Coin Size Wireless Sensor Interface for Interaction with Remote Displays

Atia Ayman, Shin Takahashi, and Jiro Tanaka

Department of Computer Science, Graduate school of systems and information engineering, University of Tsukuba, Japan
{ayman,shin,jiro}@iplab.cs.tsukuba.ac.jp

Abstract. Human gestures are typical examples of non-verbal communication, and help people communicate smoothly [1]. However, using camera to recognizing gesture needs high processing power and suffer from delays in recognition [2]. Sometimes distance between large screen and user is a problem as for example in pen based interaction user must be attached to screen. So our main motivation is how we should design a user interface that use cookie wireless sensor [3] as an input device. In this paper we describe the interface setting, method of extracting motion and direction from 3D accelometer, using the tilting gesture. Then we proposed a method that allows users to define their own tilting positions and refer it to certain directions. Then we describe a menu selection interface that is based on pie menu for interaction with remote displays. An evaluation of the proposed interface in terms of accuracy, time and attached objects has been conducted.

Keywords: Wireless sensor, interaction with large screen display, Human computer interaction.

1 Introduction

The fast development of wireless sensor devices leads to new trends in wearable devices. Gesture interface is a promising interface for ubiquitous computing environments. Human gestures are typical examples of non-verbal communication, and help people communicate smoothly [1]. Interaction with computers can be done by traditional keyboards, mouse and remote controls designed mainly for stationary interaction. Mobile devices such as PDAs, mobile phones provide new possibilities for interaction with various applications, but introduce new problems of small displays and small input devices [4]. Another way to interact with remote screen displays is by using a camera for hand recognition or LED light tracking. The camera captures an image then transfers it to some application that will do the analysis for the image and extract cursor motion. The main disadvantage of this method is that it needs a large processing power and that might cause delays between the real gesture movement and the analyzed gesture movement. Sensors have been used previously for example in [6]. Blue wand was used to control devices like TV-set, mp3 player. In this paper, we propose a technique that allows users to interact with remote graphical

J. Jacko (Ed.): Human-Computer Interaction, Part II, HCII 2007, LNCS 4551, pp. 733–742, 2007.

user interface items through the use of a 3D accelerometer sensor and tilting gesture. To enhance the accuracy of detecting tilting gesture, we propose a method by which users can define their own tilting positions. We call this method "*customized directions*". We have conducted experiments to find the relation between different objects and recognition methods. In this paper we show how to use the recognition methods and evaluate interaction with graphical user interface items like pie menus.

The rest of this paper is divided as follows: System overview, extraction of motion with 3D accelerometer, Interface evaluation.

2 System Overview

In this paper we build a system that can receive wireless sensor data, recognize and translate it into commands. The system is composed of plasma screen, Nokia sensor, Bluetooth connector and a server. The architecture of the system is shown in Fig. 1. The sensor is attached to some object like pen, sphere and cellular phone. If the sensor is turned ON it will send data to the Bluetooth connected device. The Bluetooth connected device can then send this data to manager server and hence it will execute the appropriate commands according to the user's gesture. The user is able to interact with the remote display within the range of Bluetooth distance.

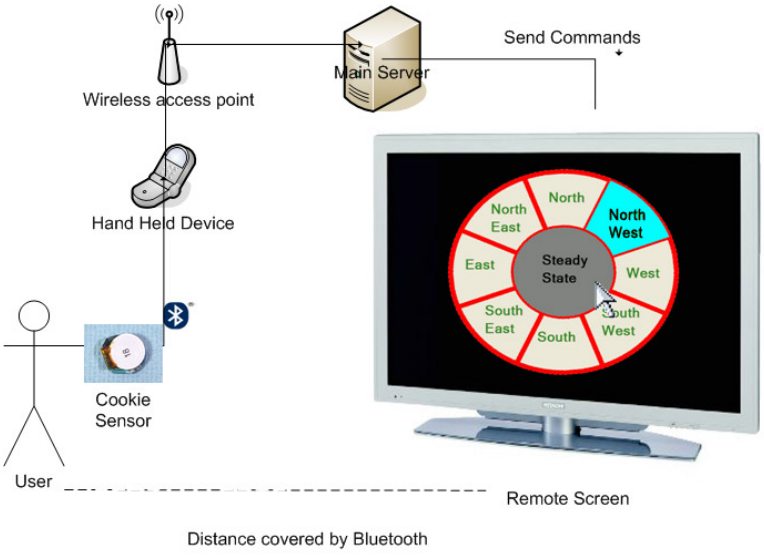


Fig. 1. Proposed system overview

The system has been used to interact with pie menus. Another kind of application that can be used with our system can be a presentation viewer that controls presentation navigation remotely. The user can flip slides forward/backward by moving hand up or down, right or left.

2.1 Nokia Sensors

Nokia cookie is an experimental device developed to test ubiquitous context aware applications. The chassis contains 6 sensors and sends data using Bluetooth. The device is about the size of 5 stacked quarter coins [3].

The sensor is composed of 2-axis linear accelerometer, Compass 3-axis sensor, Ambient light sensor, Galvanic skin response sensor, Heart rate sensor and Skin temperature sensor (see Fig.2 and Fig. 3). There are some other extension sensors that can be attached to give more flexibility to the sensor, for example, 3-axis linear accelerometer, ambient light sensor extension board (RGB color, UV), force vibration motor. The sensor has two communication interfaces, Bluetooth and UART wired connection [3].

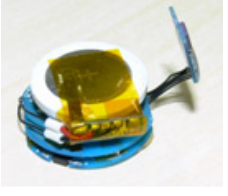


Fig. 2. Nokia Sensor with 3D accelerometer

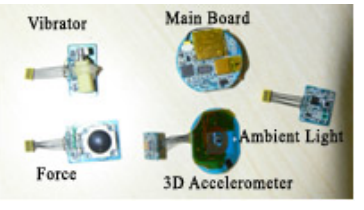


Fig. 3. Nokia Sensor Extension boards

3 Extraction of Motion with 3D Accelerometer

We propose three methods to convert 3D accelerometer data into an orientation of tilting gesture: predefined directions, customized directions and free direction movement. It is important to understand how cookie-3D-Accelometer calculates acceleration.

$$\mathbf{G}_i = 2.8 / 0.333 * (d_i / 1023 - 1 / 2). \textit{ where } \text{i } \textit{is } \text{x,y, } \textit{or } \text{z.}$$
$$\mathbf{G} = (Gx, Gy, Gz) \quad (1)$$

G is the acceleration applied to each axis and d is the voltage input to the cookie processor [3]. The sensor sends accelerometer captured data every 50 millisecond. G is calculated inside the cookie and then sent to the Bluetooth device.

3.1 Predefined Directions

By tilting the sensor in different directions the gravity value is added over the x, y, and z axis respectively. We have to define three threshold values Tx, Ty, and Tz. We pre-measure these threshold values for each of the basic 8 directions (North, North West, North east, east and west, south, south east, and south west). Also we measure the threshold values for the position of no tilting, we call this *steady state*. These positions must have no conflicts between each other. If the user tilts the sensor in some direction the system will match the value of G with the defined threshold

values. Table 1 shows each direction and the axis used for comparing threshold values. We start comparing with directions that use 3 axes: northwest, northeast, southwest, southeast then steady state. After that we compare directions that use 2 axes north, south, west and east. We call this method *predefined directions*. Fig. 4 shows the direction of gravity over the x axis while tilting the sensor in the east and west (left and right) directions.

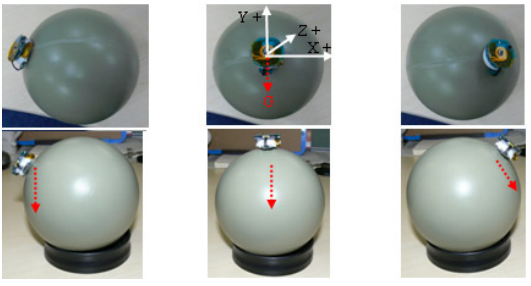


Fig. 4. Hand gesture tilting and gravity directions

Fig. 5 shows the threshold values of tilting the sensor over the x axis. The spots represent the threshold values for one tilt to east/west then returning back to steady state.

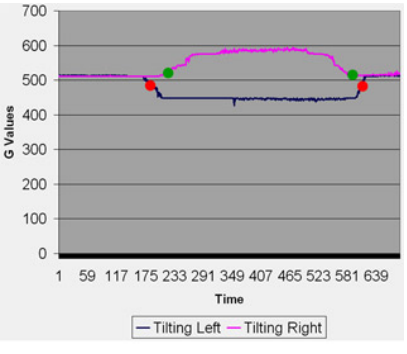


Fig. 5. Threshold values for left / right tilting

Table 1. Directions and axis used

Direction	X	Y	Z
North East	⊗	⊗	⊗
North West	⊗	⊗	⊗
South East	⊗	⊗	⊗
South West	⊗	⊗	⊗
Steady State	⊗	⊗	⊗
North		⊗	⊗
South		⊗	⊗
East	⊗		⊗
West	⊗		⊗

3.2 Customized Directions

The main idea of customized directions is to have a more flexible interface that doesn't depend on predefined threshold values. Users can assign their own gesture tilting to eight preferred directions. In the beginning the user is requested to pose his hand in any position. The system will record this position as steady state position. The user is then requested to tilt the sensor in the basic 8 directions. The user has a freedom in choosing his tilting positions.

The number 8 sphere in Fig. 6 shows the assignment of the steady state position, and the other spheres represent the positions of the eight directions. During the assigning of positions any point must satisfy two conditions to be marked as a customized user position:

First, the distance between the new point and *all* the old points must be greater than the virtual border value (Fig. 6 -dashed circle surrounding the points). The distance is calculated by this equation:

$$\text{distance} = \sqrt{difx^2 + dify^2 + difz^2} \tag{2}$$

where difx, dify and difz are the absolute difference between two successive $G_{x,y,z}$ over the x , y and z axis successively. This distance will represent a virtual border between the adjacent points to resolve any conflict that might arise while users assign positions.

Second, the user must settle hand in a certain position for at least two successive 50 milliseconds, i.e. the distance between two successive points less than or equal to 1.

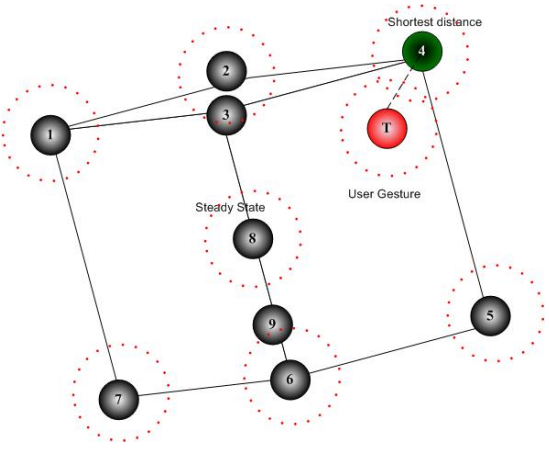


Fig. 6. Shortest distance between user gesture and customized thresholds

After defining a customized position for each direction, the user can tilt the sensor in any new position. This position will be compared to all the values in terms of distance. We get the minimum distance from all the points.

Fig. 6 shows the situation of point (T) being located near point 4. The shortest distance between the (T) point and all other points is calculated. A decision is made that orientation is in the direction of point 4.

In order to test how this method can be used, we attached the sensor to some type of cone shape. User holds the cone horizontally; we record this position as steady state (see Fig. 7). Then by holding the sensor vertically and rotating it 360 degrees. The user can assign each 45 degree tilt for one of the eight directions. Fig. 8 shows a sample of the customized gesture.

Fig. 7. horizontal position steady state

Fig. 8. vertical position up and right

3.3 Free Direction Movement of a Cursor

Using predefined directions, we extract motion while interacting with remote display screen. The algorithm depends on calculating a displacement value according to the captured acceleration $\mathbf{G}_i$ then translates it into remote screen display positions. The center point of remote display screen is the initial starting point for cursor movement. We calculate a displacement value that depends on the value of the tilt of the sensor and screen resolution factor. This displacement value was used to set the new position on the remote display screen. We call this *free direction movement*. This technique was tested to control a ready made flight simulator game.

4 Interface Evaluation

The main purpose of the evaluation is to test the usability of the interface while interacting with graphical user interface items like pie menu items. First experiment concerns measuring accuracy and time to select items. The second experiment compares different type of objects that can be attached to the sensor, so that we can find the effect of the attached object to the proposed methods. Then we evaluate the usability of customized directions method.

Subjects are 7 users between the age of 22 and 27 take part in the first experiment. We evaluate usage of Cone, Shop sticks, Sphere (ball) objects and projector mouse device. Each subject has to finish three sessions per each object. Each session is composed of 12 target selections. The first two targets are for training only. Targets are appearing on the pie menu randomly with marked color, and the user must tilt the sensor until hitting the target. Fig. 9 shows steady state, eight directions, current and next targets. The motion is composed of two gestures: selecting direction and, moving towards then returning back to steady state in order to execute another command. If the user hits unmarked pie items then this case will be counted as an error.

The primary results in Fig. 10 show that most of the users can hit targets using wireless mouse within 1~1.5 seconds. Wireless mouse takes more trials to select an item. This is because using the thumb to control mouse makes user gesture very fast without giving eye focus on next targets. This makes the cursor frequently pass by error pie menu items. This was also similar to free direction movement except it takes much time to hit targets because the motion orientation is hard to figure by users. The results show that predefined directions method can select menu items within 1.5~2 seconds. It records high frequency of selecting items from the first time. This means that predefined directions method can achieve more accurate menu selection items compared to other techniques. This is because the predefined directions method depends on moving to target directly so this means a reduced number of errors.

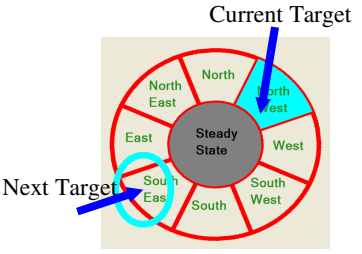


Fig. 9. Pie menu interface showing 9 directions and targets

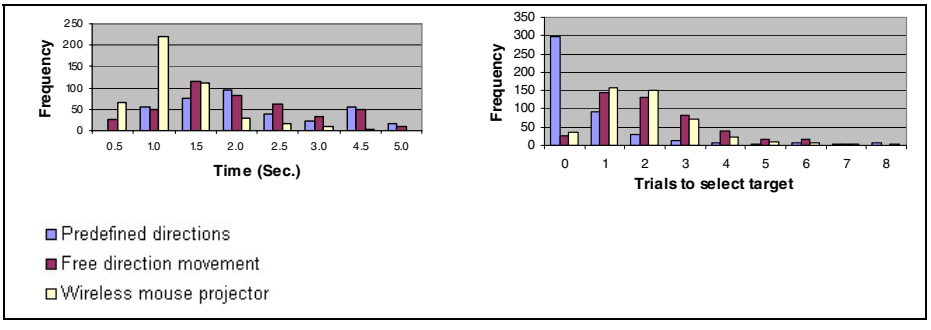
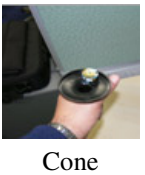


Fig. 10. Comparing proposed technique to wireless projector mouse in means of time/trials to select target

We then evaluated the usage of different type of objects that can be attached with the sensor. In this experiment the subjects are 3 users. We attached Nokia sensor to cone, shop sticks and ball objects, as can be seen in Fig. 11. For each object we have compared three methods: predefined directions, free direction movement and customized directions.

Fig. 12 (a) and (b) shows that using ball object can achieve good results for the eight directions and customized gestures methods and can also lead to a reduced number of errors. When we use the shop sticks as a controller, we observed a semi-normal distribution in the time to hit the targets, in contradiction to error rate, and this means that sensor takes time to return back to steady state. Then thin object might not be so appropriate to reach steady state fast (see Fig. 12 (c) and (d)). When we use the cone object (Fig. 12 (e) and (f)), we get good results at 1.5 seconds to hit the targets using predefined directions method and also a reduced number of errors.


Cone


Shop Sticks


Ball

Fig. 11. Attaching sensor to different objects

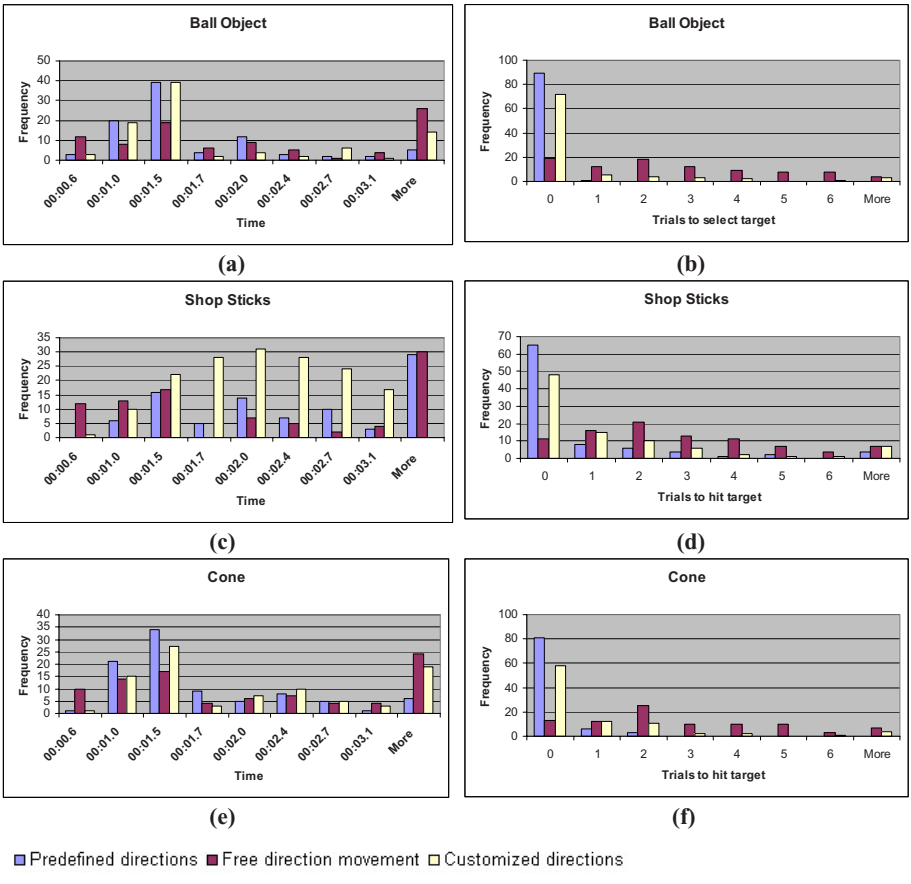


Fig. 12. Objects effect on predefined directions, free direction movement and customized directions

To know which object is suitable to which interface we compare each method and the corresponding objects. Fig. 13 shows that the sphere object is most suitable with predefined directions method and shows low error rates. Also we can observe that all the objects perform almost the same rates to hit targets. The sphere objects show low error rates. For the customized directions method it can be observed that it is very similar to predefined directions method, which means that it can achieve the same low error rate with sphere object.

Users have some comments while using the proposed methods: sometimes in free direction movement method the movements of mouse were very fast compared to real gesture. This case happened when user tilted the sensor suddenly and very fast so the displacement value calculated is very large, which makes movement be interpreted as jumps. Moving in south west and south east is difficult gesture. When an average person tilts the wrist of the hand towards the northwest and northeast direction the elbow assist the motion, while moving backwards (south) and tilting the elbow is fixed which makes the motion very difficult. When using the customized directions

interface, the users have some problems in understanding how the calibration of positions is performed. Whenever the calibration of positions is wrong, the movements are difficult to achieve. This happens when the users forget the points they mark, and this situation leads to false gesture tilting detection. Using the shop sticks with predefined directions method and moving to north and south direction is confusing. This situation can be resolved by using the customized directions interface.

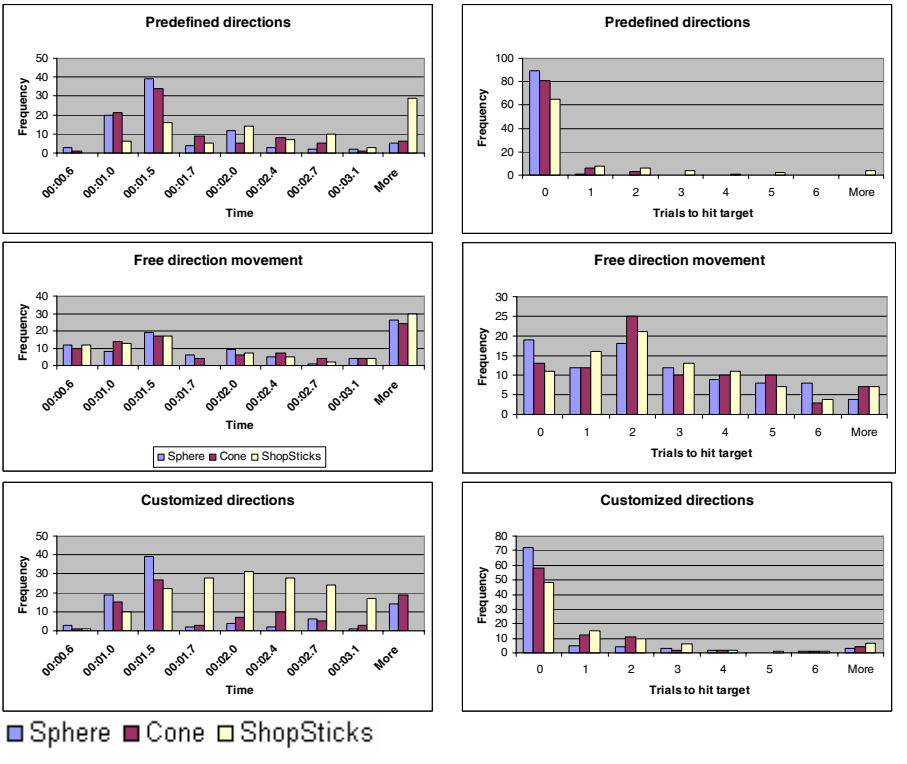


Fig. 13. Each method and appropriate object

5 Related Work

J.Kela et. al [4] use the accelerometer to extract commands from natural hand gesture movements to control external device. The proposed system in their work depends on using HMM to analyze a gesture which means that the system needs to be trained first by the users. At the beginning of recognizing gesture, an accuracy error appears which sometimes make users abandon the usage of such system. They do not propose continuous full hand gesture recognition. In our research we provide a method of continues recognition of tilting gesture only, without the need of providing training from the users except when we use customized directions method. H.Kimura, et. al [5] define three motions rotating, tilting, and knocking as gestures, then they use these

gestures for map viewer application. Users observed less motion accuracy and time delay between gesture and system action. This paper does not define personalized gestures.

6 Conclusion

In this paper we propose a gesture interface that can work using sensors and can support the interaction with remote displays. Then we define a customized gesture interface that can be adapted to user's gesture positions. We have proposed a system that makes use of these interfaces like presentation viewer, flight simulator game, pie menu interface. Then we evaluate the system in interaction with pie menus in terms of time and accuracy and suitable attached objects. It has been noticed that the proposed customized directions can be an initial good step for further enhancement of the method of detecting thresholds.

We need to conduct more experiments on evaluating different kinds of tilting positions with more users. The usage of more than one sensor must also be studied. Moreover the acquisition of full hand free gesture movement must be studied.

Acknowledgments. We would like to thank Nokia research center .Nokia Japan Co. Ltd. for using their cookie sensor.

References

1. Tsukada, K., Yasumura, M.: Ubi-Finger: Gesture Input Device for Mobile Use. In: Proceedings of APCHI 2002, vol. 1, pp. 388–400 (2002)
2. Badillo, et al.: Literature Survey on Interaction Techniques for Large Displays. TR-06-21, Computer Science, Virginia Tech (2006)
3. Hideki, O.: Nokia Cookie users manual, Nokia research center. Nokia Japan Co. Ltd (2005)
4. Kela, J., et,: Accelerometer-based gesture control for design environmen. ubiquitous computing 10, 285–299 (2006)
5. Kimura, H., et al.: CookieFalvors: Easy Building Blocks for wireless tangible input. In: CHI 2006, Montreal, Canada. work-in-progress (2006)
6. Fuhrmann, T., Klein, M., Odendahl, M.: The BlueWand as Interface for Ubiquitous and Wearable Computing Environments, EPMCC'03 (2003)