Development of a Single 3-axis Accelerometer Sensor Based Wearable Gesture Recognition Band

Il-Yeon Cho[1], John Sunwoo[1], Yong-Ki Son[1], Myoung-Hwan Oh[1], and Cheol-Hoon Lee[2]

[1] Digital Home Research Division, Electronics and Telecommunications Research Institute
Daejeon, Korea
{iycho, bistdude, handcourage, mhoh}@etri.re.kr
[2] System Software Laboratory, Department of Computer Engineering
Chungnam National University, Daejeon, Korea
clee@cnu.ac.kr

Abstract. A daily-wear wearable system is one of the most convenient mediums in practical application scenario of transferring various information data or services between two users as well as between a user and a device. To implement this service scenario, we chose to develop a wearable forearm mounted accelerometer based input system. A set of gesture commands was defined by analyzing intuitive forearm movements. A hardware system and software recognition engine that utilizes the accelerometer sensor data to recognize the gesture commands were implemented and tested. This paper describes the development techniques of a wearable gesture recognition system. It also includes discussions of software and hardware design and how variations in these affected gesture recognition rate by analyzing experimental results from the actual implementations.

Keywords: Wearable system, gesture recognition band, accelerometer sensor

1 Introduction

Wearable devices are well-known for their use in specialized fields such as medicine-art, sports, gaming, and sign language recognition [1]. However, they can also be used everyday to increase the productivity and convenience of our normal life. One currently commonplace example would be when dealing with information in an electronic format. We often encounter situations where someone asks another person for a particular data file. Such files might be stored on a USB flash disk or CD-ROM and perhaps carried in our pockets or briefcases. Without accessing a computer, it is impossible to use these devices. However, wearable computers have the potential to achieve this task quickly, easily and seem lessly. For example, one user could make a pointing gesture to trigger a file transfer to another wearable system wearer. The advantage of this approach is that we do not have to look for computers to do the task; instead, the wearable system can recognize intuitive gestures to do the task for us.

We can broaden this service scenario to other diverse situations so that the wearable system can interact with various objects like multimedia appliances. Based on this scenario, we targeted the development of the wearable system that can be operated by intuitive forearm gestures using an accelerometer sensor. One advantage of using an accelerometer sensor-based wearable system is its unrestricted operating environment where extensive vision-based device for tracking gestures are not required. By developing specific and customized gesture commands for the scenario, we suggest that we can avoid using more than one accelerometer sensors, which will reduce power consumption [2]. In software, there are intelligent algorithms that utilize neural networks or Hidden Markov Model (HMM) to power gesture recognition engines [3-7]. They have been used widely for recognizing human gestures, however they require reasonable amounts of memory and processing power and are perhaps not suitable for a low-power wearable system. This prompted us to avoid the use of such algorithms and develop a light-weight robust engine customized for our service scenario defined.

The paper begins with an overview of related work discussing a number of gesture recognition devices in Section 2. The service scenario that we've targeted for our gesture recognition device is presented in Section 3 followed by the definition and evaluation process of the gesture commands in Section 4. Section 5 will discuss the development of a customized software gesture recognition engine and the hardware design process that includes the determination of optimal accelerometer sensor location. Discussions from the final evaluation process will be in Section 6 and the paper concludes in Section 7.

2 Related Work

Methods of recognizing gestures are widely investigated using various sensing devices and software implementations [1-12]. It is known that gesture recognition algorithms such as neural networks and the HMM model technique are effective. However, most of these systems deal with vision based recognition, and are subject to environmental restrictions such as that they are unsuitable in scenarios where the background environment is changing as the user moves in real world [1].

One previous system uses accelerometer sensors placed on gloves and represents the most directly relevant work. The accelerometer sensors were placed on every finger and both wrists to monitor hand shape without the use of cameras [13]. Avoiding vision-based techniques could give more mobility and robustness, however the gesture glove could also lead to problems if we want to use it for daily use because it covers all five fingers and palm area obstructing the normal use of the hand [1][9].

Rekimoto's 'GestureWrist' seemed to closely relate to our study in terms of the form factor by adopting a wristwatch type device that enables a hands-free operation on both hands [9]. The 'GestureWrist' mainly uses the cross-sectional shape of the wrist to detect hand motions, as well as a 2-axis accelerometer sensor embedded on the wristwatch to detect inclination of the forearm. It also notes other related gesture based input devices such as [10-12] are not sufficiently unobtrusive for daily wear. Unfortunately, use of a 2-axis accelerometer sensor would prevent detecting other various forearm movements other than inclination.

Similar service to what we've targeted for our study can be seen in work by Khotake [19]. The 'InfoStick' is a small handheld device that enables a drag-and-drop operation by pointing at the target objects by using a small video camera, buttons and a microprocessor [19]. Although the results demonstrated a positive interaction technique, it has environmental restrictions because it recognizes objects with the camera, and the device had to hold by one hand which prevented the hands-free operations.

In this work, we developed a wearable device using gesture defined by intuitive forearm movements that were not considered in the previous research. From these movements, we define gesture commands which result in development of a customized recognition engine. Considering mobility is also important for wearable devices. We want to ensure our device is wearable anytime, anywhere, supports hands-free operations and uses the minimal possible sensors (requires only one 3-axis accelerometer sensor in this study) that would help elongate system's run time by consuming low power.

3 Application Scenario and Wearable System

Our application scenario involves a daily-wear wearable gesture recognition system can effectively command information, data or services to be transferred to other wearers or devices by making an intuitive pointing gestures. Data or services on the targeting devices can also be controlled using intuitive gesture commands. We argue that a wearable band type of gesture recognition device would be greatly beneficial for such activities. We defined a scenario for dealing with multimedia services:

A wearer named 'Ashley' navigates through some movie icons and selects one of them to watch a movie through her Head-Mounted-Display (HMD). She can control the volume or skip chapters of the movie as she like. Ashley's friends, 'Brandon' and 'Christopher' show up to see Ashley. They get interested in what she is watching. Brandon and Christopher both ask Ashley to watch the movie with her. Ashley intuitively points the display device (such as television) near her so that everybody can watch the movie (Figure 1-a). Ashley adjusts the volume remotely by making a gesture. Brandon and Christopher have to go back home before the movie ends. Again, Ashley intuitively points at Brandon and Christopher, one at a time to transfer the movie file or the website link that directs to the movie so that they can watch it later (Figure 1-b).

Note that the scenario can be extended to handle any general file and services. Also generalized transfers between devices are possible: a television to a digital frame, a home audio to a car audio system, a display device to a photo printer. However, we have selected the scenario of dealing with a movie service for this paper in order to achieve maximum demonstration effect because a movie can be seen easily with relatively simple setup of supporting devices.

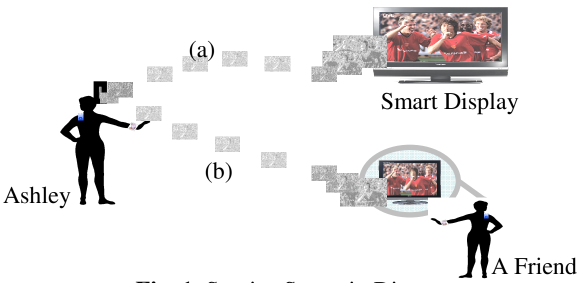


Fig. 1. Service Scenario Diagram

4 Defining Gesture Command

Based on our application scenario, we have defined 12 commands designed to be sufficient to control general multimedia appliances. Note that most of the commands can be interpreted differently according to applications they are being used to control. It is also possible that combinations of two or more gesture commands result in more complex compound commands.

Each command was then mapped to forearm gestures by considering a human's intuitive gestures used to make each operation in the real world. For example, the 'Device Selection' command is based on the act of pointing towards something, 'Select' resembles marking something important within a circle, the 'Left' gesture command is when someone tries to drag an object from right to left, 'Up' is related to how someone tries to pick up an object from ground, and 'volume up-continuously' is made by considering the gesture when we make when we adjust the volume on an audio system by rotating a circular knob. Each command was made with a counterpart; a command which resulted in the opposite action. While we were defining the gesture commands, we were also evaluating them to see how intuitive they were for various people.

Table 1. Defined gesture table.

Commands	Gestures	Commands	Gestures
Device Selection /Data Transfer		Enter/Select/Play	
Device Cancel		Esc/Cancel/Stop	
Left/Rewind/previous		Volume up (1 unit)	
Right/Fast forward/next		Volume down (1 unit)	
Up/Continue		Rotate right/Menu Navigation /volume up (continuously)	
Down/Pause		Rotate left/Menu Navigation /volume down (continuously)	

5 Implementation of Hardware and Software

As we began the hardware and software implementations that could recognize the 12 gesture commands defined in the previous section, we investigated the use of an accelerometer sensor by utilizing one of the development sensor modules that includes Kionix KXM52-1050 tri-axis accelerometer sensor shown in Figure 2. The evaluation module includes one Kionix KXM52 tri-axis accelerometer sensor and an Analog-to-Digital Converter (ADC). It has the accelerometer sensor packaged in a 5x5x1.8mm that detects acceleration and generates an analog voltage which is proportional to the acceleration. The analog value then converts to a digital value resulting in vector consists of x, y, z values.

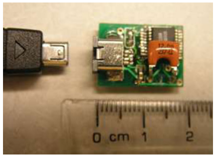


Fig. 2. Kionix KXM52 tri-axis accelerometer evaluation module [14]

In order to observe the characteristics of the sensor module and investigate how we could utilize the sensor in our development, we started to gather accelerometer sensor data from various people when they performed each of our gestures while holding the evaluation module in an upright position. We assumed that the sensor was attached in an upright position in the forearm area where it could monitor the gestures. By analyzing this sensor data, we started to implement the first version of recognition engine. We argued that if using only one sensor was sufficient for our purposes, then this would help to implement a light-weight recognition engine that would result in a fast and reliable wearable system. From this simple evaluation, we determined that we could implement the customize recognition engine that can distinguish among our 12 gesture commands.

5.1 Placement of an Accelerometer Sensor

Along with the development of the software recognition engine, we also continued our hardware design process. The most important hardware design issue we encountered was selecting the precise placement of the accelerometer sensor. We had already decided to locate it on the forearm, but the optimal position was important as it could affect the usability as well as the gesture recognition rate. For wearable design, the locations of hardware components on the body are often an important factor [16], which made us to design 3 prototypes for a experimental evaluation where the sensors were located differently as shown in Figure 3 (sensors are indicated with arrows in the figure). The locations were selected by investigating natural positions of hand and wrist area when we lift our forearm by bending the elbow until the forearm becomes perpendicular to the body as the posture seemed the most natural for making gesture command. The sensor was then placed on a flat surface resulting from the natural hand or arm posture so that the sensor can stay flat to generate robust output.

The possible location of a button which can be used to signify the start and end of gesture was also considered at this time.

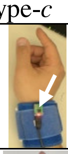
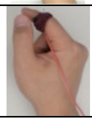
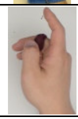
	Type-*a*	Type-*b*	Type-*c*
Sensor locations			
Button locations			

Fig. 3. Sensor and button locations (top view)

Although the type-c design, where the sensor was placed on the wrist, seemed the most hands-free and preferable for most wearable users, we initially speculated that the further the sensor was placed from the elbow and closer to the tip of the fingers, the greater the recognition rate would be. Note that the prototype-a and prototype-b in Figure 3 uses a glove for a stable placement of the sensor. However, wearing gloves is not ideal for everyday use and therefore it was outside of our target scenario. Instead, we wanted to see how the locations of the sensor affect our development by conducting an experiment that will be discussed in section 5.3.

5.2 Gesture Recognition Engine

First we classified each gesture command by the plane it traverses. Note that there are no gestures assigned that use only the y-axis because making gestures only traversing the y-axis did not seem natural but rather awkward. Other possible gesture commands can be added later if they seem suitable for the y-axis alone.

The gesture recognition engine classifies each of the users' movements according to the partitioning diagram shown in Figure 4. Each gesture data was preprocessed using normalizing and sub-sampling techniques and analyzed and characterized in terms of the maximum and minimum values of the acceleration along each axis and where they occur in time-vs.-acceleration plots as well as quantitative comparison of them in order to find parameters for the software recognition engine so that it can

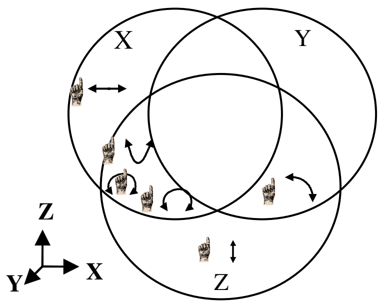


Fig. 4. Partitioning gesture commands in diagram

recognize each command. In addition, as the command set increased, more geometric characteristics were considered such as the starting/end value and vertex (local maxima/minima) locations of each input vector. This method of extracting characteristic information to distinguish gesture commands was used to determine parameters to drive a rule-based recognition engine.

5.3 Experiment Determining the Sensor Location

After we implemented the first version of recognition engine, we conducted an experiment to determine the optimal location of the sensor as discussed in section 5.1. The study had 11 participants. 2 were female, 9 male, all were right-handed except one person. The mean age was 34. The goal of the experiment was to examine the relationship between the performance of the gesture recognition engine and hardware design by determining how the accelerometer sensor location affected gesture recognition rate. Each participant was asked try on our 3 different prototypes and buttons, then make every gesture command three times. All were asked to fill out a questionnaire (categorized as ‘excellent’, ‘good’, ‘average’, ‘somewhat hard’, ‘poor’) that asks how well the prototype device worked. The results are shown in Table 2 (with responses scored from -2 to +2).

Table 2. Questionnaire result

	Sensor locations			Button locations		
Type	*a*	*b*	*c*	*a*	*b*	*c*
Score	5	4	9	13	1	3

Table 3. Gesture recognition rate according to different sensor locations

	Type-a	Type-b	Type-c
Total %	65.2	55.9	72.6

From Tables 2 and 3, we concluded that the sensor located on the wrist as shown in Figure 3 (type-c) gave the best recognition rate. Most of the testers seemed to share these sentiments as indicated by the questionnaire results illustrated in Table 2. One of the reasons why the type-c configuration showed the best result is that the accelerometer sensor is placed on the wrist so that the data has less variance than that derived from having the sensor on the top of the hand where it also monitors independent movements of the wrist. Removing this extra degree of freedom results in cleaner and more consistent data. This led us to the conclusion that monitoring wrist action (or forearm action) is the best way to monitor broad group of users with our hard-coded gesture recognition engine which is suitable to. The recognition rate of 72.6%, which was not yet considered acceptable, showed that the software recognition engine requires additional improvement with the sensor placed on the wrist and the users need a longer training period.

Finally we further developed our gesture band prototype hardware design as shown in Figure 5. In this iteration, it can be worn on the forearm in order to enable the activities of controlling and transferring multimedia files. The software recognition engine was also improved to tailor it to the scenario where the accelerometer is fixed on the wrist to achieve the maximum recognition rate. Note that the gesture band has mobility as it has its own battery and processor unit (worn on elbow in Figure 5, I.MX21 on 266MHz) running an embedded operating system and supports wireless communication (IrDA transceiver, Bluetooth and Wireless LAN) [17]. The usage of the IrDA transceiver is to trigger the data transfer between the two wearers, or between one wearer and other devices.

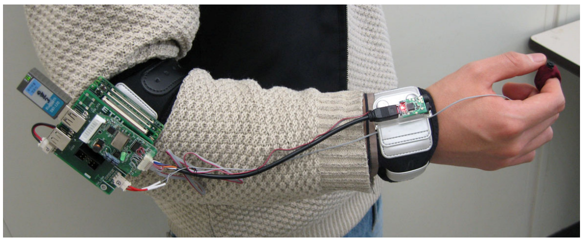


Fig. 5. Final prototype gesture band

For the future commercial production, our prototype device can be separated into two pieces depends on its usage so that it can have smaller form factor. We think the two pieces will be 1) a wristband type gesture recognition unit and 2) a portable gateway unit, and they are paired together.

6 Final Evaluation

As an evaluation stage of our development process, we needed to compare the system with an existing system that is used for similar purposes. However, to the best of our knowledge, there is no such wearable device that utilizes only one 3-axis accelerometer sensor to recognize a small group of gesture set. One part that could be compared to the existing technology was the gesture recognition software module which was one of the critical factors in this project. Since the HMM based gesture recognition technique is most commonly used and well-approved, we spent time porting an HMM based recognition engine onto our device. To do this we used the Hidden Markov Toolkit (HTK) that is available from the Cambridge University HTK home page [18].

With the gesture recognition band shown in Figure 5, we let one of our experimental participants to use the device in a regular basis (once every two weeks) and make each of our gesture commands. We observed the improvements on the recognition rate from this user after the 3 months. This is shown in Table 4. This individual user became well adapted to the wearable gesture band by achieving a recognition rate of 96.7%. The same experiment participant was asked to use the HMM based gesture recognition band as well. The resulting recognition rate of 99% was better than that of the customized engine however the recognition time (of 1.4 second) was not as quick as the customized engine (of 0.2 second). The actual number of lines in the code of the customized engine has 400 uncommented lines of code while the HMM based engine has 1170. For the compiled engine, the customized engine is 33Kbytes in size including required drivers such as USB driver and button driver, while the HMM based system is 550Kbytes including required libraries.

Generally speaking, our customized rule-based engine has weaker expendability in terms of the recognizable gesture set compare to that of learning-based engine. However, when considering that the embedded systems usually have limited CPU power and memory, the recognition rate and the response time of the customized engine using a single accelerometer sensor attached on the top of the wrist demonstrates that our recognition engine and device can be useful.

Table 4. Gesture recognition engine summary and performance

	Customized Engine(1)	HMM based Engine(2)	Ratio (1)/(2)
Recognition rate in %	96.7	99	0.977
Recognition time in sec	0.2	1.4	0.143
Number of lines of code	400	1170	0.342
Size of the code in byte	12K	41K	0.293
Size of compiled engine	33K	550K	0.060

7 Conclusions

We have presented a wearable system that can be worn on a forearm and that enables the practical application scenario of controlling and transferring various information or services.

Analyzing intuitive gestures suitable to this scenario, we defined 12 specific gesture commands. We also developed a software recognition engine that receives and recognizes the gesture commands. The method used to develop the gesture recognition algorithm was to classify gesture commands in terms of x, y, z axis and x-y, y-z, x-z planes, then design the engine such that it extracts commands by monitoring feature values of the preprocessed x, y, z data, while the x, y, z data is being cross-compared. Then we examined the relationship between the gesture recognition engines and the hardware construction design by discussing how we determined the optimal accelerometer sensor location.

After going through the evaluation process of the development considering the recognition rate compared to the existing HMM based gesture recognition engine, we conclude that the gesture recognition band with an accelerometer sensor attached to the wrist showed potential to achieve a relatively high recognition rate in real-time operation.

To summarize, we have developed a gesture recognition band that is suitable for a mobile environment with the considerations of wearability in such a way that the device could worn anytime, anywhere and supports hands-free operation. It provides a reasonable gesture recognition rate using the minimum possible sensors (requires only one 3-axis accelerometer sensor in this study). We are currently investigating how we could remove the buttons as well as to reduce the form factor to a wristwatch type wearable device.

References

1. H. Brashear, T. Starner, P. Lukowicz and H. Junker, “Using Multiple Sensors for Mobile Sign Language Recognition”, Proc. IEEE International Symposium on Wearable Computers, 2003, pp. 45-52.
2. C. Randell, H. Muller, “Context Awareness by Analyzing Accelerometer Data”, Proc. IEEE International Symposium on Wearable Computers, 2000, pp. 175-176.
3. H. K. Lee and J. H. Kim, “An HMM-based threshold model approach for gesture recognition”, Transactions on Pattern Analysis and Machine Intelligence, 1999, pp. 961-973.
4. J. Yamato, J. Ohya and K. Ishii, “Recognizing Human Actions in Time-Sequential Images Using Hidden Markov Models,” Proc. Computer Vision and Pattern Recognition, 1992, pp. 379-385.
5. J. Schlenzig, E. Hunter, and R. Jain, “Recursive Identification of Gesture Inputs Using Hidden Markov Models”, Proc. Conference on Applications of Computer Vision, 1994, pp. 187-194.
6. L. Campbell, D. Becker, A. Azarbayejani, A. Bobick and A. Pentland, “Invariant Features for 3-d Gesture Recognition”, Proc. International Conference on Face and Gesture Recognition, 1996, pp. 157-162.
7. T. Pylyanainen, “Accelerometer Based Gesture Recognition Using Continuous HMMs,” Proc. International Conference on Pattern Recognition and Image Analysis, 2005, pp. 639-646.
8. I. J. Jang and W. B. Park, “A Gesture-Based Control for Handheld Devices Using Accelerometer,” Proc. International Conference on Progress in Pattern Recognition, Image Analysis and Applications, 2004, pp. 259-266.
9. J. Rekimoto, “GestureWrist and GesturePad: Unobtrusive Wearable Interaction Devices”, Proc. IEEE International Symposium on Wearable Computers, 2001, pp. 21-27.
10. T. Baudel and M. beaudouin-Lafon, “Charade: Remote Control of Objects Using Free-hand Gestures”, Communications of the ACM, vol. 36, 1993, pp. 28-35.
11. T. Starner, J. Auxier, D. Ashbrook and M. Gandy, “The Gesture Pendant: A Self-Illuminating, wearable, Infrared Computer Vision System for Home Automation Control and Medical Monitoring”, Proc. International Symposium on Wearable Computers, 2000, pp. 87-94.
12. M. Fukumoto and Y. Tonomura, “Body Coupled FingerRing: Wireless Wearable Keyboard”, Proc. CHI, 1997, pp. 147-154.
13. J. K. Perng, B. Fisher, S. Hollar and K. S.J. Pister, “Acceleration Sensing Glove (ASG)”, Proc. International Symposium on Wearable Computers, 1999, pp. 178-180.
14. ___, “Kionix, Inc. USB Demo Board Kit User’s Manual”, User’s manual, Kionix Inc., 2006.
15. G. Kortuem, Z. Segall and M. Bauer, “Context-Aware, Adaptive Wearable Computers as Remote Interfaces to ‘Intelligent’ Environments”, Proc. IEEE International Symposium on Wearable Computers, 2000, pp. 58-65.
16. B. Thomas, K. Grimmer, D. Mackovec, J. Zucco and B. Gunther, “Determination of Placement of a Body-attached Mouse as a Pointing Device for Wearable Computers”, Proc. International Symposium on Wearable Computers, 1999, pp. 193-194.
17. H. J. Ahn, M. H. Cho, M. J. Jung, Y. H. Kim, J. M. Kim and C. H. Lee, “UbiFOS: A Small Real-Time Operating System for Embedded Systems,” ETRI Journal, vol.29, no.3, 2007 (submitted for publication)
18. HTK Hidden Markov Model Toolkit home page, http://htk.eng.cam.ac.uk/
19. N. Khotake, J. Rekimoto and Y. Anzai, “InfoStick: an interaction device for Inter-Appliance Computing,” Proc. International Symposium on Handheld and Ubiquitous Computing, 1999, pp. 246-258.