DOI:10.5392/IJoC.2010.6.4.049

Infrared Sensitive Camera Based Finger-Friendly Interactive Display System

Deepak Ghimire
Computer Engineering Department
Chonbuk National University, Jenoju, Republic of Korea

Jooncheol Kim*
Department of Electronics Engineering
Seonam University, Namwon, Republic of Korea

Kwangjae Lee
Department of Electronics Engineering
Seonam University, Namwon, Republic of Korea

Joonwhoan Lee
Computer Engineering Department
Chonbuk National University, Jeonju, Republic of Korea

ABSTRACT

In this paper we present a system that enables the user to interact with large display system even without touching the screen. With two infrared sensitive cameras mounted on the bottom left and bottom right of the display system pointing upwards, the user fingertip position on the selected region of interest of each camera view is found using vertical intensity profile of the background subtracted image. The position of the finger in two images of left and right camera is mapped to the display screen coordinate by using pre-determined matrices, which are calculated by interpolating samples of user finger position on the images taken by pointing finger over some known coordinate position of the display system. The screen is then manipulated according to the calculated position and depth of the fingertip with respect to the display system. Experimental results demonstrate an efficient, robust and stable human computer interaction.

Keywords*: Interactive System, IR Sensitive Camera, Background Subtraction, Bilinear Interpolation, Moving Average.*

1. INTRODUCTION

Different electronic devices such as mouse and keyboards are commonly used as 2-D human computer interface. But for large display system, such as advertisement displays walls, graphics display walls used for entertainment etc., the user needs to interact with the screen by standing just in front of the display, which may become difficult with mouse and keyboards.

Touch position sensing in touch-pads is a common laptop input device measuring the capacitance created between a user's finger and an array of charged plates under the touch surface. The Mitsubishi Diamond Touch display [1] sends a unique electrical signal through the sets of each of its users, and an antenna array under the touch surface localizes and identifies unique touches.

Computer vision techniques for human-computer interface have provided a good alternative to traditional mechanical input devices. While camera provide a high rate of input information, algorithms that efficiently reduce this data to desire parameters are necessary to minimize the interface latency. Dr. G.D. Morrison [2] developed a man-machine input device for large-format interactive displays. His system determines where a user has touched the display, and then treats that information as a

* *Corresponding author. E-mail : kjc1029@daum.net*
Manuscript received Jul. 28, 2010 ; accepted Dec. 20, 2010

mouse click, there by controlling the computer. He is using at least 4 cameras to find the user finger position over the screen. He is also using IR light to easily determine the pointer location through simple thresholding. Vladmir et al. [3] presented a human computer interaction by finding visual interpretation of hand gesture. Tony K. S. Cheng [4] developed interaction system based on stereo computer vision with multiple cameras. The 3-D information of human parts such as positions of fingertip is extracted from images taken simultaneously from cameras and uses this information as input to the system. A vision based non-contact interactive advertisement with a display wall [5] enables the user to interact with advertisement on a large display wall without any physical contact. Itai Katz et al. [6] developed a multi-touch surface using multiple cameras. This system determines the fingertip position with overhead camera and a side-mounted camera and calculates the 3-D coordinates of the fingertips.

In vision based finger friendly interactive system accurate finger position detection in images is very important, because this directly affects the system performance. Ye Zhou et al. [7] proposed an algorithm for finger detection in a camera based finger-friendly interactive board system. The region where a finger intersects with the stripe, which is the projection of the edge of the board on the image, is first detected and segmented from the background. A region growing algorithm is then applied to the region to segment the whole finger. Abhimanyu Sanghi et al. [8] developed fingertip detection and tracking system using the sampled hand contour for the purpose of implementing a virtual mouse, a signature input device and an application selector. Duan-Duan Yang et al. [9] proposed fingertip detection method in 2-D plane using grid sampling, hand contour detection and circle feature matching. Ravikiran J et al. [10] presented finger detection based on the concept of boundary tracing and fingertip detection for American Sign Language recognition.

For mapping detected finger position in images to screen coordinate, most of the systems use the camera calibration and stereo matching method. In this paper, a finger-friendly human computer interface using two infrared light sensitive cameras is implemented. The system interacts with users by tracking 2-D coordinate position of the screen with respect to the fingertip position as well as depth between display screen and user fingertip. In our system we first find the user finger position in left and right camera view images by pointing finger over some known coordinate position of the screen. Next we interpolate those samples of finger position all over the screen coordinate using bilinear interpolation method for both camera views individually. Once the interpolation is carried out, those interpolation matrices are used to map the detected finger position in two camera view image to display system coordinate. The finger detection procedure is implemented using intensity profile of the background subtracted image.

2. MECHANICAL STRUCTURE

As mentioned in the introduction, the cameras used in our system are sensitive to infrared light. The source of infrared light is the set of infrared light emitting diodes mounted on a board. Direction of the infrared light is towards the camera view covering whole display area of the screen. The system use Fire-i 1394 cameras including IR (infrared) filter with pixel format Y_MONO and resolution 640 X 480. The cameras are positioned below the left and right corner of the display system pointing upwards such that both camera views contain all the area of the display screen. Images are taken from left and right camera alternatively. In this system, a finger is observed only when it is close to or touches the screen. The infrared light emitting diodes on a board are set somewhere between two cameras to cover whole display area by infrared light. The system configuration is shown in Fig. 1.

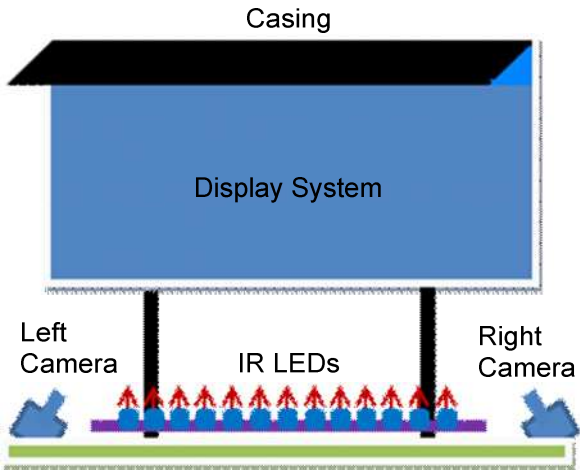


Fig. 1. System configuration

In our system the main propose of using IR light sensitive camera is only for accurate fingertip detection. The IR light is chosen because the user cannot see it but the cameras can. As the user moves his/her finger near the display surface the intensity label at the position of user finger in the image of both camera views will be brighter and finger detection becomes simple and accurate. A small black casing over the display surface is also used for improving the finger detection accuracy. Actually the casing is optional, but if there is light source just over the display screen it is necessary. The prototype of the system that we setup in our laboratory is shown in Fig.2

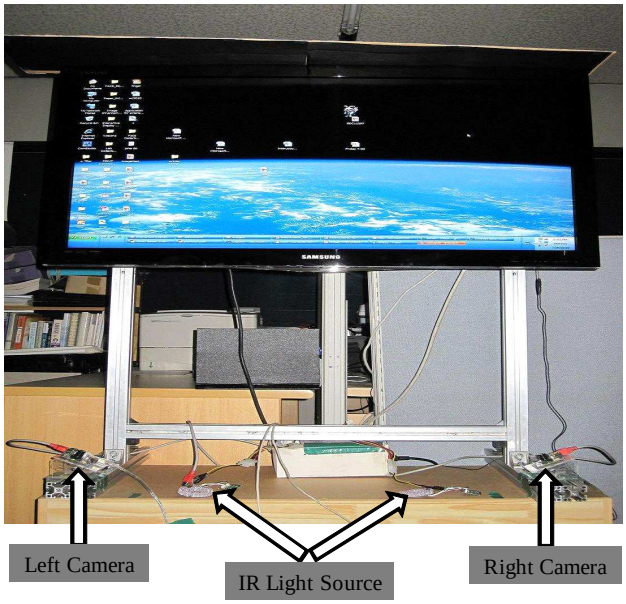


Fig. 2. Prototype of the system

3. FINGERTIP DETECTION

The fingertip detection procedure is divided into two stages. First, the region if interest (ROI) is selected and perspective transformation is carried to align the selected ROI to rectangular shape for both camera view images. Second, the resultant image form the first step is used for further processing. In this step, the vertical intensity profile of the background subtracted image is used for fingertip localization.

3.1 ROI Selection and Perspective Transformation

An ROI is selected manually at system setup time on both camera views as shown in Fig. 3 and Fig. 4.

The ROI in left camera view is selected by marking left top, right top and right bottom position of the display surface on the image. First the display surface left top position is touched by the finger and the corresponding position in the image captured by left camera is marked (left two red point in Fig. 3). The two points on each marked position indicates the operating range of the screen in terms of the distance between fingertip and surface of display system. Next, the finger is positioned over the right top position of the display surface and the corresponding position on the left camera image is marked (middle two red points in Fig. 3). At last the finger is positioned over the right bottom position of the display surface and the corresponding position on the left camera image is marked (right two red points in Fig. 3). If the user finger is outside the ROI formed by those six points, the system does not respond to the user input. Similarly the ROI on the image captured by right camera is selected as shown in Fig. 4. Here the ROI is selected by marking left bottom, left top and right top position of the display surface on the corresponding right camera image.

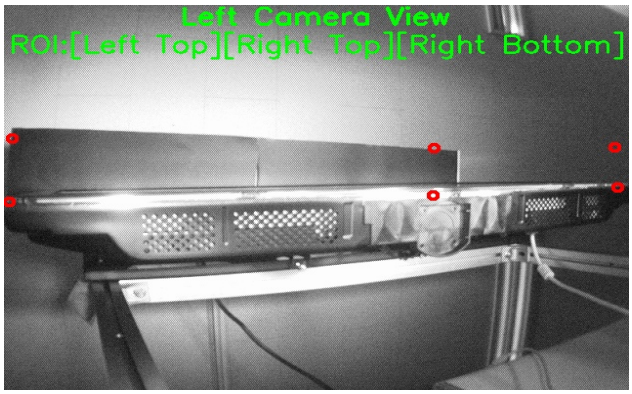


Fig. 3. Selected ROI position on the left camera view

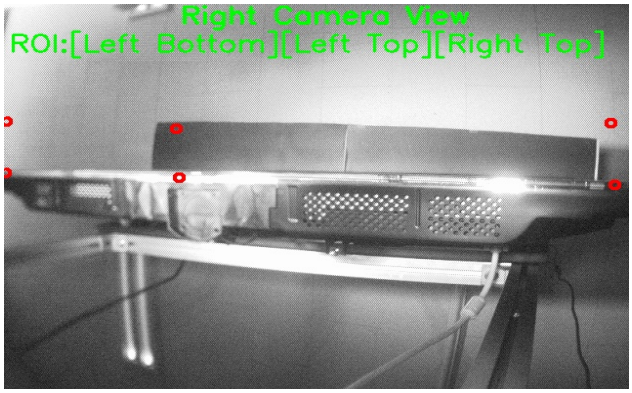


Fig. 4. Selected ROI position on the right camera view

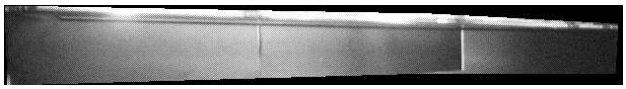


Fig. 5. ROI image from left camera captured image (Fig. 3).

The image shown in Fig. 5, ROI image from Fig. 3, is now subjected to perspective transformation. The perspective transformation maps an arbitrary quadrilateral into another arbitrary quadrilateral. Once we know the four corners of source and destination quadrilateral, following well known equations can be used to find the parameters for perspective transformation [11]:

$$X = \frac{ax + by + c}{gx + hy + 1} \quad (1)$$

$$Y = \frac{dx + ey + f}{gx + hy + 1} \quad (2)$$

where, (X, Y) and (x, y) are the destination and source coordinate respectively. The parameters a, b, c, d, e, f, g and h are found by solving Eq. (1) and Eq. (2) for four corresponding corners of source and destination quadrilaterals. Now using Eq. (1) and Eq. (2) any coordinate in the source quadrilateral can be

mapped to coordinate in destination quadrilateral. The ROI image in Fig. 5 is concatenation of two quadrilaterals. The perspective transformation is applied to each quadrilateral individually and the result is concatenated to form the resultant image which is shown in Fig. 6. Similarly the perspective transformation is applied to the ROI of right camera.

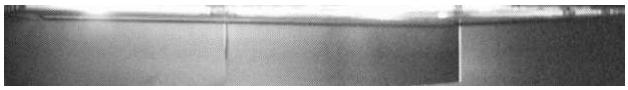


Fig. 6. ROI image after perspective transformation

3.2 Background Subtraction and Finger Localization

The goal of background subtraction is to detect all the foreground objects from given sequence of frames captured by a fixed camera. The foreground objects are detected by subtracting image of the scenes static background from the current frame. Many methods are available to obtain the image of the scenes static background from the sequence of input frames. The background image is not fixed but must adapt to illumination changes, motion changes, changes in background geometry etc. In our system we are using IR sensitive camera, so the intensity response on the user finger position in the image is bright. Thus simple background subtraction method can be used to find the user finger. On the other way our system should operate in real time, but using complicated background subtraction method will increase the computational complexity.

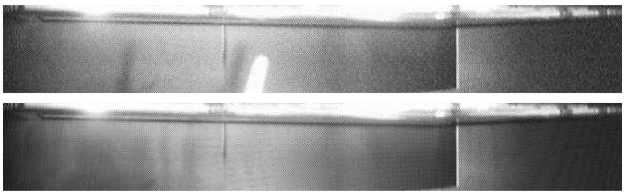


Fig. 7. Upper image: Current ROI image, and Lower image: corresponding background ROI image

The background image is calculated from the average of previous *N* frames. This process is fast but memory consuming, because *N* x *sizeof(frame)* memory is required to store previous *N* frames. This more memory requirement problem is solved by calculating background as a running average. Eq. (3) describes this process.

$$B_{i+1} = \alpha * F_i + (1-\alpha) * B_i \qquad (3)$$

where, α is the learning rate, B_i is the previous background, B_{i+1} is the current background, and F_i is the current frame. Foreground image and the corresponding background image are shown in Fig. 7.

The background image is now subtracted from the current image to determine finger in the background subtracted image. In our system in each ROI image we need to determine the horizontal position of the single object (finger). Instead of thresholding, here we determine the vertical intensity profile of the background subtracted image to locate the finger position. If the highest peak in the profile is greater than certain threshold the finger is located on the corresponding position of the ROI image. Fig. 8 shows the result of finger detection.

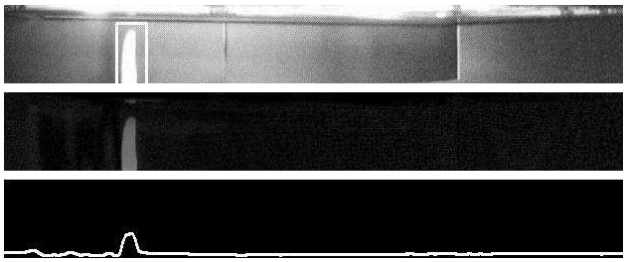


Fig. 8. Top layer: current image with detected finger position, Middle layer: background subtracted image, and Bottom layer: vertical intensity profile of background subtracted image.

4. DEPTH CALCULATION

The depth value between display surface and the user fingertip is needed for mouse manipulation. According to the measured approximate depth value mouse up, down and double click events are provided. Here, simple method for approximating the depth between user fingertip and display surface is used. The length of user finger on left and right ROI image is used for depth calculation. Let $d1$ and $d2$ be the height of the finger in left and right ROI image as shown in Fig. 8. Again, let M_d denotes the mean of $d1$ and $d2$ represented by Eq. (4).

$$M_d = \frac{d1 + d2}{2} \qquad (4)$$

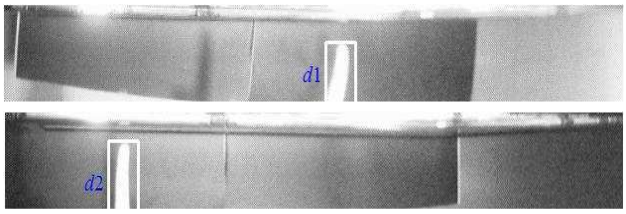


Fig. 9. Measuring finger height on both ROI image, Upper layer: left ROI and Lower layer: right ROI

The height given by Eq. (4) is inversely proportional to the real time depth between fingertip and display surface, because as user moves his/her finger near the screen surface, the height $d1$ and $d2$ in ROI images increases but real depth is decreases and

vice versa. Eq. (5) gives the approximation of real depth (Rd) between user fingertip and display surface.

$$Rd = \alpha * (H - M_d) \tag{5}$$

Where, α is a scaling factor, and can be set to any value according to mouse manipulation event and H is the height of ROI.

5. SAMPLING AND INTERPOLATION

On both ROIs only the horizontal position of the fingertip is sufficient to find the screen coordinates of the display system with respect to the fingertip position over display surface. To take samples the display surface is divided into fixed number of blocks as shown in Fig. 10, so that we know the coordinate of each corners (shown by white circle) of all the blocks. Now, pointing every corner by finger, the corresponding horizontal position of fingertip on each ROI image is noted and stored separately.

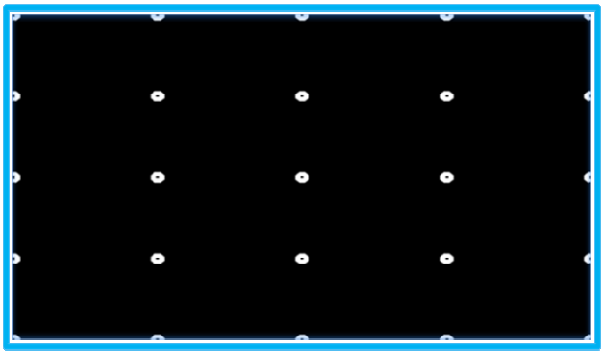


Fig. 10. Sampling position on the display system

Now we have two sample data matrices consisting of horizontal position of fingertip in each camera image ROIs at some known coordinate position of the display system. Next step is to interpolate those samples in all coordinate position of the screen. In this system bilinear interpolation method [12] for surface interpolation is implemented. The bilinear interpolation is an easy method for interpolating values on a rectilinear grid.

The function

$$f(x, y) = a_1 + a_2 x + a_3 y + a_4 xy \tag{6}$$

is called bilinear if one variable set to a constant, then the function is linear in the other variable. Suppose that we need to interpolate the values at point (x, y) between four values on a rectilinear grid. The corner coordinates are (x_1, y_1), (x_1, y_2), (x_2, y_1) and (x_2, y_2) with values v_{11}, v_{12}, v_{21} and v_{22} respectively.

The coefficients of the bilinear interpolant are determined by the values at the four corners of the grid rectangle and are computed by plugging each grid coordinate and values into Eq. (6).

$$v_{11} = a_1 + a_2 x_1 + a_3 y_1 + a_4 x_1 y_1 \tag{7}$$

$$v_{12} = a_1 + a_2 x_1 + a_3 y_2 + a_4 x_1 y_2 \tag{8}$$

$$v_{21} = a_1 + a_2 x_2 + a_3 y_1 + a_4 x_2 y_1 \tag{9}$$

$$v_{22} = a_1 + a_2 x_2 + a_3 y_2 + a_4 x_2 y_2 \tag{10}$$

Solving the four simultaneous Eq. (7), (8), (9) and (10), we can determine the coefficients of the interpolant. Once coefficients are determined the values at every (x, y) position inside the rectilinear grid can be found by using Eq. (6). This process is repeated for every grid of display system as shown in Fig. 10. Thus obtained interpolation data are stored in a matrix form for both left and right camera image ROIs separately.

6. IMAGE TO SCREEN COORDINATE MAPPING

Once interpolation matrix are determined, in order to track the fingertip position over the display surface we need only the horizontal position of the fingertip on each ROI image of corresponding left and right camera view. If the camera position or display system position are changed the interpolation matrix need to recalculate by taking new samples by selecting new ROI position. The interpolation matrices are used every time once the user fingertip is detected on both ROI image.

Let us suppose M_1 and M_2 denote the interpolation matrix corresponding to left and right camera image ROIs respectively. Again, suppose X_1 and X_2 denotes the detected horizontal position of the user finger on the left and right ROI image respectively. The flow chart of the matching procedure is shown in Fig. 11 to determine the user finger position over the display surface.

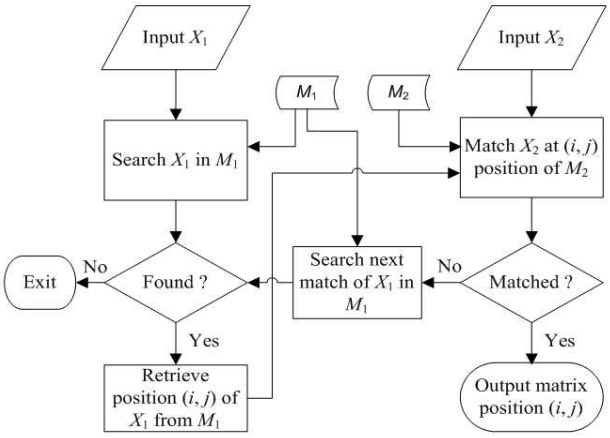


Fig. 11. Flowchart of matching procedure

The resolutions of the interpolation matrices are same all the time. If the screen resolution and interpolation matrix resolution are same, the (i, j) position returned by matching procedure described in flowchart of Fig. 10 is the screen coordinate position with respect to the user fingertip over the screen. On the other hand, if the screen resolution and interpolation matrix resolution are different we need to scale the (i, j) position returned by matching procedure for exactly matching the screen coordinate position and finger position over screen. Suppose $Hs \times Ws$ denotes the screen resolution and $Hi \times Wi$ denotes the interpolation matrix resolution. Let (x, y) denotes the screen coordinate. The Eq. (11) and (12) are used to convert from matrix coordinates to screen coordinates.

$$x = i \times (Ws / Wi) \quad (11)$$

$$y = j \times (Hs / Hi) \quad (12)$$

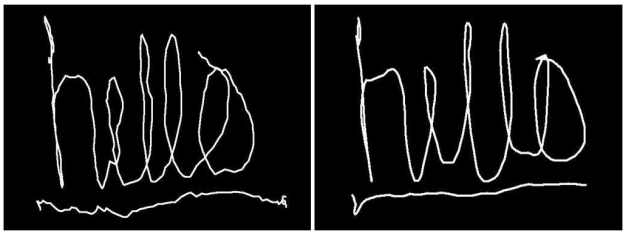
7. TEST RESULTS AND DISCUSSION

Our human-computer interaction system was tested on different system with different size and resolution as well as different visual information on the screen. The surface of the screen is not visible to both camera and hence the system is invariant to screen contents. For the testing purpose we focus on tracking the mouse and perform some mouse manipulation operations.

The sampling and interpolation is the key part of our system. The number of samples depends upon the number of divided blocks of the display surface. The system was tested by taking different number of samples and then interpolating them to standard resolution 640×480. The samples were taken by dividing 640×480 resolution screen into blocks of size 2×2, 4×3, 4×4 and 5×5. The system was tested using interpolation matrix calculated from the different number of samples separately. The experimental result shows that the number of samples taken does not affect the system performance drastically. But taking too few or too many samples does not produce good results. The preferable numbers of samples are 25 taken by dividing screen into 4×4 blocks.

Our developed interaction system was tested on both large as well as small screen: 20 inch, 40 inch and 52 inch. For big screen we need large vertical distance between camera and bottom edge of the display surface and for small screen we need small distance.

Direct system interaction according to the detected coordinate position of the screen with respect to the fingertip is little bit difficult. Because of finger instability, camera noise, illumination changes etc. the movement of mouse pointer according to finger movement is not smooth. Some flickering problem arises. Even if the user tries to locate finger at same position of the screen the detected coordinate position is not stable because user can't locate fingertip at the same position perfectly. To solve this problem we apply the simple moving average to the set of detected coordinate positions of fingertip over the screen. The window size taken for calculating simple moving average was 3, which means that the average coordinate of the three consecutively detected latest coordinates was calculated every time. Fig. 12 shows the comparison of paint drawings using finger by applying moving average and without applying moving average. In Fig. 12 (a), which is the result without applying moving average, one can see that the drawing is not as smooth as it is in Fig. 12 (b), which is the result after applying moving average.



(a) (b)

Fig. 12. Paint drawing examples using finger (a) without applying moving average and (b) applying moving average.

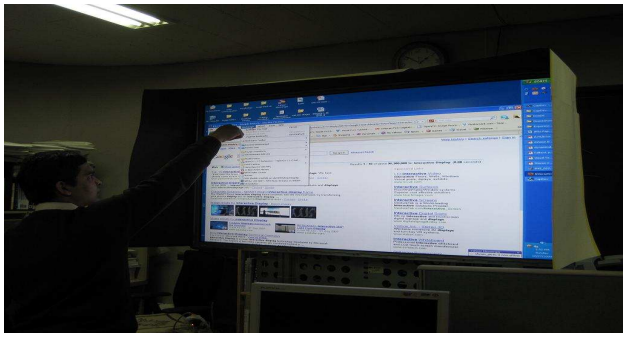


Fig. 13. Test on the system using finger for menu selection.

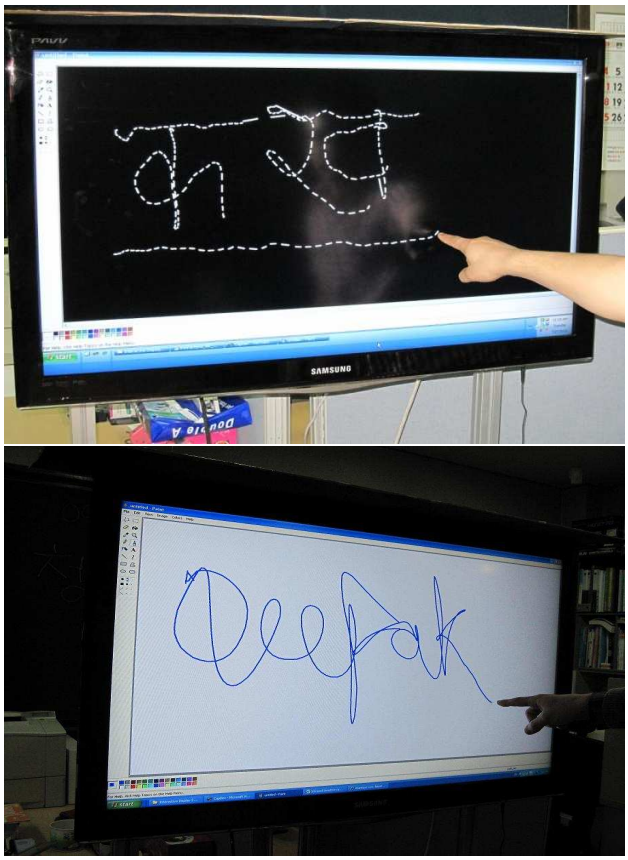


Fig. 14. Test on the paint application using finger

Fig. 15. Playing online game with the help of finger

Fig. 13 shows an example of screen manipulation. In this figure we can see that the user finger is used to select the menu of the web browser. Fig. 14 shows another example for evaluating system performance in terms of subjective point of view. The dotted line is generated by implementing mouse up and down event alternatively as the finger moves over the display surface. Fig. 15 shows another example of screen manipulation with the help of user finger.

In order to evaluate the performance of our system, we did lot of experiments with different display information on the screen. The mouse movement was smooth and can reach any portion of the display surface. For stability measure we tried to locate mouse pointer at the fixed position of the screen for some amount of time, which turns out to be very stable. We find the standard deviation of 30 consecutively detected coordinate positions taken by pointing finger at fixed position of the screen from the centre coordinate. The detected range of standard deviation was 1 to 4 pixels, which proves the stability of the system.

As described in [13], the speed and accuracy of mouse movement is not seriously degraded for latencies up to 75 ms and the latency between 25 ms and 75 ms is optimal. We also calculated the latency of our system. With 2.40 GHz of CPU, 3.25 GB of RAM, in our system one camera have 15 ms of latency. As already described the images are taken sequentially form two cameras. Hence the system gives response to the user input only when the images from both cameras were processed. So the proposed system has total of 30 ms of latency. This means that in the proposed method with above system configuration each camera can operate in a more than 66 frame/sec and can return more than 33 screen coordinate per second. This proves that the user can move his hand in a normal velocity and will not feel any delay between input and system response time.

8. CONCLUSION

In this paper we presented the vision based human-computer interface system that allows the user to interact with display system using his/her finger even without any physical contact. The strong point of the proposed system is that it can operate in any general propose computer and do not need any special system. The stability, simplicity and small latency prove that the proposed method is superior in comparison with the conventional HCI system. The experimental results show that the system can operate in real time, robust to screen contents and illumination changes.

REFERENCES

[1] Mitsubishi DiamondTouch, last visited July 20, 2010, *http://www.merl.com/projects/DiamondTouch/*

[2] Morrison G. D., “A CMOS Camera-Based Man-Machine Input Device for Large-Format Interactive Displays”, ACM SIGGRAPH 2007 courses, SIGGRAPH ’07. ACM, New York, NY, 65-74.

[3] V. I. Pavlovic, R. Sharma, T. S. Huang, “Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review”, IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 19, No. 7, July 1997, pp. 677-695.

[4] T.-S. Cheung, “Stereo Computer Vision with Multiple

Cameras", Hongkong University of Science and Technology.
[5] T. Fukasawa, K. Fukushi, H. Koike, "A Vision-Based Non-contact Interactive Advertisement with a Display Wall", ICEC 2006, LNCS 4161, pp. 394-397.
[6] I. Katz, K. Gabayan, H. Aghajan, "A Multi-touch Surface Using Multiple Cameras", ACIVS 2007, LNCS 4678, pp. 97-108.
[7] Ye Zhou, G. Morrison, "A Real-time algorithm for Finger Detection in a Camera Based Finger-Friendly Interactive Board System", *Proc. ICVS*, 2007.
[8] A. Sanghi, H. Arora, K. Gupta, V.B. Vats, "A Fingertip Detection and Tracking System as a Virtual Mouse, a Signature Input Device and an Application Selector", *Proc. of the 7th International Caribbean Conference on Devices, Circuits, and Systems*, April 2008.
[9] D.-D. Yang, L.-W. Jin, J.-X. Yin, L.-X. Zhen, J.-C. Huang, "An Effective Robust Fingertip Detection Method for Finger Writing Character Recognition System", *Proc. of the 4th International Conference on Machine Learning and Cybernetics*, August 2005, pp. 4991-4996.
[10] J. Ravikiran, K. Mahesh, S. Mahishi, R. Dheeraj, S. Sudheender, N. V. Pujari, "Finger Detection for Sign Language Recognition", *Proc. IMECS 2009*, March 2009.
[11] Academics, Perspective Transform Estimation, last visited November 11, 2010, *http://alumni.media.mit.edu/~cwren/interpolator/*
[12] R. Jain, R. Kasturi, B. G. Schunck, "Machine Vision", McGRAW-HILL International editions, pp. 382-383, 1995.
[13] I. MacKenzie and C.Ware, "Lag as a determinant of human performance in interactive systems", *Conference on Computer-Human Interaction*, vol. 1, no. 4, pp. 356, 1994.

Deepak Ghimire
He received the B.E. degree in computer engineering from Pokhara University, Nepal in 2007. In 2009, he enrolled as a student of M.E. in computer engineering at Chonbuk National University, Republic of Korea. His main research interests include image processing, computer vision, and pattern recognition.

JoonCheol Kim
He received the B.S., M.S., and Ph.D. degrees in electronic engineering from Chonbuk National University, Jeonju, Korea, in 1986, 1988, and 1995, respectively. From 1988 to 1991, he was researcher at LG Industrial System R&D Laboratory. Since 1993, he is now with the School of the Engineering in Seonam University. His research interests include image processing, computer vision.

Kwangjae Lee
He received B.E. degree in Electronic engineering from Chonbuk National University, in 1986. He also received M.S., Ph.D., degrees in Avionics from Korea Aviation University, in 1988 and 2002 respectively. From 1988 through 1993, he was a researcher at LS Cable R&D Laboratory. Since 1996, he is now with the School of the Engineering in Seonam University. He is also the academic director and vice editor of Korea Entertainment Industry Association. His research interests include HCI, Digital Art, Entertainment Technology and Serious Games.

Joon-Whoan Lee
He received his BS degrees in Electronic Engineering from the University of Hanyang University in 1980. He received the MS degree in Electrical and Electronic Engineering from KAIST in 1982, and the Ph.D. degree in Electrical and Computer Engineering from the University of Missouri in Columbia in 1990. He is currently a Professor in the Department of Computer Engineering of the Chonbuk National University. His research interests include image processing, computer vision, emotion engineering.