Evaluation of a Pointing Interface for a Large Screen Based on Regression Model with Image Features

Koichi Shintani  
Graduate School of Information Science and Technology, Osaka University, Japan  
Email:shintani@lab.ime.cmc.osaka-u.ac.jp

Tomohiro Mashita  
Cybermedia Center, Osaka University, Japan

Kiyoshi Kiyokawa  
Cybermedia Center, Osaka University, Japan

Haruo Takemura  
Cybermedia Center, Osaka University, Japan

Abstract—Gesture input interfaces for mobile devices with a touch screen have become widespread. Although gesture interfaces in common use are limited to the small screens of these mobile devices, pointing interfaces for large screens using handheld devices or attachments are common. However, these devices increase the user’s cognitive load because they are unable to cancel the effect of spatial cognition. This paper presents a pointing interface for a large screen. This interface is based on learning a user’s pointing motion and providing for intuitive pointing by canceling the effect of spatial cognition. Multiple linear regression analysis is used for learning and estimation. This method was implemented in a prototype system with a large projected screen, and experiments with the prototype system confirmed the validity of the proposed method.

I. INTRODUCTION

Gesture input interfaces for mobile devices with a touch screen have become widespread. However, commonly used gesture interfaces are limited to the small screens used in these mobile devices. Therefore, it is a challenging problem to develop a gesture interface intended for mid- to large-size screens. Hands-free interfaces such as “hand gesture control” (TOSHIBA) [1] and “Project Natal” (Microsoft) [2] are recent attempts to commercialize a gesture interface. One of the difficulties in making a gesture interface for mid- to large-size screens is ensuring the accuracy of the pointing input using gestures. Direct touch sensors cannot be used for mid- to large-size screens. A pointing rod, laser, and so on, have been used for pointing at far objects. For example, screen interaction using a laser has been reported [3]. However, such handheld devices are not suitable for gesture interfaces because these devices can increase the cognitive load. So, it is important for gesture interfaces to be able to estimate an indicated position from a pointing gesture without using handheld devices.

In the field of cognitive science, it is well known that a pointing gesture is related to the sense of sight and spatial cognition [4], [5]. Brain [6] classified the characteristics of spatial cognition into reaching distance and walking distance. By evaluating pointing gestures, Yoshida et al. [5] verified that spatial cognition within walking distance causes an error between real space and cognitive space. Errors due to spatial cognition is not negligible for interactive systems with a mid- to large-size screen within walking distance.

In this paper, we propose a pointing interface that prevents the effect of spatial cognition within walking distance. The proposed method estimates the position on a large screen indicated by a pointing gesture without using any handheld or body-worn devices. This pointing interface can be used to achieve intuitive human computer interaction.

II. RELATED WORK

In this section, we refer to existing pointing interfaces and describe an interface suitable for a large screen within walking distance.

A. Pointing interfaces

There are many pointing interfaces, such as a mouse or a dataglove. Figure 1 shows the classifications based on cognitive load and screen size. The cognitive load on the vertical axis indicates the difficulty in understanding how to use the system. The interfaces on the upper side have a low cognitive load and are intuitive. Touchscreens are the most intuitive devices for small screen sizes. On the horizontal axis is the screen size. The WiiRemote [7] or other interfaces are for mid- to large-size screens. The goal of our research is to achieve a pointing interface that would be located on the upper right of Fig. 1.

1) Pointing interfaces suitable for a large screen: This section introduces pointing interfaces suitable for a large screen. Vogel et al. [8] used a motion capture system. The cursor’s coordinates on a screen are aligned with the user’s coordinate system. This pointing system achieves a pointing interface based on the relative motion of ray-casting or the hand’s 3D position, which is measured by optical markers attached to a glove. WiiRemote uses an infrared camera and infrared lights from LEDs close to the screen. This system calculates the cursor’s coordinates from the LED locations in the image of the camera in the handheld device.
These interfaces achieve a pointing system by capturing the motion of a human pointing at a large screen. However, these systems are based on a direction from a pointing fingertip or device, and they do not consider the effect of spatial cognition.

2) Non-contact interfaces with cameras: Non-contact interfaces with cameras recognize the user’s gestures from captured images. “Hand-Bin” [9] uses a camera near the screen to provide a pointing system based on the relative hand motion. Another product is EnhancedDesk [10], which uses fingertip detection or gesture recognition.

Non-contact interfaces have the advantage that users do not need to know a lot about the system and devices. Therefore, non-contact interfaces are suitable to achieve an interface that is intuitive and has a low cognitive load if it is possible to recognize a user’s motions or gestures accurately.

B. Requirement for the large-screen pointing interface

In this section, we outline the requirements for our pointing interface. Our system does not use handheld or body-worn devices because these kinds of devices can influence a user’s pointing motion and increase the cognitive load. Interfaces with cameras can achieve a pointing interface appropriate for the upper right of Fig. 1 because it is possible to achieve a non-contact interface.

Therefore, a pointing interface with low cognitive load for a large screen must cancel the effect of spatial cognition, and be non-contact, and not use handheld devices. Table I shows the characteristics of some camera-based interfaces. All the related studies in Table I find the indicated position by extending from a pointing finger or the position of a region of interest in a captured image. Interactive Hand Pointer (IHP) [11] determines the pointing direction from a pointing finger. EnhancedDesk [10] does not cancel the effect of spatial cognition because this system is used on a desk, in otherwords, EnhancedDesk is a system within reaching distance and does not have to cancel the effect of spatial cognition.

III. PROPOSED METHOD

A. Approach

The characteristics of spatial cognition in a pointing motion are assumed to affect the relation between a pointing gesture and an indicated object. However, spatial cognition is affected by the environment. Moreover, the effect of spatial cognition is not fixed, because it can vary depending on the individual. Therefore, it is difficult to define a model that directly expresses the characteristics of spatial cognition.

But, is not necessary to express the characteristics of spatial cognition directly. A model that expresses the relation between a user’s pointing motion and a corresponding indicated position is sufficient for a pointing interface for a large screen because this can cancel the effect of spatial cognition. We assume that the effect of spatial cognition appears in the user’s arm motion when pointing. Our method uses a multiple linear regression model to express the relation between the indicated positions and image features of the pointing arm.

B. Flow of proposed method

Figure 2 shows the flow of the proposed method. This method has two phases: learning and estimation. In the learning phase, we need a training set of supervised data. We use pairs of an indicated position on the screen and the corresponding image features as supervised data. Binarized images of pointing gestures are obtained through background subtraction and noise reduction. Noise reduction is conducted by erosion and dilation.

1) Estimating regression coefficients: To estimate partial regression coefficients $B$, we conduct a multiple linear regression analysis. The variables for the multiple linear regression
TABLE I
CHARACTERISTICS OF PREVIOUS CAMERA-BASED METHODS.

<table>
<thead>
<tr>
<th>Method</th>
<th>Canceling spatial cognition</th>
<th>Non-contact and non-handheld devices</th>
<th>Large screen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vogel [8]</td>
<td>poor</td>
<td>moderate</td>
<td>good</td>
</tr>
<tr>
<td>Hand-Bin [9]</td>
<td>poor</td>
<td>good</td>
<td>good</td>
</tr>
<tr>
<td>EnhancedDesk [10]</td>
<td>good</td>
<td>good</td>
<td>poor</td>
</tr>
</tbody>
</table>

analysis are as follows:

\[
A = \begin{bmatrix} a_1 & \cdots & a_N \end{bmatrix}^T, \quad (1)
\]

\[
B = \begin{bmatrix} b_0 \ b_1 \ \cdots \ b_M \end{bmatrix}, \quad (2)
\]

\[
C = \begin{bmatrix} c_0^T \\ \vdots \\ c_N^T \end{bmatrix}, \quad (3)
\]

where \(a_i\) is an indicated position, \(A\) is a matrix of dependent variables, \(N\) is the number of samples, \(C\) is the independent variable matrix, and \(c_i\) is an \(M\) dimensional feature vector.

We compute \(B\) that minimizes \(F\) in the following equation,

\[
F = (A - \hat{A})^2 = (A - CB)^2, \quad (4)
\]

to calculate the partial regression coefficients, where \(\hat{A}\) is the estimated value. \(B\) is obtained using the pseudo inverse as follows,

\[
B = [C^TC]^{-1}C^TA. \quad (5)
\]

2) Estimating indicated positions: An indicated position \(\hat{a}\) is estimated from its corresponding extracted image features\(c\). \(\hat{a}\) is obtained as

\[
\hat{a} = b_0 + [c][b_1 \ \cdots \ b_M]^T. \quad (6)
\]

IV. PROTOTYPE SYSTEM

For our pointing interface, we implemented a prototype system. The prototype system consists of a camera and projector.

Fig. 3 presents a system overview and Fig. 4 shows the system composition. The distance between the screen and the user is approximately 2500 mm. The screen size is shown in both Fig. 3 and Fig. 4. 

The camera used in the system is Flea2 FL2-08S2C (Point Grey Research Inc.), Table II shows the specifications of the camera. The projector used in the system is MP522ST (BenQ), and Table III shows the specifications of the projector. For a large screen, we use a short-focus projector Table IV shows the computer specifications.

TABLE II
SPECIFICATIONS OF FL2-08S2C.

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image sensor model</td>
<td>Sony ICX204 1/3</td>
</tr>
<tr>
<td>Maximum resolution</td>
<td>1032×776</td>
</tr>
<tr>
<td>Maximum frame rate</td>
<td>30FPS</td>
</tr>
<tr>
<td>Digital interface</td>
<td>Bilingual 9pin IEEE-13946</td>
</tr>
<tr>
<td>Dimensions (L×W×H)</td>
<td>29 mm×29 mm×30 mm</td>
</tr>
<tr>
<td>Mass</td>
<td>38 g (without optics)</td>
</tr>
</tbody>
</table>

TABLE III
SPECIFICATIONS OF MP512.

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Native resolution</td>
<td>SVGA (800×600)</td>
</tr>
<tr>
<td>Projection system</td>
<td>DLP</td>
</tr>
<tr>
<td>Brightness</td>
<td>2200 Ansi lumens</td>
</tr>
<tr>
<td>Contrast ratio</td>
<td>2500 : 1</td>
</tr>
<tr>
<td>Resolution support</td>
<td>VGA(640×480)─SXGA(1280×1024)</td>
</tr>
<tr>
<td>Image size</td>
<td>25” to 30”</td>
</tr>
<tr>
<td>Throw ratio</td>
<td>1.97─2.16</td>
</tr>
<tr>
<td>Lens</td>
<td>F=2.6, f=22 mm</td>
</tr>
<tr>
<td>Aspect ratio</td>
<td>4:3, 16:9</td>
</tr>
</tbody>
</table>

TABLE IV
COMPUTER SPECIFICATIONS.

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS</td>
<td>Linux Ubuntu 8.04</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel Core2Duo 3.0GHz</td>
</tr>
<tr>
<td>Memory</td>
<td>DDR2 1.0GB×2</td>
</tr>
</tbody>
</table>
following steps. The covariance matrix of a binarized image $R$ is defined by the following equation

$$
R = \begin{bmatrix}
S_{xx} & S_{xy} \\
S_{xy} & S_{yy}
\end{bmatrix}.
$$

(7)

The eigenvalues and eigenvectors of $R$ are obtained through singular value decomposition. The eigenvectors $v_0$ and $v_1$ correspond to $\lambda_0$ and $\lambda_1$, respectively, where $\lambda_0 \geq \lambda_1$. These eigenvectors indicate the principal axes. We calculate the principal axis direction $\theta = \arctan(v_{0y}/v_{0x})$, where $v_{0x}$ is the $x$ element of $v_0$ and $v_{0y}$ is the $y$ element of $v_0$.

The eigenimage is calculated by projecting onto the eigenspace through principal component analysis of all the learning datasets. We determined that the dimension of the eigenspace is 20.

V. EXPERIMENTS AND RESULTS

To evaluate the accuracy of the estimation of indicated positions, we conducted experiments with six test subjects who were graduate and undergraduate students. Supervised data was prepared for each test subject before the experiments. To collect supervised data and evaluate our method, our system projected fixed target images. Figure 5 shows examples of the target images. The lines in Fig. 5 divide the screen into $10 \times 10$ blocks. The circles are the pointing targets, which are located on the block intersections during training. The pointing targets for evaluation are located on the midpoints of the trained targets, as shown in Fig. 5 (b). A piece of supervised data is a binarized image of a pointing gesture with the corresponding target’s coordinates. Figure 6 shows examples of the binarized images. The order of pointing is from the top left to the bottom right and the subjects did not rest in between pointing motions.

A. Evaluation of estimation accuracy

To evaluate accuracy, we used two types of datasets. One is a dataset with target positions that were the same as the training data. We call this dataset “Test Dataset.” Figure 5 (a) shows an example of a target image for this dataset. The other is a dataset with target positions at the midpoints of the training data. We call this dataset “Midpoint Dataset.” Figure 5 (b) shows an example of a target image for this dataset. We evaluated the accuracy using these datasets.

Figures 7, 8 and 9 show the results of the estimation of indicated points of the training dataset, Test Dataset, and Midpoint Dataset respectively. The circles in these figures indicate the correct coordinates, and the lines connect the correct to the estimated coordinates. The blue lines show the points where the estimation error along the $x$ axis is less than 102 pixels and the estimation error along the $y$ axis is less than 76 pixels. These thresholds were calculated from the block size of Fig. 5. The red lines show the points where the error along the $x$ and/or the $y$ axis is larger than the corresponding threshold.

Table V shows the mean error of all test subjects for the Test Dataset and Midpoint Dataset. Table VI shows the unbiased variance of all test subjects for the Test Dataset and Midpoint Dataset.

<table>
<thead>
<tr>
<th>TABLE V</th>
<th>MEAN ERROR OF ALL TEST SUBJECTS.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Moment features</td>
</tr>
<tr>
<td>Test Dataset</td>
<td>71.8 pixels</td>
</tr>
<tr>
<td>Midpoint Dataset</td>
<td>79.3 pixels</td>
</tr>
</tbody>
</table>
Fig. 7. Estimation of the training dataset.

Fig. 8. Estimation of the Test Dataset.

Fig. 9. Estimation of the Midpoint Dataset.
There are large errors on the left side of Fig. 8 (a), as indicated by the red lines. In addition, errors down a column point in the same direction. We believe that these errors are due to the order of pointing. Therefore, the motion of pointing at a target at the top of a row is propagated to subsequent pointing motions. In the result of one test subject, the estimated positions of the Midpoint Dataset are shifted downwards. It is confirmed that the proposed method with the moment feature is better than that with the eigenimage, based on the mean error and unbiased variance of all test subjects.

We implemented a pointing interface using our pointing system and observed the behavior of the pointing cursor. In the case of the eigenimage, it is difficult to point to the edge of the screen due to the inaccuracy of estimation, as shown in Fig. 9 (b). In the case of the moment feature, inaccuracy at the edge of the screen is reduced. Comparing the two types of features, the moment feature can reduce errors along outer area. The mouse cursor in the system using eigenimages is unstable and wavy. On the other hand, cursor is stable in the system using moment features. The system with moment features can estimate pointing at 20 fps with a camera resolution of $640 \times 480$.

### B. Increasing the amount of supervised data

We evaluated the relation between the size of the training dataset and accuracy. We used six datasets for learning. Figure 10 shows the result of the estimation. In this experiment, we used the moment feature for estimation and learning. It is obvious that the estimation errors in Fig. 10 are lower than those in Fig. 8 (a). We presume that the influence of small movements occurring in the pointing motions is lessened by increasing the size of the training set.

### VI. Conclusion

We propose a method to estimate the positions on a large screen indicated by a user’s pointing gestures. The proposed method is based on a multiple linear regression model of the image features. Using a prototype system consisting of a projector and camera, we evaluated the estimation accuracy and compared the moment features and eigenimages. The results from the evaluation confirm that the moment features are stable and suitable for our system. Future work includes the development of a method to reduce errors by using small movements in the pointing motion.

### ACKNOWLEDGMENT

This research was supported in part by “Global COE (Centers of Excellence) Program” of the Ministry of Education, Culture, Sports, Science and Technology, Japan.

### TABLE VI

<table>
<thead>
<tr>
<th></th>
<th>Moment features</th>
<th>Eigenimage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test data</td>
<td>1832.9</td>
<td>2451.2</td>
</tr>
<tr>
<td>Intermediate data</td>
<td>1438.3</td>
<td>2101.2</td>
</tr>
</tbody>
</table>

### REFERENCES


