Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, Hong Kong, 19-22 August 2007

A VIRTUAL BUTTON INTERFACE USING FINGERTIP MOVEMENTS

KAZUHIRO MORIMOTO[1], CHIYOMI MIYAJIMA[1], KATSUNOBU ITOU[2], AND KAZUYA TAKEDA[1]

[1] Dept. of Media Science, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8603, Japan
[2] Faculty of Computer and Information Sciences, Hosei University, 3-7-2, Kajino-cho, Koganei, Tokyo 184-8584, Japan
E-MAIL: morimoto@sp.m.is.nagoya-u.ac.jp, {miyajima,takeda}@is.nagoya-u.ac.jp, itou@k.hosei.ac.jp

Abstract:

This paper presents a virtual push-button interface created by drawing a shape or line in the air with a fingertip. As an example, we develop a four-button interface for entering multi-digit numbers by making pushing gestures within an invisible 2x2 button matrix inside a square drawn by the user. Trajectories of the fingertip movements entering numbers are captured with a 3D position sensor mounted on the tip of the forefinger. Two methods are used for recognizing the gestures of drawing a square and pushing buttons. The first method recognizes the numbers according to the relative positions between the range of the drawn square and pushed points. The second method models normalized fingertip movements with hidden Markov models (HMMs). They are evaluated with four experimental participants, and 92.5% and 95.4% recognition rates are obtained, respectively.

Keywords:

User interfaces; Hidden Markov models; Gesture recognition; Position sensors

1. Introduction

There has been increasing attention focused on user-friendly human-machine interfaces with intuitive forms of interaction through speech or gestures. Nonverbal communication such as gestures often conveys a stronger message than verbal communication only. One example of gesture-based interfaces is *GestureWrist* [1], which was designed for interacting with wearable computers by recognizing hand gestures and forearm movements measured by using a wristband-type input device. Another is *Ubi-Finger* [2] developed for controlling electrical appliances such as televisions. Simple finger gestures assigned for actions including turning a television on or off, adjusting its volume, and selecting the channel were captured by acceleration and bending sensors mounted on a forefinger stall. Other vision-based hand or finger gesture interfaces have been also developed for many years [3].

In this paper we focus on finger-gesture interfaces and present a virtual push-button interface. The user of the interface first draws a shape or line somewhere in the air with the forefinger to create an input space for virtual buttons. The user then pushes invisible buttons as if they physically existed in the drawn shape or on the drawn line. If compact wireless 3D position sensors become ubiquitous in the future and virtual push-button interfaces are developed with such sensors, we will be able to operate machines more easily and conveniently in several situations such as while driving. Furthermore, it will become possible to interact with systems without using remote control handsets or finding the locations of manual operation buttons. This would lead to safer driving.

As a prototype of such a virtual button interface, we have developed a four-button interface for entering multi-digit numbers in which users push a 2x2 arrangement of buttons inside a square they have drawn. Here, the four buttons correspond to two-bit digits from 1 to 4. We conduct finger-gesture recognition experiments for the virtual push-button interface, in which participants draw a square in the air and then enter a multi-digit number consisting of the digits 1 to 4. Finger gestures of entering multi-digit numbers are recorded with a 3D position sensor mounted on the tip of the forefinger. Two methods are used to recognize the finger gestures for entering numbers. The first does not use any statistical model but simply recognizes the numbers according to the relative positions between the range of the drawn square and pushed points, which are estimated by determining where the sign of the velocity of fingertip movement changes. The second method models fingertip movements with hidden Markov models (HMMs) [4, 5] after normalizing the 3D coordinates in relation to the size and position of the squares. Their performances are evaluated using 240 randomly chosen three- to five-digit numbers recorded by four experimental participants.

1-4244-0973-X/07/$25.00 ©2007 IEEE

2. Virtual Button Interface

Push button interfaces are familiar to everyone and widely used for controlling home electrical appliances, e.g., turning lights on and off, selecting TV channels, and calling from a touch-tone phone. In this paper we develop a virtual push-button interface based on finger gestures. The user of the virtual button interface first draws a shape or line somewhere in the air with the forefinger to create an input space for virtual buttons. The user then pushes invisible buttons as if they were physically inside the drawn shape or on the drawn line.

By changing the combination of the shape of a geometric figure or line and button arrangement, we can assign various kinds of functions to the buttons, such as, assigning the on/off switching functions to two buttons inside a drawn rectangle, assuming that its right and left halves respectively correspond to on and off. Alternatively, we can adjust the audio volume to high or low by pointing at the upper-right and upper-left points on a drawn circle, respectively. If such interfaces are developed, we can flexibly operate systems in several situations such as while driving. Also, this method can potentially be integrated with speech recognition interfaces.

As an example of such an interface, we develop a four-button interface for entering multi-digit numbers by pushing 2x2 invisible buttons inside a square drawn by the user. The four buttons correspond to two-bit digits between 1 and 4. That is, the upper-left, upper-right, lower-left, and lower-right quarters are assigned to 1, 2, 3, and 4, respectively. A user draws a square in the air and then enters a multi-digit number consisting of digits 1 to 4. As shown in Figure 1, finger gestures of drawing a square and pushing virtual buttons are recorded by using a 3D position sensor mounted on the tip of the forefinger. Figure 2 presents an example of the fingertip trajectory for a 5-digit number "12433" captured by the sensor. Here, we assume that squares are drawn almost parallel with the y-z plane and buttons are pushed perpendicularly to the y-z plane i.e., parallel to the x-axis.

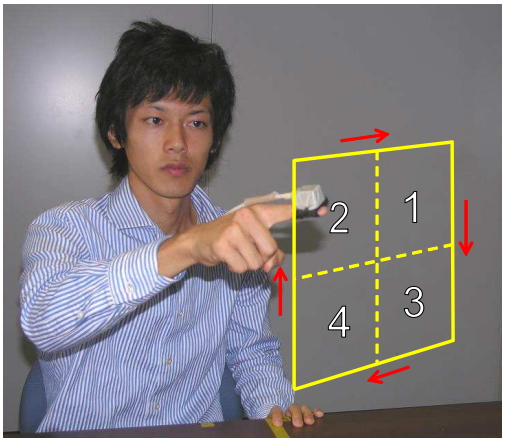

Figure 1. Recording of finger gestures of virtual button pushing using a 3D position sensor mounted on the fingertip.

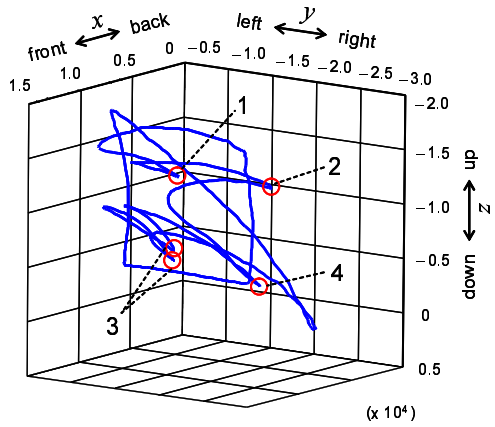

Figure 2. An example of a finger-gesture trajectory for five-digit number "12433" recorded with the 3D position sensor.

3. Finger gesture recognition

Two methods are used for recognizing the finger gestures of drawing a square and pushing buttons.

3.1 Finger gesture recognition using the relative positions (Method 1)

The first method does not apply any statistical model but simply recognizes the numbers according to their relative positions between the range of drawn square and pushed points. The procedure is explained in detail as follows.

(1) Estimate the times T_{BEG} and T_{END} when beginning and finishing the drawing of a square by finding the points where the sign of the velocity of fingertip movement changes, i.e., the first time derivatives of y-z coordinates captured with the 3D position sensor come to $\Delta y = \Delta z = 0$.

(2) Determine the range of the drawn square on the y-z plane, which is regarded as the rectangle surrounded by two pairs of parallel lines, with the minimum and maximum y and z coordinates of the trajectory recorded during between T_{BEG} and T_{END}.

(3) Determine the time sequence t_n, $n = 1, 2, \ldots, N$ when buttons are pushed for entering an N-digit number by finding the points where the first time derivative

of x position data comes to $\Delta x = 0$. (N is not given.)

(4) Determine the sequence of digits according to the relative positions of the y-z coordinates of the pushed points (y_n, z_n), $n = 1, 2, \ldots, N$, in the range of drawn square.

3.2 Finger gesture recognition based on HMMs (Method 2)

The second method models normalized finger movements with HMMs. Three-dimensional position data captured for each number are normalized to have the same position and size of a square before being modeled with HMMs because the positions and sizes of the drawn squares can not be the same. We simply normalize the y- and z-axes data with their minimum and maximum values obtained while drawing a square and the x-axis data with minimum and maximum values of x-axis data obtained while pushing buttons. The following seven sub-sequences of finger movement are modeled with left-to-right HMMs.

(i) before drawing a square,

(ii) while drawing a square,

(iii)–(vi) pushing a button for digit 1, 2, 3, or 4,

(vii) after pushing buttons.

4. Experiment

To evaluate the performances of the above two methods, we performed finger gesture recognition experiments.

4.1 Experimental conditions

Finger gestures of 240 randomly chosen three- to five-digit numbers produced by four subjects were recorded with a 3D position sensor mounted on the tip of the forefinger. A square was drawn with a continuous line beginning and ending at the top left corner every time before entering a multi-digit number. The height and width of the squares were about 30 cm. The 3D trajectories of the fingertip movements were recorded at a sampling rate of around 100 Hz using the Flock of Birds magnetic tracker. Cubic spline interpolation was then applied to equalize the sampling periods to 10 ms.

An N-digit number was counted as correctly recognized only if all the N digits composing it were correctly recognized. For Method 2, the four-fold cross-validation procedure was used, i.e., data of three of four subjects were used for training HMMs, and those of the remaining subject were used for testing. Recognition results were averaged over the four tests.

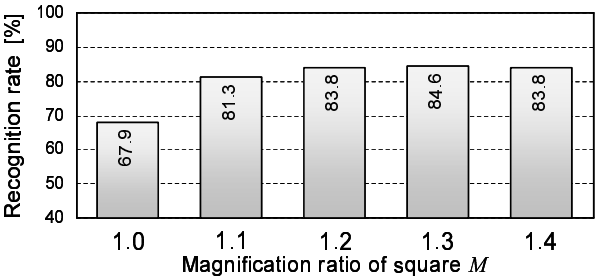

Figure 3. Recognition rates for Method 1.

4.2 Experimental results

4.2.1 Method 1

Figure 3 shows the recognition rates for Method 1. In the figure, M on the horizontal axis corresponds to the magnification ratio of the original square to extend the input space of buttons with extra margins around the square. For example, "$M = 1.1$" means that pushed points (y_n, z_n) inside the drawn square or an extra 10% of margin space around the square are accepted as input digits. By extending the input area with $M = 1.3$, the recognition rate was improved from 67.9% to 84.6%.

Figure 4 shows the deletion error rate for each digit. The deletion errors were significantly reduced after extending the input area, especially for the two lower buttons corresponding to digits "3" and "4." These results suggest that there are spacial differences between the two spaces of the drawn squares and the range of finger motion due to the constraint imposed by the kinematics of the user's arm and finger.

The difference between the drawn square and the range of button pushing is illustrated in Figure 5. We can see that the range of button pushing is not parallel with the drawn square and tilted to the back. To correct this spacial difference, we used the intersection points of the fingertip trajectory and the plane of the drawn square instead of the pushed points. Figure 6 shows the results when using the intersection points. The recognition performance was significantly improved by the correction and a rate of 92.5% was obtained for $M = 1.2$.

4.2.2 Method 2

Then finger gestures were modeled with HMMs. HMM parameters were trained with the Baum-Welch algorithm following initialization by the segmental k-means algorithm [5].

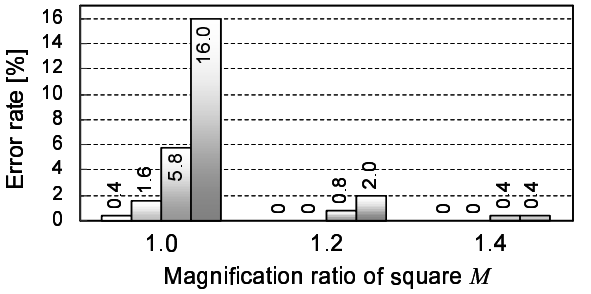

Figure 4. Deletion error rates for each digit.

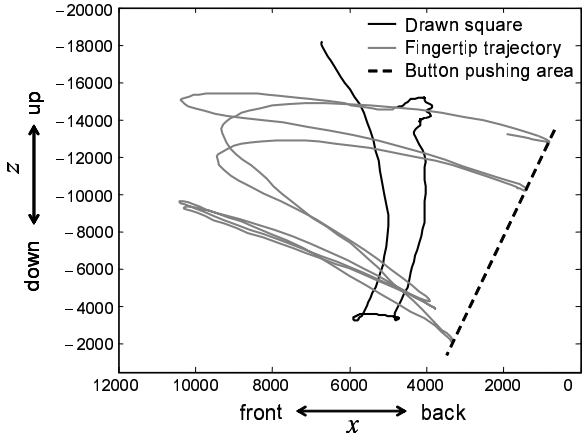

Figure 5. Difference between the two spaces of the drawn square and the range of finger motion.

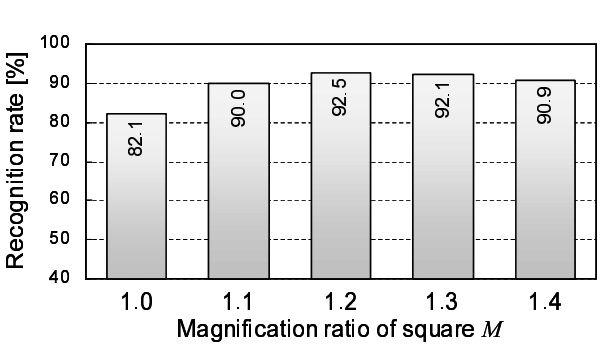

Figure 6. Recognition rates for Method 1, where the intersection points of the fingertip trajectory and the plane of the drawn square were used instead of pushed points.

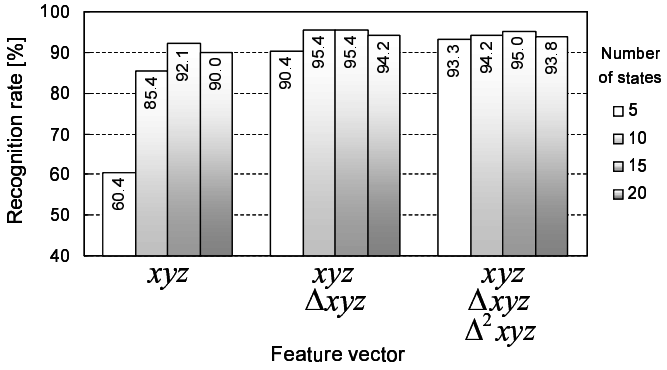

Figure 7. Recognition rates for Method 2.

Figure 7 displays the recognition results for HMMs. The dynamic features (first time derivatives) of the 3D position data x, y, and z, denoted with Δx, Δy, and Δz, were calculated as linear regression coefficients with a window length of 40 ms. We also obtained Δ^2 coefficients from a linear regression of the Δ coefficients. A 92.1% recognition rate was obtained when using static features alone (x, y, and z), which is comparable to the result of Method 1. The additional use of dynamic features significantly improved the recognition performance and a recognition rate of 95.4% was achieved by using six-dimensional feature vectors including x, y, z, Δx, Δy, and Δz. This corresponds to a 38% improvement over the result of Method 1.

5. Conclusion and future work

In this paper, we presented a virtual push-button interface created by drawing a square in the air with a fingertip. We developed a four-button interface for entering multi-digit numbers that works by users pushing imaginary points within an invisible 2x2 button matrix inside a square. Trajectories of the fingertip movement were captured with a 3D position sensor. Two methods were used for recognizing the finger gestures involved with drawing a square and pushing buttons. The first method used relative positions between the range of the drawn square and pushed points, and a 92.5% recognition rate was obtained by using the intersection points of the fingertip trajectory and the plane of the drawn square. The second method modeled normalized fingertip movement with HMMs and achieved a 95.4% recognition rate by using dynamic features of finger trajectories.

Future work includes normalizing the position data with linear or nonlinear mapping of the space of button operation into the space of drawn figures. We also plan to develop other types of virtual button interfaces as described in Sect. 2.

References

[1] J. Rekimoto, "GestureWrist and GesturePad: Unobtrusive wearable interaction devices," Proc. of the 5th International Symposium on Wearable Computers, Oct. 2001.

[2] K. Tsukada and M. Yasumura, "Ubi-Finger: Gesture input device for mobile use," Proc. of 5th Asia Pacific Conference on Computer Human Interaction, vol.1, pp.388–400, Nov. 2002.

[3] V.I. Pavlovic, R. Sharma, and T.S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.19, no.7, pp.677–695, July 1997.

[4] X.D. Huang, Y. Ariki, and M.A. Jack, Hidden Markov models for speech recognition, University Press, Edinburgh, 1990.

[5] S. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland, The HTK Book, Microsoft.