

Virtual Touch Screen for Mixed Reality

Martin Tosas, and Bai Li

School of Computer Science and IT, University of Nottingham
Jubilee Campus, Nottingham, NG8 1PG, UK
mtb@cs.nott.ac.uk

Abstract. Mixed Reality (MR) opens a new dimension for Human Computer Interaction (HCI). Combined with computer vision (CV) techniques, it is possible to create advanced input devices. This paper describes a novel form of HCI for the MR environment that combines CV with MR to allow a MR user to interact with a floating virtual touch screen using their bare hands. The system allows the visualisation of the virtual interfaces and touch screen through a Head Mounted Display (HMD). Visual tracking and interpretation of the user's hand and finger motion allows the detection of key presses on the virtual touch screen. We describe an implementation of this type of interfaces and demonstrate the results through a virtual keypad application.

1 Introduction

Classical interface devices used in the VR/MR environment include mice, keyboards, and joysticks, but these devices do not fully realise the potential of a VR/MR environment. More sophisticated devices such as 3D mice, wands, and data gloves can be used in a 3D environment, however, they are not only expensive but also need cables and sensors to operate, which renders the approach intrusive. The intrusion is more apparent in an MR environment, where the difference in handling interaction with virtual objects using these devices and interaction with real objects using bare hands creates a barrier between the real and the virtual world.

Computer vision techniques are the least intrusive for interaction in a MR environment. In the case of hand-based interaction, some require the use of special gloves or markers attached to the hand to facilitate the interaction. This is less intrusive than the use of data gloves or 3D mice, as these special gloves or markers are lighter and do not need any wiring. However unadorned hand tracking can make it possible for a MR user to interact with the virtual and the real in the same way, by using their bare hands.

In this paper we propose a virtual touch screen interface, operated with the user's bare hands, as a way of HCI for MR environments. In a MR environment it is possible to lay windows, icons, menus, buttons, or other type of GUI controls, etc, on the surface of the virtual touch screen. Users could see these interfaces floating in front of them using a HMD, and could interact with these virtual interfaces using their bare hands. One or many cameras can be used to track the user's hands and interpret their motion so to detect when the user is 'clicking', or pressing a button on the virtual surface. The paper is organised as follows. Section 2 of this paper, is a brief literature review on some work related to this research. Section 3 presents a realization and applications of a multi-user virtual touch screen interface. Section 4 shows the results of a virtual touch screen implementation, in the form of a virtual numeric keypad. Section 5 gives some conclusions, and future work directions.

2 Related Work

Of special relevance to this research is the digital-desk concept. The basic idea is to augment a desk by tracking the hands of the user and interpreting what is happening on the work-surface. This is achieved by means of one or two cameras situated on the top of the work-surface. The system presents information on the desk by means of a projector situated on top of the work-surface. For example, the system in [1] tracks the user's fingertip by correlating a template of the fingertip with the image on the desk. The correlation is performed only over the area surrounding the last detected finger tip position. If the system loses the track of the hand, the user has to put his finger on a square situated in one of the corners of the desk to resume tracking. Results of the method are shown in a finger drawing application.

BrightBoard [2] is a system that uses a video camera and audio feedback to enhance the facilities of an ordinary whiteboard, allowing the user to control a computer through simple marks made on the board. The system described in [3] is a 3D hand gesture interface system. It acquires gesture input from two cameras, recognizes three gestures, and tracks the user's hand in the 3D space. Five spatial parameters (position and orientation in 3D) are computed for index finger and the thumb. The capability of the system is demonstrated with some example applications: video game control, piloting of a virtual plane over a terrain, interaction with 3D objects by grasping and moving the objects. Also in [4] is described a system that tracks the user's hand using two cameras, one from the top, and the other from the side. The system allows drawing in 3D space and handling of 3D virtual objects. [5] describes three gesture recognition applications: the first allows the user to paint with a finger on a wall using virtual ink, the second allows the user to control a presentation using hand gestures, and the third allows the user to move virtual items in a brainstorming session.

Using a wearable computer and a head-mounted camera for tracking the hand, [6] is concerned with a system that allows the user to encircle an object, thereby coarsely segmenting the object. The snapshot of the object is then passed on to a recognition engine for identification. [7] introduces the concept of 'steerable interfaces for pervasive computing spaces'. This type of interfaces can be displayed on a wall or on a surface near the user. A 3D environment designer allows the user to define the geometry of the environment, surfaces where the interface could be displayed, and the positions of cameras and projectors in the environment. The technologies needed to realize this concept include projecting images at will on different surfaces, visually tracking user's head and hand orientations and positions, and to placing sound at different places in the environment.

A system called ARKB [8] is more similar to the research described in this paper, in which the user sees a virtual keyboard lying horizontally through a video-see-through HMD and is allowed to 'type' on it with both hands. However, the user needs to wear markers on the fingers to allow hand tracking by the two cameras on the HMD. Using stereo matching, the system detects when one of the markers is inside the volume of a virtual key and considers the key as being pressed. Finally in [9] is described an application that a virtual touch screen can enable - an immersive VR application of Japanese characters writing. The user can write using a virtual keyboard, and interact

with their hands over some 3D widgets arranged in a virtual desk. The system also takes speech input. However, a CyberGlove is used to recognize user's hand motion.

This paper describes a new form of HCI for MR - visual tracking and interpretation of the user's hand to allow interaction with virtual interfaces on a virtual touch screen and real objects in the same way by using the hand.

3 Virtual Touch Screen Interfaces

A virtual touch screen will be perceived by MR users as a number of GUI elements floating in front of them. MR users can interact with these elements using their bare hands by 'clicking' with their fingers, on windows, icons, menus, etc, in the same way as popular GUI like MSWindows could be operated via a real touch screen. Some advantages of this type of interfaces are:

- Can be projected wherever necessary and moved to wherever necessary.
- Can be comparatively cheaper than using screens, keyboards, mouse, or data gloves, for the amount of functionality that such a system could offer.
- Suitable for use in harsh working conditions. For example in outdoors or some other environment potentially harmful for keyboard, mouse, or screen, like dusty, humid, or underwater environments.
- Suitable for use in situations where physical contact with the user is not appropriate. For example in a hospital operating theatre, a doctor might be in the middle of an operation on a patient and need to check for some information, or select other functions of a medical AR system. They could do this by using a virtual touch screen without risk of contaminating the hands.

3.1 A Realisation of Multi-User Virtual Touch Screen Interfaces

This section describes how a multi-user virtual touch screen could be realised in an MR environment. The concepts are explained in the context of several development stages of a virtual keyboard. The choice of a virtual keyboard is a good example of a virtual touch screen use, as the hand tracking component involved in recognizing key presses on a virtual keyboard is also applicable for recognizing other types of finger 'clicks' on buttons, icons, menu items, etc. The system consists of a HMD to allow the user to see the projection of a floating virtual keyboard, and/or other interface elements like windows, icons, etc; a camera to obtain visual input; the software to track user's hands, detect keystrokes on the virtual keyboard, visualise the keyboard and produce visual or acoustic feedback when keys are pressed. Here it is assumed that a single camera is used, but other arrangements using multiple cameras are also allowed.

The easiest implementation would be a *static virtual keypad*. With static here it is meant fixed or tied to some spatial location. The scenario is that the MR user walks towards a machine and sees a projected floating hand shape near the machine. The hand shape indicates that in that position there is a virtual interface ready to be initialised and used. To initialise the virtual interface, the user needs to put his/her hand on top of the projected hand shape. The system then takes a snapshot of the hand and analyses it to initialise hand model parameters for that particular user, such as the lengths of the

fingers and finger segments, skin colour, etc. The floating hand shape also sets the initial position for the hand tracking process. If the resolution of the camera is good enough; the snapshot taken from the user's hand could be also used for identification or verification of identity purposes [10]. The hand shape would then disappear and the virtual keypad would appear in its place, ready to be operated by the hand. Visual and acoustic feedback could be produced as the system recognizes the user input, see Fig 1.

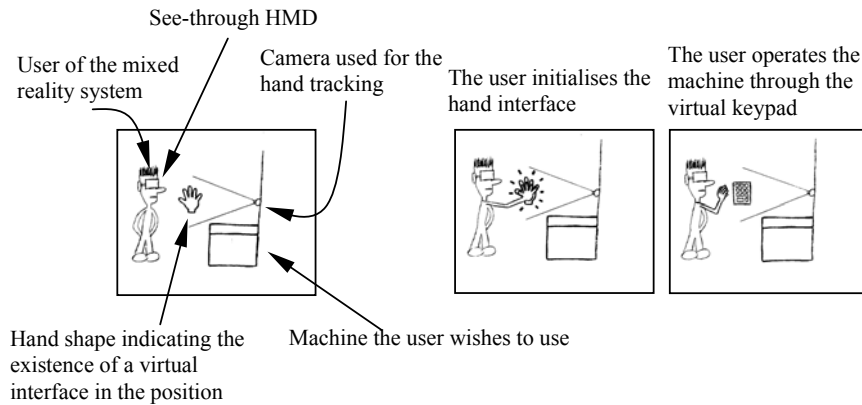


Fig. 1. Initialisation of a static virtual keypad

The tilt angle of a virtual keypad with reference to the camera could be adjusted for the user's comfort, as can be seen in Fig. 2.



Fig. 2. Keypad's tilt angle

The next stage of development would be to implement a *static alphanumeric virtual keyboard* that could be used with both hands. This is based on the previous keypad idea but in this case hand tracking is slightly more complex as it involves both hands typing at the same time. No restrictions on the hand movement are made to simplify the tracking. Other approaches can be implemented for better results, like different types of keys, different sizes, different tilt angles of the keyboard.

Finally the same concept could be repeated in the form of a *floating virtual keyboard*. In this case the camera for hand tracking is mounted on the user's HMD, in the case of a video see-through HMD this would have already been provided. The keyboard and windows can be presented in three ways in the MR environment: head-stabilized, body-stabilized, and world-stabilized [11], see Fig. 3. However, the user could reposition different virtual elements around themselves by means of some gesture or just 'clicking' with a finger on some area of a window, keyboard or any other element, or drag the window to a new position.

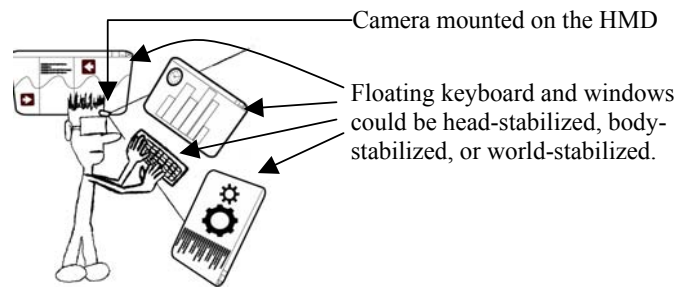


Fig. 3. Floating virtual keyboard and windows

Hand tracking in this case is from the back of the hand, which could be more difficult than tracking from the front, as in the case of a static keypad. However if all the interface elements are conceptually placed on the surface of a sphere with the user's head as the centre, then when the hand is actually typing, the view of the hand from the camera is always the same, this can be used to simplify the hand tracking algorithm.

Virtual touch screen interfaces can provide a Windows, Icons, Menus, Pointers (WIMP) interface inside a MR environment. This WIMP interface can be a very useful part of a complex multifunctional MR/AR system. A virtual touch screen can be used to select different functionalities of the MR/AR system. For example, when the user has to manipulate some virtual objects, it will make more sense to use various non-WIMP forms of interaction like speech recognition, gaze tracking, gesture recognition or body/head/arms/hands tracking. However, in a complex system some modalities could interfere with each other. In this case a menu in a virtual touch screen could allow a selection from, for example, hand/arm/body tracking for controlling a robot arm, hand pointing at a 3D location, or gesture recognition to control other type of objects. Ultimately a virtual touch screen could easily present interactive 2D information, texts, and allows the input of alphanumeric information, selection of sequential and discrete items from menus, etc. inside a MR/AR environment.

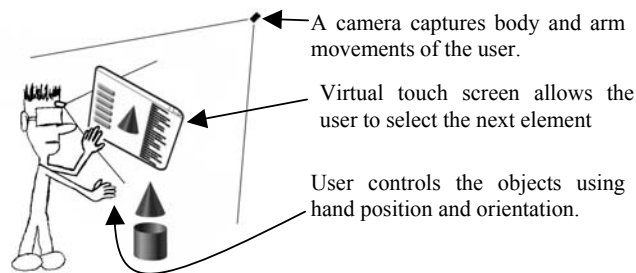


Fig. 4. Virtual touch screen complementing non-WIMP modality for an assembly task.

In Fig. 4 is shown a hypothetical MR user working on the assembling of some virtual objects. The user could select the next object to assemble and see its properties using a virtual touch screen interface and the other hand could control the assembly operation.

A virtual touch screen would not be suitable for data entry tasks, as the user would have to hold their hand on the air for a long time, resulting in considerable exertion. However it could be suitable for interactive retrieval of information, selection of items from menus, etc. In this case, the user would be clicking, for most of the time, with one of their fingers on some area of the virtual touch screen and only sporadically having to input texts. In addition, the system should allow the user to withdraw their hand from the camera field of view so that the user can have a rest, and resume the hand tracking later.

4 Virtual Keypad Implementation

To illustrate the feasibility of virtual touch screen type of interfaces, a virtual numeric keypad has been implemented. Once the user types on it, the keys pressed will appear on a notepad. The assumptions made here are, the hand is parallel to the camera's image-plane, roughly in vertical position, and against a black background.

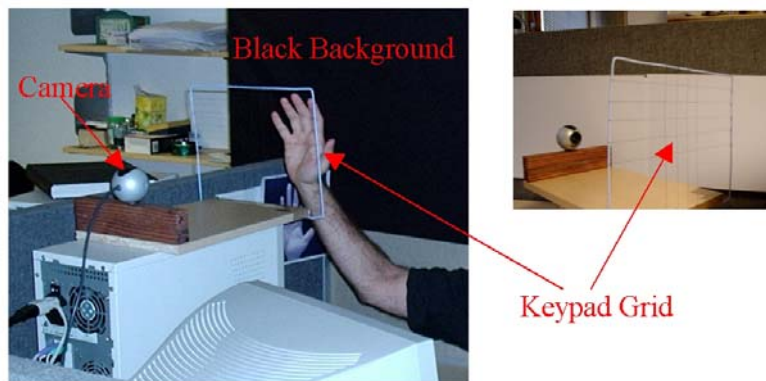


Fig. 5. Set up of the system; camera, keypad grid, and black background

The virtual numeric keypad is intended to be visible through a HMD in an MR application. However at the current stage of this research the main interest is to track a human hand and interpret the finger movements in an appropriate way so that key presses on a virtual numeric keypad can be detected. The view that provides the HMD has been substituted by a 'transparent' physical keypad. By transparent it is meant that the user can see the keypad, but tracking can still be performed through it. The first attempt to this 'transparent' physical keypad was to use a piece of glass standing at a certain distance in front of the camera and parallel to the camera's image plane and draw the numeric keypad on it. In this way the user can see a keypad and attempt to type on it as if it were a real keypad. The hand movements are then recorded and interpreted, and key presses are detected. At the same time on the computer screen a coloured keypad and the tracked hand can be seen so that some feedback is available on the screen, instead of on the HMD. Finally the 'transparent' keypad is implemented using a frame with some strings forming a grid pattern. The grid serves as the keypad, and the squares forming the grid are the individual keys. For the resolution of the camera, the strings are captured with the width of a single pixel and are later eliminated by an image processing

routine. Fig. 5. shows the set up of the system. When the HMD is used instead of the frame and the grid of strings, the MR system will have to display the keypad exactly in the same position as where the grid of strings is.

In the following sections three stages of the virtual keypad operation are described: initial hand tracking, hand model initialisation, and hand tracking with key press detection. As soon as the user puts their hand in front of the keypad, therefore in front of the camera too, the initial hand tracking starts. To initialise the hand model and start using the keypad, the user has to bring their hand, with fingers stretched and hand parallel to the keypad, closer to the keypad up to the point where the palm is touching the strings. Thereafter, hand tracking for key press detection will start.

4.1 Initial Hand Tracking

The initial hand tracking is strictly speaking hand detection. The features detected are the finger-tips, finger-valleys, and hand width. In Fig. 6. the detected finger-tips are marked with blue circles, the finger-valleys are marked with red circles, and the hand width is a horizontal red line that goes from the finger-valley between thumb and index fingers, to the edge of the hand in the other side.

As the background is black the detection of these features is easily performed using basic image processing techniques. The image pixels are analysed in a single pass searching for patterns that look like peaks and valleys. This is followed by a validation step that guarantees that the peaks and valleys found are indeed valid hand features, and not the results of noise, illumination changes, etc. The peaks and valleys found are only counted if they make up five peaks and four valleys, and their relative positions are consistent with a vertical hand. The scheme allows for small rotations of the hand but larger rotations are discarded as they can interfere with the hand tracking for key detection later on. The finger-tip and finger-valley positions will allow in the next two stages, to point out lines longitudinal to the fingers.

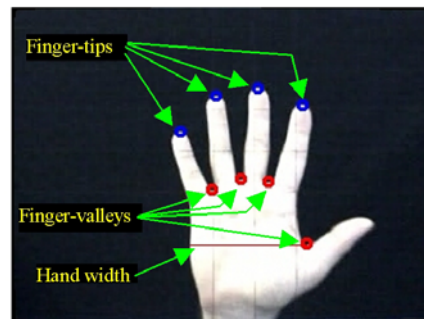


Fig. 6. Finger-tips, Finger-valleys, and Hand width.

The hand width is used as a measure of relative distance of the hand to the camera. This approach is taken because it is very simple to implement and is independent of the finger's flexion/extension, unlike other measures like the hand's area, which depends on the distance to the camera and the flexion/extension of the fingers. The distance from the

camera to the keypad is known, so when the hand is far away from the keypad, its distance can be approximated comparing the current hand width with the hand width at initialisation time. For this purpose it is important that the hand keeps a vertical orientation at all times.

4.2 Hand Model Initialisation

In section 3.1 it is described how the user has to put their hand on a floating hand shape. The purpose of this step is to take some reference measurements and initialise a hand model. For this particular implementation the hand model consists of the length of the lines longitudinal to the little, ring, middle, and index fingers (called finger lines); and the hand width. Here it is assumed that when the user has the hand inside the floating hand shape, the hand is parallel to the camera's image plane. If the hand model was more complex, then all the necessary reference measurements should be made at this point.

4.3 Key Press Detection

After the hand model is initialised, the hand tracking and key detection operate. When the user's hand goes near enough to the virtual keypad (keypad grid), the keypad displayed on the computer's screen changes colour from red to green, indicating that the hand is on reach to the keypad, ie. the user can type on the keypad from that distance. From that moment on, and while the keypad is green, the lengths of the finger lines are continuously monitored to detect changes in length that can be identified as a key presses. Given one of this length changes (a key press), the final position of the finger tip, before the finger recovers back to the rest position, is checked whether it is inside of the area of a key, in which case the key press of such a key is recognised.

In Fig. 7. is shown an example sequence that involves typing in a telephone number on the virtual keypad. The horizontal axis represents the frame number in the sequence, and the vertical axis represents the length of the finger line. In blue is the length of the Index finger line, in pink is the length of the Middle finger line, and in Yellow is the length of the Ring finger line.

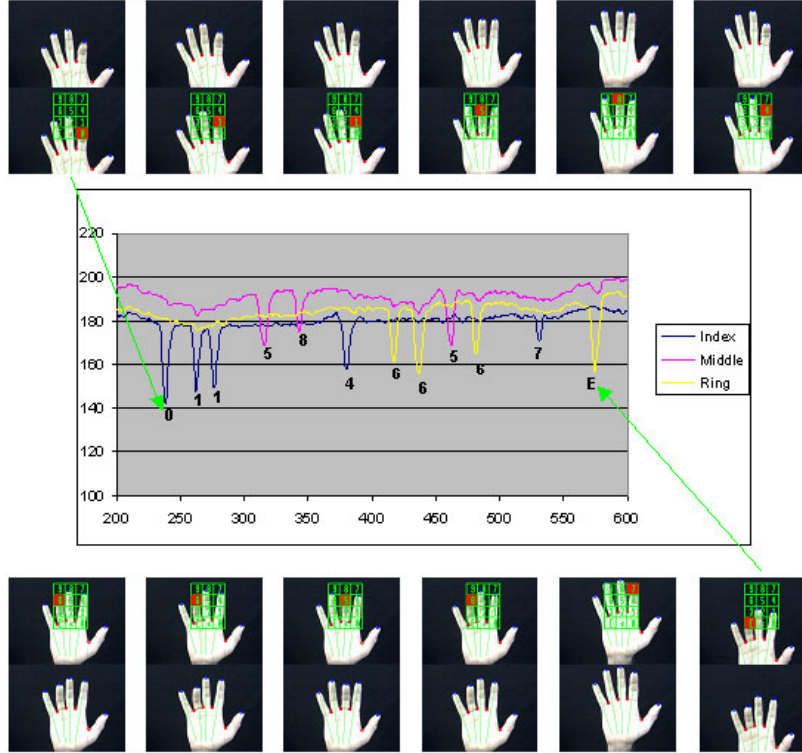


Fig. 7. Typing in a telephone number. In the chart the horizontal axis is the frame number inside the sequence, and the vertical axis is the finger line's length

4.4 Test Results

The virtual numeric keypad application was tested with a long typing sequence to check the accuracy of the system. The test involved a sequence containing 121 mostly random key presses. The input to the system was from live video, and the whole sequence was recorded as seen on screen, i.e. showing the virtual keypad, and displaying the finger lines and finger tips/valleys on the hand. After the whole sequence was recorded, a visual inspection of the recorded video allowed verifying how many true-positives, false-negatives and false-positives were produced. The results were:

Table 1. Virtual keypad test results

	Number	Percentage of the total
True-Positives	105	86.7%
False-Positives	4	3.3%
False-Negatives	12	9.91%
Total	121	

The system is slowed down by the recording of the output. This has a negative effect on the accuracy of the system. The recording frame rate had to be set at a trade-off between loss of accuracy and being able to inspect the video properly to identify true-positives, false-negatives and false-positives. The 86.7% accuracy is sufficient for the user to input numeric values quite consistently. In case of the occasional false-negative (a key is pressed but not recorded), the user can press the key again. In the case of a false-positive (a key is not pressed but is recorded as being pressed), the system provides a backspace key for the user to delete the recorded number. With the current approach, the causes of false-negatives and false-positives are due to the following:

- Breakdown in hand tracking or hand is out of the camera field of view
- Hand is not parallel to the keypad
- Self-occlusion of fingers, or cluttered background, varying lighting conditions, etc,
- Key press does not produce enough variation in the length of the finger line
- Movement of other fingers are detected as key presses.

The accuracy of the system allows for rudimentary use. In addition the system provides keys to correct the input. However the accuracy also depends on how good the user is at using this type of keyboard. There is a learning period when the user has to learn to type in a particular way. This is a limitation of the current approach.

5 Conclusion

We have presented a virtual keypad application that illustrates the virtual touch screen interface idea. With the approach used in the virtual keypad key press detection, the results presented are the best possible. There are still some possible improvements following this model, like considering the finger lines to go from the finger tips to a calculated position for the Metacarpophalangeal joint, reducing the influence of adduction/abduction movements on the length of the finger line. However the accuracy with which the position of this joint could be calculated on the image is poor using the current hand model. On the other hand, it can be seen that in order to improve the accuracy and relax the constraints of the system, ie. black background, hand parallel to the keypad, no hand rotation, typing in a particular way, etc, the hand model has to be more sophisticated and the tracking approach has to change. The desired requirements of a virtual touch screen are:

- The hand tracking has to be possible in general background conditions, uncontrolled lighting conditions, and views.
- The use of a virtual touch screen should be as natural as possible, for example if the interface on the touch screen is a keyboard, the typing should not need to exaggerate the finger movements.
- The accuracy of the system should be 100%.

Ideally, all these requirements should be met. For this purpose we are planning to use a robust 2D tracking of the hand contour using active contours and deformable template techniques [12][13]. This 2D tracking allows doing robust tracking of the hand contour, which can be interpreted in combination with a more sophisticated 3D hand model, so

that the positions of hand and fingers can be calculated more accurately. The 3D hand model can be a kinematic model of the hand, which would include the lengths of each finger segment and whatever necessary motion constraints. The hand model initialisation stage will have to be able to gather the information needed to initialise the new 3D hand model. At this point many applications can be developed. One example could be to implement a windows manager on a virtual touch screen that can run the same applications as in MSWindows platform, or other type of platforms. This windows manager would need certain adaptations so that texts can be inputted with virtual keyboards.

6 References

1. F. Berard J. Crowley and J. Coutaz. Finger tracking as an input device for augmented reality. Proc. Int. Workshop Automatic Face and Gesture Recognition, pages 195-200, 1995.
2. Peter Robinson Quentin Stafford-Fraser. BrightBoard:a video-augmented environment. Conference proceedings on Human factors in computing systems, ACM Press, pages 134-141, 1996.
3. Jakub Segen and Senthil Kumar. GestureVR: Vision-based 3d hand interface for spatial interaction. The Sixth ACM International Multimedia Conference, Sep 1998.
4. Hideo Saito Kiyofumi Abe and Shinji Ozawa. 3-d drawing system via hand motion recognition from two cameras. Proceeding of the 6th Korea-Japan Joint Workshop on Computer Vision, pages 138-143, Jan 2000.
5. Francois Bérard Christian Von Hardenberg. Bare-hand human computer interaction. Proceedings of the ACM Workshop on Perceptive User Interfaces, 2001.
6. T. Keaton S. M. Dominguez and A. H. Sayed. Robust finger tracking for wearable computer interfacing. Proc. Workshop on Perspective User Interfaces, Nov. 2001.
7. Gopal Pingali, Claudio Pinhanez, et al. Steerable interfaces for pervasive computing spaces. First IEEE International Conference on Pervasive Computing and Communications (PerCom'03), page 315, 2003.
8. Wontack Woo, Minkyung Lee. ARKB: 3d vision-based augmented reality keyboard. ICAT2003(International Conference on Artificial Reality and Telexistence), pages 54-57, Dec, 2003.
9. Yuji Y. Sugimoto Noritaka Osawa. Multimodal text input in an immersive environment. ICAT, pages 85-93, December 4-6, 2002.
10. Landa Silva, Recobos Rodriguez. Biometric identification by dermatoglyphics. Proceedings of the 1996 IEEE International Conference on Image Processing (ICIP 1996), pages 319-322, 1996.
11. Mark Billinghurst and Hirokazu Kato. Collaborative Mixed Reality. Proceedings of the First International Symposium on Mixed Reality. pages 261-284. 1999.
12. Andrew Blake and Michael Isard. Active Contours. Springer, 1998.
13. Michael Isard and J. MacCormick. Hand tracking for vision-based drawing. Technical report, Department of Engineering Science, University of Oxford, 2000.