}{M}\sum_{i=j}^{j+M-1}a(i) \quad j=T_0,\cdots,T_{end}-M \quad (6)$$

where, a(j) is the acceleration value in either x-axis or y-axis, M is a parameter to adjust the strength of filtering.
For the ten Arabic Numerals, acceleration value curves are drawn as a function of time in Fig.4. Green curve (light) is the acceleration in x-axis and blue (dark) is for y-axis.



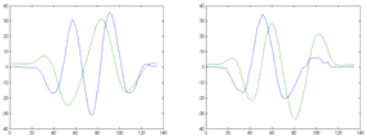


Fig. 4: Acceleration curves in x-axis and y-axis by time

3.2 Feature Extraction

Feature extraction is one of the most important parts for every kind of pattern recognition, as well in the handwritten character recognition based on 3D accelerometer. As we analyzed above, we will recognize characters by the acceleration information directly without integrating it once or twice. In the field of vision-based character recognition, researchers have found many effective methods of feature extraction, and although all of them process the data in position domain instead of acceleration domain, we could still use these methods for reference. In [13], they developed a system to recognize characters using accelerometer, and define feature points as local minima or local maxima points where the signal values are minimal or maximal within some time interval. After that, they use BN to do recognition. We call this method the name of Extremum Points. In [5], they develop a method to extract feature in vision-based character recognition by the trajectory's orientation information. In this section, we will introduce the methods of Trajectory Orientation (TO) and Curve Feature (CF) in turn.

3.2.1 Trajectory Orientation

We draw a trajectory curve by acceleration information directly. Therefore, we regard the sequence of acceleration vectors as a serial of points in 2D coordinate space, and an acceleration trajectory curve is got by connecting the points. The x-axis is the acceleration in x-direction and the y-axis is for y-direction. In this way, Fig.4 is redrawn into Fig.5.

To extract the orientation information, it is necessary to divide the 2D coordinate space into some parts. There are theoretically infinite ways to achieve this task, but some constraints exist at the same time as follows. First, the space couldn't be divided into too many parts, because the sampling rate of accelerometer is a constant value, and if so, the quantization would be imprecise. Second, the number of divisions can't be too small, or the acceleration information would be largely lost at this step. Third, the method to divide the space must reflect the features of curves. For example, as Fig. 5 shows, there are many circles in the curves, so we should use the angle value instead of the distance in quantization method, if all the points are put into polar coordinates system. Finally, considering the range of accelerometer, all points in these trajectories must be in a rectangle, so the task is to find out a way to divide this rectangle. Fig.6 represents a method of quantization based on orientation in polar coordinates system. When the trajectory curve passes through a division, it would produce one digit (without iteration) or a serial of one digit (with iteration) and details of quantization methods will be presented in the section of HMM.

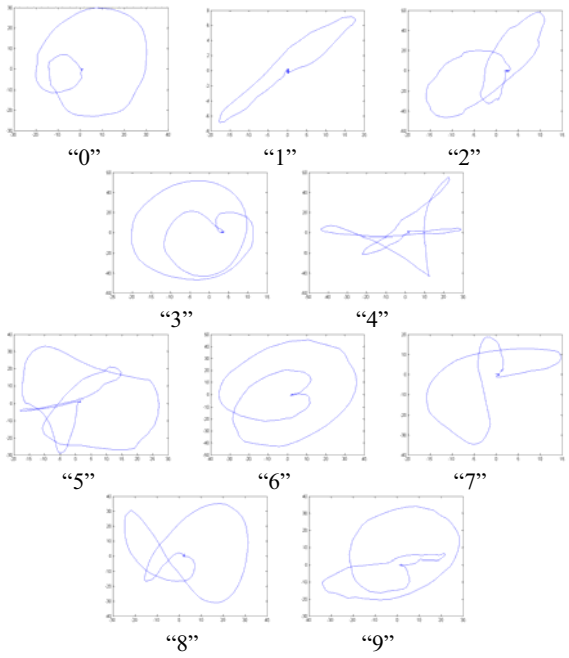


Figure 5. Acceleration Trajectory Curve

In this way, for every handwritten character we could extract a serial of numbers according to the acceleration trajectory. The numbers contain the angle information of the handwritten character in polar coordinates. Besides, it is noted that there are many repetitions for the numbers after quantization, so we would also prove that the repetitions have much important information and are helpful for recognition. In fact, if we discard all the reputations, any one of the quantized numbers would be always different from the two numbers beside that.

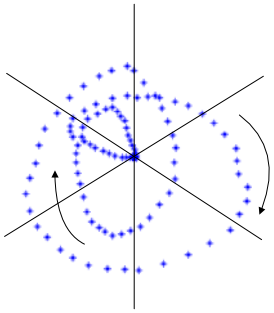


Figure 6. Quantization in Polar Coordinates System

3.2.2 Curve Feature

For on-line character recognition in acceleration domain, every two points near each other contains the information in which direction the acceleration changes. As Fig.7 shows, the coordinates of every point denote the acceleration values in x-axis and y-axis at that time, and according to the values we could calculate out the absolute value and direction of the acceleration vector at that time. Like TO, the vector's direction is quantized by 6 numbers, so in this way we could extract a serial of numbers from the curve. In fact, the vector could be regard as the acceleration point's acceleration vector, or instantaneous changing value of acceleration.

3.3 Discrete Hidden Markov Model

In this part, we use software, Discrete Hidden Markov Model Simulation (HMMSim) to simplify the calculation [14],.

3.3.1 Hidden Markov Models

Markov model is a mathematical model of stochastic process where these processes generate a random sequence of outcomes according to certain probabilities [15, 16]. It is trainable and the underlying stochastic process is unobservable, so we call it hidden markov model. The challenge is to determine the hidden parameters from the observable parameters. The extracted model parameters can then be used to perform further analysis, such as pattern recognition applications. An HMM can be considered as the simplest dynamic Bayesian network and all the mature theories on dynamic BN could be used in HMM directly. Both Continuous HMM (CHMM) and Discrete HMM (DHMM) are widely used in Pattern Recognition and many other fields. For DHMM, the observable states and invisible states are all discrete.

An HMM is a collection of finite states $S = S_1, S_2, \cdots, S_N$ interconnected by transitions. Each state has a number of distinct observation symbols $V = v_1, v_2, \cdots, v_M$ corresponding to the physical output of the system [7]. An HMM can be specified by the following notation:

$$\lambda = (A, B, \pi) \qquad 7$$

where, the state transition probability distribution $A = \{a_{ij}\}$, and a_{ij} denotes the probability of transition from S_i to S_j. The observation symbol probability distribution at the given state S_i, $B = b_i(x)$. The initial state probability distribution vector $\pi = \pi_i$.

A is an $N \times N$ matrix, B is an $N \times M$ one and π is a $1 \times N$ vector. All of them should be initialized by experience at first. Number N represents the number of states in the model and number M is of distinct observation symbols per state. In [10], HMM are divided into three topologies: Fully Connected (Ergodic) model, Left-Right model and Left-Right Banded model. In this paper, we use the first one to initialize the models, both N and M are set to 10 and all the units in A, B and π are set to 0.1. Although these values are imprecise initially, they would keep approaching a perfect outcome in the process of numerous iterative calculations. In fact, the initializing parameter and the number of states do not have a significant effect on the gesture recognition results, because during the optimizing process the parameters would be absolutely not the same as the initialized ones [17]. Initialize the models in the format of Left-Right model or Left-Right Banded model could accelerate the training and classifying, but may not have great use in the results, refer to Fig 8.

3.3.2 Classification

Generally, there exist three problems for HMM to solve in an application system. First, when we have a number of trained HMMs describing different systems, and a sequence of observations, how to chose an HMM most probably generated the given sequence, or calculate out the probability of every HMM for the observations. The second problem is to find the most probable sequence of hidden states given some observations. The last one is to generate a HMM from a sequence of observations.

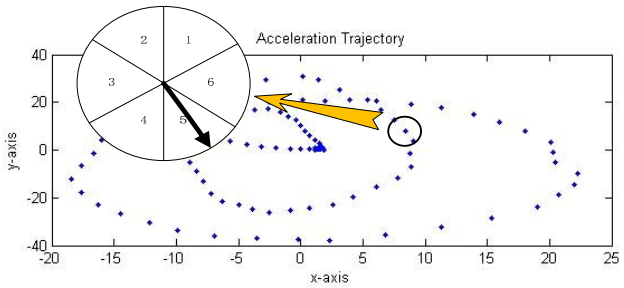


Figure 7. Extracting information from the adjacent points in Curve Feature

By HMMSim, we could find the parameters of the HMM which maximize the probability to happen upon certain sequences of observations (HMM Problem 3) and estimate the probability to happen upon a particular sequence of observations (HMM Problem 1). Fig.9 is the flow chart of HMM training and testing in detail.

At the beginning, an HMM file is loaded, which contains matrix A, B and vector π in it. This is both an initialization file and the basis of all the iterations. Then for every Arabic numeral, there are several patterns sampled already to train HMMs. After the step of training and saving, we get ten trained HMMs. For every trained model, using Forward Algorithm (FA) we can calculate out a probability value of an observation sequence and this is just the HMM problem 1. The final step is to compare all the values of probability and the Arabic numeral corresponding to the biggest value is the outcome of classification.

4. Experimental Results

Experiments are designed to test the performance of the two feature extraction methods TO and CF. One goal is to find out respective characteristics in the system, the other is to verify which one is better between the Full-Connected HMM and the Left-Right one, in on-line character recognition based on accelerometer. From this point of view, we have to set up every combination of the four feature extraction methods and the two HMM structures, so groups of experiments are organized and each one contains the experiments results of all the ten Arabic numerals in order to make a comparison between each other. The methods of feature extraction are realized in Matlab Language and we use HMMSim to construct HMMs 10. Both Matlab and HMMSim have the function of converting programs into C++ language, so it wouldn't be a tough task to compile this system into executable file on Windows.

We firstly set up a system with the feature extraction of TO. To test which structure of HMM is better between Left-Right HMM structure and Full-Connected. LR structure is implemented at first. All the iterations of numbers, which are extracted from the original trajectory by TO as Fig.7, are discarded according to the requirement of LR.

Without the iterations, the extracted feature number could only jump from one state to another, and it is impossible to jump from one state to itself. We use the numbers with iterations directly to make a comparison. By the experiment results of both the two feature extraction structures, we draw a table as Tab. 1, where x-axis and y-axis denote the numbers we write and the recognized numbers by this system.

From the tables, we can see that about half the results are incorrect for numeral "2" and "3" when the HMM structure of LR is used. System with such a low performance is impractical and unreliable at all. However, when we use the HMM structure of FC, the test results are improved in great deal. For nearly every Arabic numeral, FC could achieve a higher correct rate than LR.

Table 1. Test results by feature extraction method "TO"

(a) HMM structure of Left-Right by TO

Out \ In	0	1	2	3	4	5	6	7	8	9
0	20		7	11						
1		19								
2			12							
3	1		2	10						3
4		1			20					
5		1				20				
6							18			
7								20		
8									21	
9										18
None					1	1		1		

(b) HMM structure of Full-Connected by TO

Out \ In	0	1	2	3	4	5	6	7	8	9
0	21									
1		19								
2			20							
3			1	21						
4		1			19		1			1
5		1				19				
6							20			
7								21		
8									21	
9							2			17
None					2	2	1			3

Three definitions are brought out to show the performance of different methods and structures.

$$RR(N) = \frac{R(N)}{T(N)} \quad 8$$

$$CR(N) = \frac{C(N)}{R(N)} \quad 9$$

$$TCR(N) = \frac{C(N)}{T(N)} \quad 10$$

where, *RR* denotes the Recognition Rate and no matter the recognition is correct or not, *CR* denotes the Correct Rate in the recognized numbers and *TCR* denotes the Total Correct Rate. Besides, we define *R* as the number of Arabic numerals that are recognized (maybe correct or not), *C* as the number of corrected recognized Arabic numerals and *T* as the total number that user has input by the system. *N* denotes the number we test, which could be any number between 0 and 9.

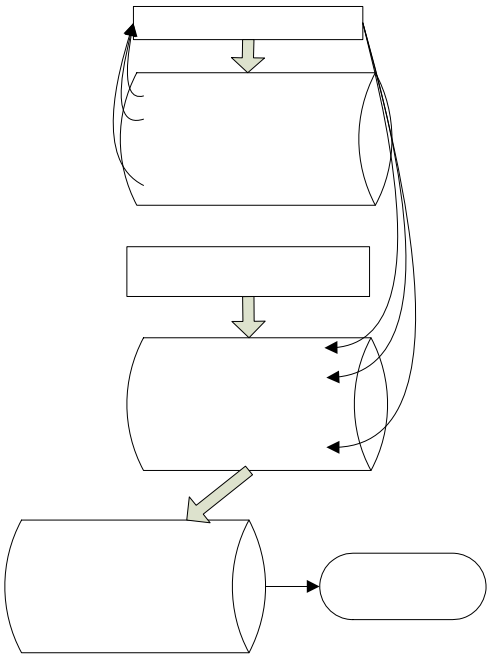


Figure 8. Flow chart of classification by HMM

The reason to define *RR*, *CR* and *TCR* is that in different conditions it is not definitely whether *CR* or *TCR* is more important. For example, if a user only wants a stable system that the recognized results are highly reliable, he/she would need the system with higher *CR*. Although the output of high *CR* system could be more reliable and users needn't delete the incorrectly recognized characters, maybe the users have to repeat the characters that the system can't recognize. In other hand, the systems with high *TCR* have the characteristic of the highest correct probability for any input character and the system with very high *TCR* would be a perfect system. The system with high *RR* can recognize mostly all the characters users input, and the system with high *CR* could be sure about the correctness of its recognition. Thus, it would be the perfect system with both very high *RR* and *CR*, which certainly have very high *TCR*.

we find that some Arabic numerals like "0" and "8" are easier to recognize because we could extract more trajectory features comparing with other numerals. We calculate out the RR, CR and TCR for every method according to their definitions as Tab. 2 shows.

Table 2. Statistics by feature extraction methods and HMM structures (%)

	LR-TO	FC-TO	LR-CF	FC-CF
RR	98.57	96.19	90.01	99.05
CR	86.19	98.02	93.22	95.67
TCR	84.76	94.29	83.81	94.72

As the analysis above, both *CR* and *TCR* are important criteria to evaluate a system, and in different occasions, we will think much of different ones. For one thing, the *CR* of Left-Right Trajectory-Orientation method is the highest and gets very close to a hundred percent, so we could regard the recognition results of this combination feasible and reliable. In the case of requiring highly reliable recognition results, but not caring much about repeating the input characters, FC-TO is the best choice. For another thing, due to the highest *RR*, although the *CR* of FC-CF is lower than FC-TO, the TCR of FC-CF is larger than any other kind of combination of feature extraction and HMM structure methods anyway.

5. Conclusion and future work

Based on the high performance 3-D accelerometer, an on-line recognition system was designed to recognize handwritten Arabic numerals in the acceleration domain instead of the position domain, in order to reduce the accumulated error during integrating. In the proposed system, characters to be recognized were represented by discrete points instead of continuous lines and the quantized numbers were analyzed by the Discrete HMM.

In the similar manner, we could construct systems to recognize all of the 26 characters in the alphabet or the sign language, and the latter's complexity is much higher. In fact, a lot of research has already been conducted to recognize sign languages with some positive results. However, most of the current systems require too many sensors and the recognition algorithms are also very complex. So it is another important direction to simplify the character recognition systems based on inertial sensors.

[1] http://www.micoach.com/

[2] http://nikeplus.nike.com/nikeplus/

[3] Renado De Mori, "Spoken language understanding: a Survey", *2007 IEEE Workshop on Automatic Speech Recognition and Understanding*, 9-13 Dec. 2007

[4] Hiromichi Fujisawa, "Forty years of research in character and document recognition:an industrial perspective", *Pattern Recognition* 41 (2008) 2435-2446

[5] Henry S. Baird, Daniel Lopresti, "Robust Document Image Understanding Technologies", *HDP' 04 ACM*, November 12, 2004, Washington, DC, USA

[6] Sung-Jung Cho, Jong Koo Oh and Won-Chul Bang, "Magic wand: a hand-drawn gesture input device in 3-D space with

inertial sensors" *Proceedings of the 9th International Workshop on Frontiers in Handwriting Rcognition*, 2004

[7] Meng Chen, Bufu Huang and Yangsheng Xu, "Human Abnormal Gait Modeling via Hidden Markov Model", *Proceedings of the 2007 International Conference on Information Acquisition*, Jeju City, Korea, July 9-11, 2007

[8] Jin He, Huaming Li and Jindong Tan, "Real-time Daily Activity Classification with Wireless Sensor Networks using Hidden Markov Models", *Proceedings of the 29th Annual International Conference of the IEEE EMBS*, Lyon, France, August 23-26, 2007

[9] Rejean Plamondon, Sargur N. Srihari, "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, VOL 22, NO.1, Jan 2000

[10] Mahmoud Elmezain, Ayoub Al-Hamadi, "Gesture Recognition for Alphabets from Hand Motion Trajectory Using Hidden Markov Models", *2007 IEEE International Symposium on Signal Processing and Information Technology*, 2007

[11] Jian Kang Wu, Liang Dong and Wendong Xiao, "Real-Time Physical Activity Classification and Tracking using Wearable Sensors", *2007 6th International Conference on Information Communication and Signal Processing*, Singapore, pp. 1-6, Dec. 2007

[12] Sung-Do Choi, Alexander S. Lee, and Soo-Young Lee, "On-Line Handwritten Character Recognition with 3D Accelerometer", *Proceeding of the 2006 IEEE International Conference on Information Acquisition*, August 20-23, 2006, Weihai, Shandong, China

[13] Eun-Seok Choi, Won-Chul Bang, "Beatbox Music Phone: Gesture-based Interactive Mobile Phone using a Tri-axis Accelerometer", *IEEE International Conference on Industrial Technology,* pp. 97-102, Dec. 14-17, 2005

[14] J. F. Velez, A. Sanchez and A. B. Moreno, "HMMSim: An Interactive Simulation Tool for Teaching Discrete Hidden Markov Models", *4th International Symposium on Computer Science Education,* Vigo, Spain, November 22, 2002

[15] Sushmita Mitra and Tinku Acharya, "Gesture Recognition: A Survey", *IEEE Transactions on System, Man and Cybernetics,* Vol.37, NO.3, May 2007

[16] N. Liu, B. C. Lovell and P. J. Kootsoolos, "Model Structure Selection & Training Algorithms for a HMM Gesture Recognition System" , *International Workshop in Frontiers of Handwriting Recognition*, PP. 100-106, October 2004

[17] VM Mäntylä, "Discrete hidden Markov models with application to isolated user-dependent hand gesture recognition", Espoo 2001, Technical Research Centre of Finland, VTT Publications 449. 104p