Real-time Monocular Tracking of View Frustum for Large Screen Human-Computer Interaction

Kelvin Cheng, Masahiro Takatsuka
National ICT Australia and
ViSLAB, School of Information Technologies,
The University of Sydney,
NSW 2006, Australia
{kcheng,masa}@it.usyd.edu.au

Abstract

This paper introduces a novel approach towards direct interaction with large display systems. Monocular computer vision is utilised to avoid restraints imposed by input devices. Tracking the user's head and determining the view frustum in real-time is one of the key processes in our proposed human-computer interaction system. We also proposed using a view frustum to model the user's interaction volume allowing flexible interaction with the display. Finally, we demonstrate the feasibility of this new concept and provide an accuracy analysis of our prototype system.

Keywords: Human-computer interaction, monocular computer vision, view frustum, interaction volume, large displays.

1 Introduction

Since its mainstream introduction with the Apple Macintosh in the mid-1980s, the mouse-operated desktop graphical user interface (GUI) has become the interface of choice for personal computers (PCs). This interface was designed to support the tasks common to computers at that time, namely word processing, spreadsheets, drawing, project planning, and other "productivity tasks". These tasks are typically performed with a user sitting in a chair at a desk. The desk provides a surface for the operation of the mouse as well as placement of the monitor, typically a high-resolution display less than two feet from the user. The mouse has gone through numerous cycles in ergonomic improvement, from the earliest brick-like units with mechanical rollers to the sculpted optical wireless mice prevalent today. Various studies (MacKenzie and Jusoh, 2001, Mithal and Douglas, 1996, Shneiderman, 1987) have shown this to be a simple to use and efficient input device.

Jacob, a human-computer interaction (HCI) visionary, in his 1996 survey (Jacob, 1996), predicted that "it is likely that computers smaller and larger than today's workstation will appear, and the workstation-size machines may disappear". Today, we do indeed see that both smaller and larger computers (in terms of display size) have indeed appeared – small devices such as PDA and smart phones, and large displays such as projected wall-sized displays. As computer technology becomes more advanced, they can increasingly handle tasks beyond simple productivity applications and towards the manipulation of rich media. Large displays have become less expensive in the last few years, as have the performance of graphics processors; computer users can afford and demand more screen real estate. Large-scale display systems spanning an entire wall are widely used in many modern information technology facilities, especially for non-interactive purposes such as presentations. Where they are used interactively, the user interaction devices typically consist of a standard keyboard and mouse.

However, there are a number of reasons why these devices are less than optimal for large displays. From the outset, in 1968, Douglas Engelbart developed the mouse to provide a way for users to interact with personal computers (Engelbart and English, 1968). It was never designed to be used in a large display environment. As a result, the mouse only performs moderately well when scaled to large screens.

The computer mouse that is currently being used is a pointing device. It is an intermediary device that facilitates the user, providing a means for humans to interact with the computer. It is a tool for mapping hand movements into an on-screen cursor so that on-screen objects can be manipulated. However, this mapping only provides an *indirect* interaction. This indirectness comes about because the output space is not the same as the input space. The output space is the display (the monitor in most cases sitting on top of a desk in front of a user) while the input space is the table (the horizontal desk that the mouse lies on). This indirectness causes reduced freedom as well as reduced efficiency.

One approach to reduce this indirectness is by using our own hand as the input device, thereby throwing away the intermediary device that has restricted our mobility and freedom. This is an interesting concept because the pointing gesture is natural, intuitive, and easy to use and understand. Indeed, children can use body postures and gestures such as reaching and pointing by about 9 months (Lock, 1980). The act of pointing using the index finger to point at something is defined as the "deictic gesture"(Wahlster, 1991).

Our project will build on previous work and take another step further towards this ultimate goal. By making use of the location of the user as well as their hand, we can determine the location on the screen the user is pointing at.

The first stage in our project is the main contribution of this paper – to locate the user, determine what they see and determine the interaction space available to them. We approximate a virtual frustum between the user's eye and the large display. This is important in establishing the interaction space available to the user, between the user and the screen. The user is then able to extend their arm within this volume and interact with the screen. It is envisioned that no matter how large the display is or how far away the display is, the user will still be able to interact with the large screen in a consistent manner.

Previous approaches and the state-of-the-art techniques in this area will be presented in the next section followed by an overview of our proposed interactive system. We will then go into details about the view frustum and how it was implemented. Detailed accuracy analysis will follow and we will conclude with some final remarks.

2 Related Work

In the human-computer interaction history, most of the earlier input devices did not take into account what the user sees and are mostly intrusive.

2.1 Handheld Intrusive Devices

Intrusive devices for HCI are ones where, in order to use them, the user must wear or don some kind of device, or hold with their hand.

The *lightpen* and the *light gun* were some of the first pointing devices produced (Myers, 1998). These provide a direct approach to interacting with CRTs. Although they have good accuracy, their major drawback is that they must be used with a CRT display.

The *Logitech Spaceball*, *Gyration Gyromouse* and *Interlink RemotePoint* represent another category of input devices designed for personal use as a mouse replacement. The advantages include wireless ability, more affordable and more natural. However, the user still interacts through an intermediary device.

Recent inventions such as the handheld *Personal Digital Assistant* (PDA) and *graphics tablets* use a stylus device, such as a specialised pen, to perform input. Such devices are only useful for personal use, where they allow direct interaction. However, they are still indirect when used with a large display because the user performs input on the handheld device and receives feedback on the large display on the wall.

Laser pointers have the advantages of mobility, direct interaction, and being comparably inexpensive, with the notable disadvantages of lag and instability with the human hand. Many studies into these systems have been carried out (Kirstein and Muller, 1998, Sukthankar et al., 2000, Olsen and Nielsen, 2001) where a normal red laser point on the screen is captured by a video camera.

2.2 Hand Tracking and Gesture Recognition

DiamondTouch (Dietz and Leigh, 2001) is a touch sensitive table from Mitsubishi Electric Research Laboratories (MERL) that can detect and identify multiple and simultaneous users' touches. Antennas are placed under the surface of the table, each with a different electrical signal. When the user touches the table, a small electrical circuit is completed, going from a transmitter to the table surface to the user's finger touching the surface, through the user's body and onto a receiver on the users' chair. One restriction with this approach is that the user must be seated to operate this device.

Rekimoto proposed a similar approach with SmartSkin (Rekimoto, 2002). Rather than using electricity, it relies on capacitive sensing by laying a mash of transmitter and receiver electrodes on the surface. When a conductive object (such as the human hand) comes near the surface, the signal at the crossing point decreases, enabling the detection of the multiple hand position. They are also able to detect the hand even when it is not actually touching the surface.

Computer vision has been a major technique in the HCI research community. It is mostly used to track different parts of the human user in much of the research surveyed.

Various researchers have been dealing with tracking hands above surfaces. The most notable is the implementation of DigitalDesk (Wellner, 1993) which used a projector and a camera looking down from above a normal desk. It allows users to use their finger, detected by using finger motion tracking, and a digitizing pen. It supports computer-based interaction with paper documents, allowing such tasks as copying numbers from paper to the digital calculator, copying part of a sketch into digital form and moving digital objects around the table.

In similar research, rather than aiming a camera at a desk, the camera was directed at a vertical whiteboard. Magic Board (Crowley et al., 2000) used a steerable camera so that screens can be larger than the field of view of the camera. Rather than using motion detection, cross-correlation is used, which extracts small regions from an image (e.g. image of a pointing finger) as a template for searching in later images. Hardenberg and Berard (Hardenberg and Berard, 2001) also used a camera which captures hand activities on a whiteboard. They developed an improved hand segmentation and fingertip detection algorithm and demonstrated their implementation. They used bare-hand-pointing on the whiteboard surface as a mouse replacement, as well as using the number of outstretched fingers to control the movement of presentation slides. The SMART board (SMART, 2003) fixes 4 tiny cameras on four corners of its display which can detect any objects that come into contact with the surface, or hovers above it. IBM developed a "multi-surface interactive display projector" so that it can make any surface in the room interactive (Pinhanez, 2001). It is done by attaching a rotating mirror so that any surface in the room can be projected onto, and captured by the camera. Finger detection is performed on the same surface that is being projected, generating a "click" event as if it is a computer mouse.

The use of infrared has been used to remove the need for colour hand segmentation and background subtraction due to the fact that they sometimes fail when the scene has a complicated background and dynamic lighting(Oka et al., 2002). In HoloWall (Matsushita and Rekimoto, 1997) and Barehands (Ringel and Henry Berg, 2001), infrared LEDs are emitted from behind the wall, as well as from a back-projected projector. An infrared filtered camera is also positioned behind to detect the presence of hand or finger when it comes near the wall (within 30 cm) which reflects additional infrared light. EnhancedDesk (Oka et al., 2002) uses an infrared camera from above a horizontal desk to track objects with a temperature around 30-34 degrees C (human body temperature). They also developed a fingertip detection algorithm that makes use of a reduced search window. They are able to track multiple hands and fingers, predict fingertip trajectories in real-time, as well as recognise symbolic gestures using Hidden Markov Model (HMM). The Perceptive Workbench (Leibe et al., 2000) uses infrared illuminated from the ceiling and a camera under a table. When the user extends their arm over the desk, it casts a shadow which can be picked up by the camera. A second camera fitted on the right side of the table captures a side view of the hand. Combined together, the location and the orientation of the user's deictic (pointing) gesture can be computed. The approach assumes that the user's arm is not overly bent. It fails when shadow from the user's body is cast, as well as when two arms are extended at the same time.

All of these techniques require that the user be at the location they want to point at, requiring large movement of the arm as well as pacing across the surfaces. They will not work well when surfaces are hard to reach.

Maggioni and Kammerer developed a computer system that is able to detect 3D position and orientation of human hands under noisy and changing environments in real-time. (Maggioni and Kammerer, 1998). A camera is placed on top of the computer and is used to capture the hand. A virtual 3D model of the user's hand is displayed on the screen, and thus movement of the real hand will cause the virtual hand to move. They used two approaches for detecting the position and gesture of the hand. In one approach, a marker is attached to a glove that the user wears so they can calculate the x, y and z rotation of the hand. The second approach finds human skin colour, and searches for fingertips by locating a T-like structure. They are then able to recognise six static hand gestures based on the number of visible fingers and their orientation.

To further Magonni's work, Segen and Kumar (Segen and Kumar, 1998) developed a vision based system, GestureVR, which allows hand gesture interaction with a 3D scene. Two cameras are used to track the thumb, the pointing finger, three simple gestures (point, reach and click), as well as the hand's 3D position and its orientation in real-time. Background subtraction is used to find the boundary of the hand and curvature at the boundary is measured to detect fingertips.

The problem with such solutions is that they are still indirect interactions, since they do not point to objects directly. Gestures must be learned to indirectly control the user interface and objects on-screen. In addition, the interaction is restricted to a small display area only. The user's hand must be right under the camera looking downwards. Users are not free to move around. They must also sit in front of the computer, restricting the user's freedom.

2.3 Head Tracking

The *Polhemus FasTrak* and *Logitech 3D Mouse* belong to a category of industrial strength tracking systems primarily designed for 3D motion tracking. FasTrak is an electromagnetic tracking system that computes the position and orientation of a tiny receiver as it moves through space. The major problem, however, is its vulnerability to electromagnetic interference and radiation, particularly from the monitor. In addition, this system has a limited range of three metres and a latency of four milliseconds. It is primarily designed for 3D motion capturing in a Virtual Reality environment. The 3D mouse is another similar tracking system. It uses a stationary triangular transmitter which emits ultrasonic signals to track the movement of a smaller triangular receiver. This resolves the problem of interference from radiation, but introduces interference by other equipment that uses ultrasonic signals. The system also has a limited range of two metres and a high latency of 30 milliseconds. These are typically used for CAD object manipulation and Virtual Reality which are cumbersome and expensive, costing up to US$6000.

Another category of input devices tracks the position of the head of the user and moves the mouse cursor on the display correspondingly. These are primarily developed to provide full mouse control to people who cannot use their hands but have good head control. Devices in this category consist of the Synapse Head Tracking Device and Origin Instruments HeadMouse. SmartNav is a consumer product from NaturalPoint (NaturalPoint, 2004) which is a hands-free mouse developed for people with disabilities. An infrared camera is placed on top of the monitor facing the user. Within the camera are four infrared LEDs which illuminate infrared light. A small reflective dot is worn on the user's forehead, cap, glasses or microphone boom. When the user moves their head, the camera detects the dot's movements and translates that motion into mouse cursor movement on screen.

The Perceptual Window (Crowley et al., 2000) uses head motions to control the scrolling of a document both vertically and horizontally, effectively controlling the window view point of the document. Skin colour is detected by using a ratio of histograms of normalised colour with a table look-up.

2.4 Combined Approaches - hand and head tracking

Perhaps the most direct form of interaction is being able to point at something with your hand without any restrictions such as walking up to a particular surface. These systems have the advantages of having no physically touchable surfaces and are thereby highly suitable for hygienically demanding environment such as factories or public spaces.

Depending on the system set-up, they usually allow users to interact with the display wherever they are standing.

The Hand Pointing System (Takenaka, 1998) developed by Takenaka Corporation uses two cameras attached to the ceiling to recognise the three-dimensional position of a user's finger. A mouse click is mimicked by using a forward and backward movement of the finger.

The Free-Hand Pointing (Hung et al., 1998) is a similar system that also uses a stereo camera to track the user's finger, by first detecting and segmenting the finger with a global search. The fingertip and finger orientation is then determined in a local search. Apart from the "finger-orientation mode" where the line from finger to display is determined from the finger orientation, they have also introduced the "eye-to-fingertip mode" where the line from finger to display is determined from the eye (either left or right eye) to fingertip. They found that the latter approach is more susceptible to noise because of the higher resolution given by the larger distance between eye and fingertip. However, users need to raise their hand high enough so that it is between the eye and the display.

Colombo developed the PointAt System (Colombo et al., 2003) which allows users to walk around freely in a room within a museum, while pointing to specific parts of a painting with their hand. Two cameras are set-up to detect the presence of a person by using a modified background subtraction algorithm as well as skin colour detection. The tip of the pointing hand and the head centroid is then extracted. By using visual geometry and stereo triangulation, a pointing line is then deduced. This method can also be applied to more than two cameras, and does not require manual calibration. Dwell clicking is used in this implementation.

Similar to Colombo, Nickel and Stiefelhagen (Nickel and Stiefelhagen, 2003) also used a set of stereo cameras to track the user's hand and head to estimate the pointing direction in 3D. Pointing gesture is recognised by using a three-phase model: *Begin* (hand moves from arbitrary position towards pointing position), *Hold* (hand remains motionless while pointing) and *End* (hand moves away from pointing position). They found that adding head orientation detection increases their gesture detection and precision rate. Comparing three approaches to estimate pointing direction, they found that the hand-head line method was the most reliable in estimating the pointing direction (90%). The others were forearm direction and head orientation.

There are a number of problems associated with using two cameras (Takatsuka et al., 1998). One is the reduced acquisition speed, as there is a need to process two images entirely to locate the same point. Compared with monocular techniques, it is easier to find the exact location of a certain object in the scene using stereoscopic view, and this is the reason why many gesture based computer vision researches have been based on using stereo cameras. However, no literature has been found using monocular vision to allow the use of remote hand pointing gesture whilst being non-intrusive.

3 Overview of Our Interaction System

Our goal is to come up with a natural interactive system for large screens that improves on previous solutions. To provide natural interaction, the system must allow users to point at the display directly using their bare hand. The idea is to place a webcam above the screen and have it pointing downwards so that it can sense the location of the user. To determine the user's pointing direction, a line would be drawn from the user's eye through the user's pointing finger that intersects with the screen. The resulting position on the display would be the user's intended target. An on-screen cursor would be shown at the same location that the user is pointing (figure 1).

We use a widely accepted fundamental mathematical principle: that we can draw exactly one straight line that goes through two different points, and in our case, two points in 3D space. Having also known the location of the webcam it is then possible to construct an interaction volume to model the interaction with the origin at the eye of the user. This can be achieved by determining the exact location of the user using face detection. The user is then able to use their hand and finger within this volume and point at certain on-screen objects. This would give an illusion that the display is directly in front of the user and all they have to do is point at this virtual display. To determine the pointing direction, image processing must be performed to track the hand and fingertips. In geometric terms, we are trying to find two points in 3D space which would give us a straight line that intersects the screen and thus giving us a 2D coordinate in terms of the display.

We have therefore designed a system that is simple, direct, natural and easy to use. Hand gesture is non-intrusive as the hand is the input device, and users do not need to hold any specialised devices, nor do they need to wear any special gear, or touch any surfaces. By using computer vision they are not tied down to any particular location – therefore, they are free to move around and still be able to point at the screen easily (unconstrained and untethered).

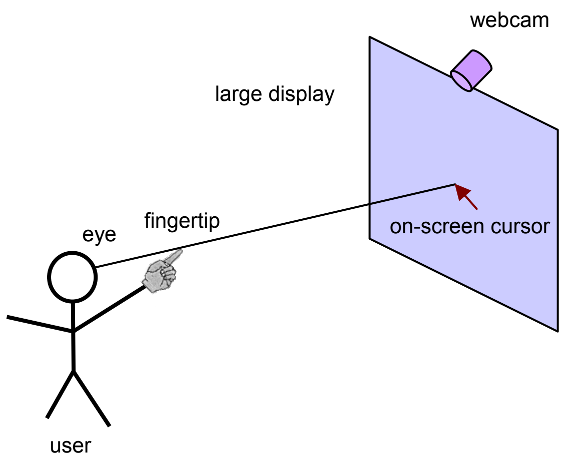

Figure 1: Overview of our interactive system. A webcam tracks the user's eye and fingertip. A line is extracted from these two points and extended to the display area where the on-screen cursor is calculated.

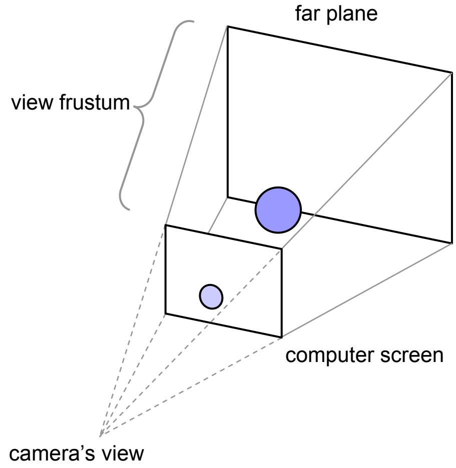

Figure 2: View frustum in computer graphics is the area within a rectangular pyramid between a near plane (computer screen) and a far plane. Only object(s) within the view frustum are drawn on the computer screen.

We have only required users to use the pointing gesture because it is the simplest and easiest gesture to perform.

We have also decided to use a consumer web camera instead of any other specialised equipment since it is cheap and widely available. Our decision for using only one camera is that we only need to analyse one video stream rather than multiple streams. In addition, computation is reduced as there is no longer a need for matching multiple streams (e.g. stereo matching in a two camera setup). This in turn makes it much easier and simpler to allow real-time computation. A single camera is more suitable in some situations such as when the system is designed for personal use at the home or in the office, or when the budget needs to be kept at a minimum. One of the research challenges is, therefore, to find methods and techniques that make single camera systems as accurate as multi-camera set-ups.

The rest of this paper will focus on the first process – the determination of the view frustum.

4 The View Frustum

In computer graphics, a view frustum is the field of view of a camera, or a region of space which may appear on the screen (figure 2). The frustum is usually the area within a rectangular pyramid intersected by a near plane (usually a computer screen) and a far plane (Wikipedia, 2004). Objects outside the view frustum are not drawn to improve performance. It defines how much of the virtual world the computer user will see (Sun, 2004).

An interaction volume is an area where the interaction occurs between the user and the system (the display, in our case). In computer vision based systems, this area must be within the field of view of the camera(s). Users can then use their hand and fingers within this area to define the point of interest on the screen.

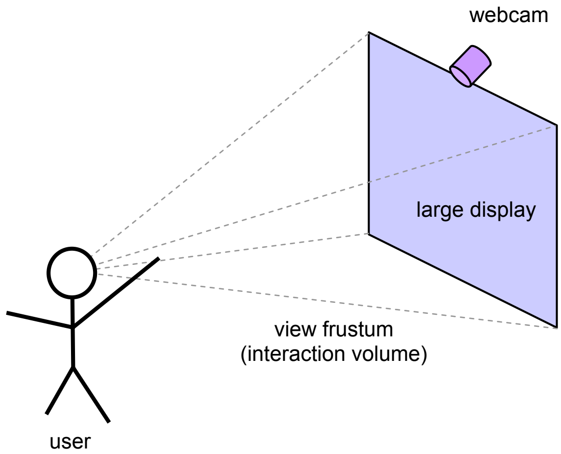

Figure 3: The view frustum is estimated by detecting the user's head and eye, with the origin at the eye.

In human-computer interaction systems that do not require the explicit knowledge of the user's location, the interaction volume is static. To adequately interact with the system, the user must adjust to the volume's location by pacing or reaching out. In addition, because the interaction volume is not visible, users must discover it by trial and error. On the other hand, when the user's location is known, the interaction volume can adjust to the user, and is always in front of the user.

To achieve this, a camera can be used to detect the face of the user. A view frustum can then be constructed between the origin (at the eye position) and the large display (figure 3). The view frustum can therefore be used as a model for approximating the interaction volume. The user can use their hand and fingers within this volume to interact with the display.

4.1 Depth Determination

As mentioned, we define one end of the view frustum as the eye of the user. Specifically we define this point as the mid-point between the two eyes, the mid-eye ***e*** (figure 4). To determine this point in 3D space, we need to first use a face detection algorithm on each frame we receive from the web camera. However, this only gives us the x and y coordinates in terms of the web camera. For our approach to work we need to also determine the third dimension z – the distance between the camera and the eye ***ce***.

Over the years various approaches have been used to determine depth in computer vision. Most of the previous work, as seen in section 2, has been dealing with stereo cameras in which depth can be determined by stereo matching. Of the ones that only require a single camera, Murphey et al. (Murphey et al., 2000) detected depth by comparing two frames taken from a moving vehicle. However, in our situation, our camera will not be allowed to move. In another study, Cantoni et al. (Cantoni et al., 2001) used a texture analysis approach as well as a histogram inspection approach to determine depth. Both approaches can only, "determine whether a point within an

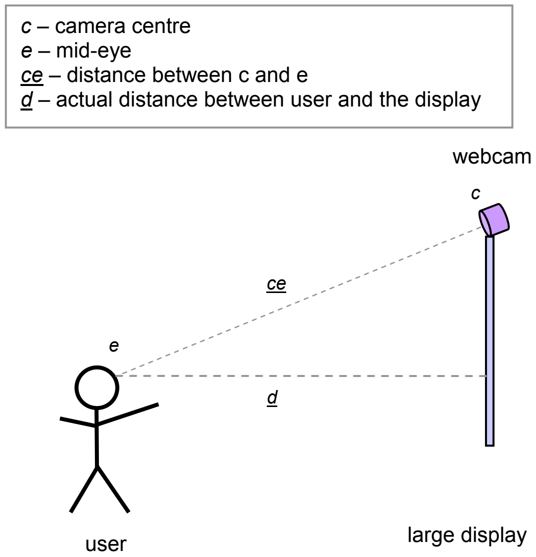

Figure 4: One end of the view frustum is determined to be the mid-eye location of the user. By using the size of the face, the distance between the user and the display can be calculated.

image is nearer or farther than another with respect to the observer". We require an approximation that can estimate the distance of the user with respect to the camera, not against each other.

Our approach is to determine the depth from the width of the face. For example, if we assume the face to be 17cm wide, when the user moves forward their face size, in terms of the number of pixels captured by the web cam, would increase. When they move backward, the face size would decrease. With this approach, we need to either:

(1) Assume that all users have similar face size (e.g. 17cm), or

(2) Add a calibration phase at the start to measure (manually or automatically) the actual human face size.

For simplicity, we assume that users have similar face size.

5 Implementation

The viability of the proposed model is demonstrated by implementing a proof-of-concept prototype using Visual C++ and an open source computer vision library – OpenCV – started by Intel Research's Visual Interactivity Group and now available on sourceforge.net (OpenCV). It provides image processing, recognition, measurement, and geometric image processing functions which is needed for our implementation. Using the Haar Face Detection algorithm (a cascade of boosted classifiers working with haar-like features) proposed by Viola (Viola and Jones, 2001) and improved by Lienhart (Lienhart and Maydt, 2002), we were able to detect the human face in real-time. To increase the tolerance and robustness, we also used an eye detector (OpenCV, 2004) which searches for the two eyes using a similar classifier cascade but is trained for detection of frontal views of eyes. If only the face or the pair of eyes is detected, the recorded mid-eye position will be used, but if both are detected, the average will be used. A sample screenshot is illustrated in figure 5. It should be noted that different techniques could have been used for face and eye detections to the same effect. The face and eye detections occur at every frame and the user's distance is re-estimated every time.

5.1 Limitation

The face detection algorithm can only detect frontal face images, therefore the user's head cannot be tilted too much away from the neutral position in all three axis, yaw, pitch and roll. In the next section we will test the robustness of the head detection technique. One implication is that the camera cannot be positioned too high above the display since users typically look towards the screen and not directly at the camera.

As noted earlier, we assumed that the head size is constant for the purpose of this proof-of-concept prototype. At its current state, to allow other users to use the system, the face size needs to be adjusted manually, or a face calibration stage added.

Our system can only accurately detect faces between the

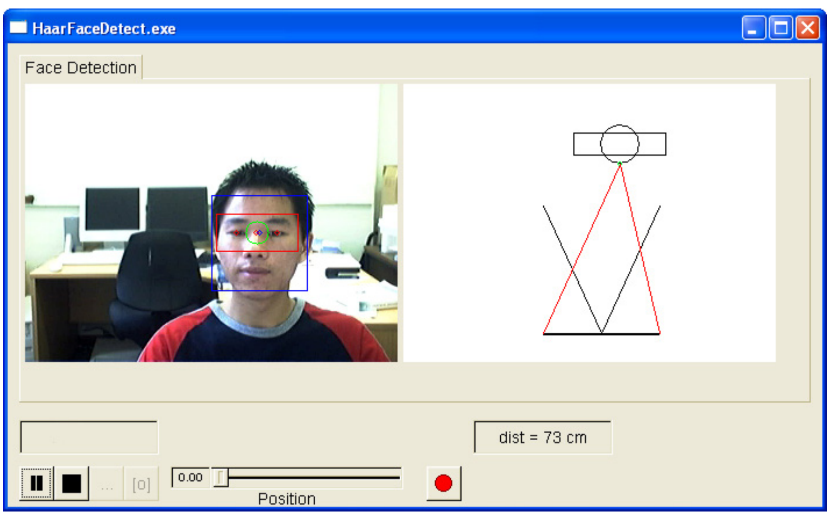

Figure 5: Screenshot of our current implementation.

Left: view of the webcam. The large square indicates the detection of a face. The smaller rectangle indicates the detection of a pair of eyes. The circle represents the averaged mid-eye position.

Right: a top view of the estimated position of the user (head is represented by a circle). The inverted "V" extended from the user's head indicate the view frustum. The black "V" indicates webcam's field of view. The horizontal black line at the bottom represents the display.

range of approximately 30cm and 180cm away from the camera.

6 Experiment and Procedure

To evaluate the performance of our system, we conducted an informal experiment to determine the accuracy of our system. We also tested the detection rate of the two different detection algorithms as well as the effect of rotation of the face.

We marked on the floor distances at 30cm intervals up to 180cm. For each distance, we took ten different positions within the field of view of the camera, ensuring that the whole face was inside the camera's view. We then recorded the distance estimated by our system, as well as whether the face or the eyes were detected at all. To control for rotation effect, effort was made to ensure that the user's head was facing the camera as orthogonal as possible. At the end of the experiment, we measured the amount of rotation the system can cope with before it can no longer detect the face. It should be noted that all measurements were done manually.

6.1 Results

It can be observed from table 1 and figure 6 that our system can provide a good estimation of distance. Its peak performance is at a distance from 90cm to 150cm. At 30 and 60cm, it always overestimated the distance, while at 180cm, it always underestimated the distance. Table 2 suggests that the face detection algorithm performs best from 60cm to 150cm. If the user is too close to the camera the whole face might not be visible to the camera, and if too far away, the face might be too small for detection. As for the eye detection algorithm, it works best when it is up close, and its performance reduces incrementally until it cannot positively detect any eye features from 120cm onwards. Aggregating all data, we see that the best operational distance is from 90cm to 150cm. It is also observed that the eye detection might not be as helpful as originally thought.

Results for the head rotation (table 3) shows that our system is robust enough to accept small head rotations in each direction, but is most vulnerable to head tilting.

Actual distance (cm)	Estimated distance (cm)			Mean accuracy (%)
	lowest	mean	highest	
30	39	40	43	67
60	64	67	72	83
90	78	86	101	96
120	107	117	127	98
150	140	149	162	99
180	150	164	169	91

Table 1: Numerical results of the estimated distances at each distance, as well as the mean accuracy that the system produced.

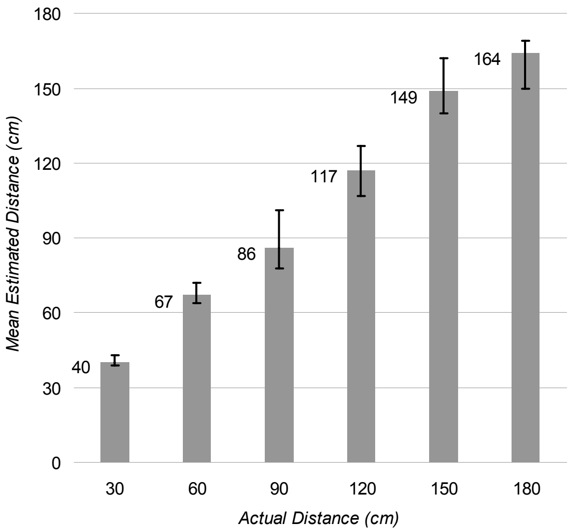

Figure 6: Graph of distance estimated using face detection.

Actual distance (cm)	Face detection rate (%)	Eye detection rate (%)
30	40	100
60	100	90
90	100	70
120	100	0
150	100	0
180	70	0

Table 2: Numerical results showing face and eye detection rate at each distance.

Head Rotation	Positive rotations (degrees)	Negative rotations (degrees)
Pitch (looking up or down)	40	20
Yaw (turning right or left)	30	30
Roll (tilting right or left)	10	10

Table 3: Estimated maximum head rotation on each axis before non-detection at a distance of 100cm. (Values are estimated manually)

7 Conclusion and Future Work

We have presented an overview of our interactive system designed for large screen interaction using natural hand pointing gesture. We have also discussed in detail our conceptual model of a view frustum. Using monocular computer vision with an inexpensive webcam, we were able to implement the view frustum successfully in

OpenCV and analyzed the accuracy of our proof-of-concept prototype. We were able to get an accuracy of at least 67% and an optimal performance of 99%. We see that our system performs best when the user is at a distance of around 0.9 to 1.5 meters away from the camera and is robust enough to cope with small head rotations.

While conducting this research, we have come across several design issues that may require further investigations. We have assumed that users line their fingertip up with their eyes to point at objects of interest. It is also possible to estimate the pointing direction from the forearm or the pointing finger. A formal user study has not been found to investigate the style of pointing users prefer and we intend to conduct such study in the future.

Where the eye-fingertip style is used, as it was in our system, there is also the question of whether users would line up their finger with a point mid-way between the eyes, or would one or the other eye dominate. User experiment along this line would be very beneficial.

Our next step will be to implement our interactive system fully as described in our system overview. This involves detecting the fingertip of the human hand as well as computing the resulting location of the mouse cursor. An initial informal usability experiment will be conducted to test the performance of our prototype.

8 Acknowledgement

This project is funded by the CMCRC (Capital Markets Cooperative Research Centre Limited).

National ICT Australia is funded by the Australian Government's Department of Communications, Information Technology and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Centre of Excellence program.

9 References

Cantoni, V., Lombardi, L., Porta, M. and Vallone, U., Qualitative Estimation of Depth in Monocular Vision. In *Proceedings of the 4th International Workshop on Visual Form*, (Capri, Italy, 2001), Springer-Verlag, pp.

Colombo, C., Bimbo, A. D. and Valli, A. (2003) *IEEE Transactions on systems, man, and cybernetics,* **33,** 677-686.

Crowley, J. L., Coutaz, J. and Berard, F. (2000) *Communications of the ACM,* **43,** 54-64.

Dietz, P. and Leigh, D., DiamondTouch: A Multi-User Touch Technology. In *Proc. UIST '01*, (2001), ACM Press, pp 219-226.

Engelbart, D. C. and English, W. K., A Research Center for Augmenting Human Intellect. In *AFIPS Conference Proceedings of the 1968 Fall Joint Computer Conference*, (San Francisco, CA, 1968), pp 395-410.

Hardenberg, C. v. and Berard, F., Bare-Hand Human-Computer Interaction. In *Proceedings of the ACM Workshop on Perceptive User Interfaces*, (Orlando, Florida, 2001), ACM Press., pp 1-8.

Hung, Y.-P., Yang, Y.-S., Chen, Y.-S., Hsieh, I.-B. and Fuh, C.-S., Free-Hand Pointer by Use of an Active Stereo Vision System. In *Proceedings of 14th International Conference on Pattern Recognition (ICPR)*, (Brisbane, 1998), pp 1244-1246.

Jacob, R. J. K. (1996) *ACM Computing Surveys,* **28,** 177-179.

Kirstein, C. and Muller, H., Interaction with a Projection Screen Using a Camera-Tracked Laser Pointer. In *Proceedings of the Conference on MultiMedia Modeling (MMM '98)*, (1998), IEEE Computer Soceity, pp 191-192.

Leibe, B., Starner, T., Ribarsky, W., Wartell, Z., Krum, D., Weeks, J., Singletary, B. and Hodges, L. (2000) *IEEE Computer Graphics and Applications,* **20,** 54-65.

Lienhart, R. and Maydt, J., An Extended Set of Haar-like Features for Rapid Object Detection. In *Proceedings of International Conference on Image Processing*, (2002), IEEE, pp 900-903.

Lock, A., Language Development: past, present and future. In *Bulletin of British Psychological Society 33*, (1980), pp 5-8.

MacKenzie, I. S. and Jusoh, S., An Evaluation of Two Input Devices for Remote Pointing. In *Proceedings of the 8th IFIP International Conference on Engineering for Human-Computer Interaction 2001*, (2001), pp 235-250.

Maggioni, C. and Kammerer, B. (1998) In *Computer Vision for Human-Machine Interaction*(Eds, Cipolla, R. and Pentland, A.) Cambridge University Press, pp. 23-51.

Matsushita, N. and Rekimoto, J., HoloWall: Designing a Finger, Hand, Body, and Object Sensitive Wall. In *Proc. UIST '97*, (1997), ACM Press, pp 209-210.

Mithal, A. K. and Douglas, S. A., Differences in Movement Microstructure of the Mouse and the Finger-Controlled Isometric Joystick. In *Proceedings of the CHI '96*, (Vancouver, 1996), ACM, pp 300-307.

Murphey, Y. L., Chen, J., Crossman, J. and Zhang, J., A Real-time Depth Detection System using Monocular Vision. In *SSGRR conference*, (2000), pp.

Myers, B. A. (1998) *ACM Interactions,* **5,** 44-54.

NaturalPoint SmartNav. http://www.naturalpoint.com/smartnav/, date accessed: 1/9/2004

Nickel, K. and Stiefelhagen, R., Pointing Gesture Recognition based on 3D-Tracking of Face, Hands and Head-Orientation. In *Proceedings of ICMI '03 International Conference on Multimodal Interfaces*, (Vancouver, 2003), ACM Press, pp 140-146.

Oka, K., Sato, Y. and Koike, H. (2002) *IEEE Computer Graphics and Applications,* **22,** 64-71.

Olsen, D. R. and Nielsen, T., Laser Pointer Interaction. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI'01)*, (Seattle, WA, 2001), ACM Press., pp 17-22.

OpenCV http://sourceforge.net/projects/opencvlibrary/, date accessed: 5/8/2004

OpenCV forum, posted by Yusuf Bediz http://groups.yahoo.com/group/OpenCV/message/18953, date accessed: 28/7/2004

Pinhanez, C., Using a Steerable Projector and a Camera to Transform Surfaces Into Interactive Displays. In *CHI 2001*, (2001), IBM T J Watson Research Center, pp 369-370.

Rekimoto, J., SmartSkin: An Infrastructure for Freehand Manipulation on Interactive Surfaces. In *Proc. CHI '02*, (2002), ACM Press, pp 113-120.

Ringel, M. and Henry Berg, Y. J., Terry Winograd, Barehands: Implement-Free Interaction with a Wall-Mounted Display. In *Proc. of CHI 2001*, (2001), ACM Press, pp 367-368.

Segen, J. and Kumar, S., GestureVR: Vision-Based 3D Hand Interface for Spatial Interaction. In *ACM Multimedia Conference*, (1998), Bell Laboratories, pp 455-464.

Shneiderman, B. (1987) *Designing the User Interface: Strategies for Effective Human-Computer Interaction,* Addison-Wesley Publishing Company.

DViT: Digital Vision Touch Technology, White Paper, SMART Technologies Inc, 2003 http://www.smarttech.com/dvit/DViT_white_paper.pdf, date accessed: February

Sukthankar, R., Stockton, R. G. and Mullin, M. D., Self-Calibrating Camera-Assisted Presentation Interface. In *Proceedings of International Conference on Automation, Control, Robotics and Computer Vision 2000*, (2000).

Sun Microsystems Inc., definition of View Frustum, http://java.sun.com/products/java-media/3D/forDevelopers/j3dguide/glossary.doc.html#47353, date accessed: 31/8/2004

Takatsuka, M., West, G. a. W., Venkatesh, S. and Caelli, T. M. (1998) Curtin University of Technology, School of Computing, Perth, WA, Australia.

Takenaka Corporation, Input System Which can Remotely Click the Screen Without hand Contact Developed http://www.takenaka.co.jp/takenaka_e/news_e/pr9806/m9806_02_e.htm, date accessed: 5/7/2004

Viola, P. and Jones, M., Rapid Object Detection using a Boosted Cascade of Simple Features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, (2001), IEEE, pp 511-518.

Wahlster, W., User and discourse models for multimodal communication. In *Intelligent User Interfaces*, (New York, 1991), ACM Press., pp 45-67.

Wellner, P. (1993) *Communications of the ACM,* **36,** 87-96.

Wikipedia, the free encyclopedia, definition of View Frustum http://en.wikipedia.org/wiki/View_frustum, date accessed: 1/9/2004