Visual Gesture Interfaces for Virtual Environments

Rochelle O'Hagan
Co-operative Research Centre for Advanced Computational Systems
The Australian National University
Rochelle.OHagan@syseng.anu.edu.au

Alexander Zelinsky
Research School of Information Sciences & Engineering
The Australian National University
Alex.Zelinsky@anu.edu.au

Abstract

Virtual environments provide a whole new way of viewing and manipulating 3D data. Current technology moves the images out of desktop monitors and into the space immediately surrounding the user. Users can literally put their hands on the virtual objects. Unfortunately techniques for interacting with such environments have yet to mature. Gloves and sensor based trackers are unwieldy, constraining and uncomfortable to use. A natural, more intuitive method of interaction would be to allow the user to grasp objects with their hands and manipulate them as if they were real objects.

We are investigating the use of computer vision in implementing a natural interface based on hand gestures. A framework for a gesture recognition system is introduced along with results of experiments in colour segmentation, feature extraction and template matching for finger and hand tracking and hand pose recognition. Progress in the implementation of a gesture interface for navigation and object manipulation in virtual environments is discussed.

1. Introduction

1.1. Overview

The advent of virtual environments introduces a whole new set of problems for user interfaces. The creation of 3D objects and worlds in which the user is immersed allows such people as scientists, engineers, doctors and architects to visualise complex structures and systems with high degrees of quality and realism. Shutter glasses provide a stereo or 3D view of the scene, which is no longer confined to a desktop monitor, but may be a large table, projection screen or room. The limiting factor in these systems is currently the interaction.

Virtual environments attempt to create a world where the interaction feels real. In many ways, the user's ability to interact with the pictures is as important as the quality of the pictures themselves [1]. Current mechanical, acoustic and magnetic input devices track the user and allow control of movement, selection and manipulation of objects in virtual scenes. However these interactions are often limited and unintuitive, while the devices are awkward, unwieldy and prone to distortion from the physical environment. We are interested in developing an alternative, natural interface that more closely models the way we interact with the real world. The user should be able to reach out, grab, point and move 3D objects just as we do with real objects.

We are investigating techniques for vision-based gesture recognition and have developed a framework for a vision-based gesture interface to virtual environments. Our system allows the user to manipulate objects within the environment in a natural manner. Manipulations include selection, translation, rotation and resizing of objects and also changing the viewpoint of the scene (eg. zooming in or out). Additionally, the system allows the user to navigate or 'flythrough' 3D data. This paper reports on work in progress in the development of a vision based gesture interface for navigation and manipulation of 3D objects within a virtual environment.

1.2. Background

In developing natural and effective methods of interacting with virtual environments it is important to first identify the properties an interface should have, the tasks required and how existing technologies attempt to fulfil these requirements. This includes determining an appropriate set of gestures for natural, intuitive, direct manipulation of objects. In selecting the gesture set, consideration should be given to: the nature and use of the human hand; the naturalness of the movements or gestures for the tasks required; and the ease of identifying and classifying the movements or gestures.

This requires investigation of several areas; namely interaction requirements, human gesture and gesture recognition techniques.

1.2.1. Interaction Properties. In order to achieve immersion within a virtual environment, the user must be able to interact effectively with the virtual world. By "effectively", we mean minimising cognitive load and maximising goal success. In 2D interaction, a good user interface is characterised by being powerful yet easy to learn [2]. Similarly, a successful 3D user interface should be natural, intuitive or at least easy to learn, and powerful enough to allow the user to accomplish the required tasks.

To achieve such an interface, interaction must be fast, accurate and natural. Similar to 2D interfaces, direct manipulation provides a natural way of working with virtual objects. Direct manipulation interfaces feature a natural representation of objects and actions to hide the feeling of performing tasks through an intermediary (the computer). The underlying belief is that allowing the user to directly perceive and interact with the virtual objects will lead to a more natural and effective interface. Accuracy is important so that both user and the interface are interacting with the same object. Low latency is also important so that the user sees the results of the interaction in real time, ie. there is minimal lag between an action and the response.

1.2.2. Interaction Tasks. Virtual environments are used in many application areas including design [3, 4], visualisation [5, 6], training [7, 8, 9], education [10] and teleoperation. There is a multitude of tasks the user may wish to perform within these areas. Many of these will be application or domain-specific, for example, changing the layout of doors or windows in a room in an architectural design system, or moving along a drillhole in a mine modelling application to determine where the highest concentrations of a particular rock type are. However, most complex virtual environment interaction tasks can be broken down into a set of basic "building blocks" - similar to the approach proposed by Foley for 2D graphical user interfaces [2].

By identifying the basic interaction tasks and implementing them in a natural way, improvements should be seen in the useability and effectiveness of interaction with the virtual environment. Hand [11] and Bowman [12, 13, 14] identify similar sets of universal tasks for virtual environments; namely object selection, manipulation, and viewpoint control. Viewpoint control refers to users interactively positioning and orienting their viewpoint within the scene. Since head tracking is usually used to determine the orientation of the viewpoint, this task primarily concerns translation of the viewpoint or moving to various positions within the scene, including zooming in or out. Selection is the task of specifying an object or objects for some purpose, and includes selection of items for application control (eg. selecting menu items). Manipulation refers to the positioning and/or orienting of a virtual object or objects. Selected objects may be manipulated in space, or selected for another purpose such as altering of object attributes (eg. colour, texture), resizing, deletion or editing.

In terms of deciding an optimal method for interacting with a virtual environment, we believe it is important to consider the best way of implementing each task. For each of the universal interaction tasks, there may be many *interaction techniques*. For example, an object may be selected by choosing the appropriate entry in a list of selectable objects, or by pointing with a mouse, tracked stylus or hand. Each of these methods achieves the task, but uses a different interaction technique, and may use a different input device. There are advantages and disadvantages of each technique, and the choice of a particular technique may be dependent on the available hardware, user preference and experience, or the precision required by the task. As well as providing for the universal tasks common to most virtual environments, interaction techniques should be designed with a view to the application-specific tasks.

1.2.3. Current Devices. Interaction techniques and input devices for virtual environments typically consist of two areas: tracking of the user's head, hands or whole body to allow the user to specify the translations and rotations; and extra controls such as buttons for selection, mode changes, etc.

Tracking systems are available using a variety of technologies ranging from mechanical devices to magnetic, acoustic or optical trackers. The trackers are typically limited to a working volume determined by the range of the device, and most require a physical connection between the user and the system, usually in the form of a sensor or receiver attached to the body and associated cabling. In some cases specific clothing such as a glove is worn.

The limitations of being physically connected to the system constrain the ability of the user to move naturally within the environment and often induce awkward motions to achieve the desired manipulation. With the exception of gloves, the devices are all suited to tracking a single object. In order to track the various parts of an articulated object such as the hand, multiple sensors are necessary. Glove-based trackers can provide information about the various joint positions which is useful in determining hand poses, however obtaining this data without requiring the glove or cabling would be more comfortable and would also allow multiple users to move in and out of the environment without having to share specific equipment.

In contrast to the above interface tools, vision-based gesture recognition provides a natural interface to virtual environments that combines the advantages of glove-based devices with the unobtrusive nature of vision based tracking. Users can use their hands to manipulate virtual objects without being connected to the system in any way. One or more video cameras observe the user and the images are processed to determine the position, orientation and pose of the user's hands. Recently there has been considerable interest in gesture recognition, and in the use of vision-based interfaces to virtual environments [15, 16, 17, 18].

1.2.4. Human Gesture. It is well known that gesture forms a major part of human communication. One definition of gesture is "body movements which are used to convey some information from one person to another" [19]. The form of gestures may vary, but most cultures use gesture to convey information in addition to speech. However, classical definitions of gesture seldom refer to the use of the hand as a manipulator. Humans are quite familiar with direct manipulation of objects. Even as infants, we have the ability to grasp objects. Once grasped, we can use our hands to change the object's position, orientation and shape. Alternatively, we can use another object or tool to make these changes. Direct manipulation gesture-based interfaces for computers should, therefore, be intuitive and familiar to users. In an HCI context, the definition of gesture should be extended to include the manipulative use of the hands to allow the user, as Pavlovic suggests, "to perform tasks that mimic both the natural use of the hand as a manipulator, and its use in the human-machine communication" [20].

In implementing an interface for virtual environments that directly involves the use of the hands as manipulators, an examination of the way we grasp and manipulate objects may help in determining a natural and efficient gesture set. The human hand has a skeletal structure consisting of some twenty-seven bones and sixteen joints, combining to provide over thirty degrees of freedom. Many of these degrees of freedom are coupled by the nature of ligaments, tendons and muscles [21]. Knowledge of these couplings and of the anatomy of the human hand can aid in developing a suitably constrained model of the hand for gesture recognition.

As well as through physical constraints, the problem of gesture recognition can be further constrained by limiting the hand movements to be recognised. If the set of tasks the user wishes to complete is itself limited, and providing the set of recognisable movements is natural and intuitive, this restriction shouldn't affect the effectiveness and useability of the interface.

2. Vision based gesture recognition system

The first stage of our vision-based gesture interface is to provide a means for the user to navigate through the large datasets and scenes often used in virtual environments. In work by Segen [22], navigation through a landscape was accomplished using the fingertip as an indicator of the direction of travel. We have chosen to utilise the flexibility of the wrist and the whole hand for navigation. By tilting the wrist up and down while holding the hand flat, the user can fly higher and lower with the model. Similarly, rolling the wrist and changing the direction of the hand initiates a turn. This is an intuitive set of movements which should be familiar to users. Figure 1 shows an example of this form of navigation.

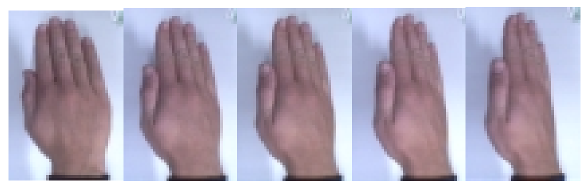

Figure 1 Movement sequence for a roll to the right

The system is based on a Barco Baron projection table. The Barco Baron projection table provides a virtual working environment of approximately $1.5m^3$. The user views images projected on the screen in 3D through CrystalEyes stereo shutter glasses. Graphics for the display are generated by an SGI Onyx2.

A twin camera system mounted above the projection table is used to provide stereo images of the user and, more specifically, the user's hand(s). The camera system consists of two Sony EX37 colour video cameras and a multiplexer to interlace the two images into one frame. Camera calibration is carried out to determine the intrinsic and extrinsic camera parameters necessary for computing accurate 3D positions [23, 24].

Image processing is carried out on a Hitachi IP5005 colour image processing board installed in a Pentium 90 PC under Linux. The IP5005 allows a wide range of image processing functions to be applied to colour video images at frame rate including inter-image arithmetic operations, filtering, normalised correlation, histograms and labelling. Camera images are digitised by the video input section and copied into image memory, which is capable of storing up to forty 512x512 pixel image frames. Each image is then de-interlaced to return the stereo pair. Further processing is applied to left and right images individually. Camera images and data in image memory can be converted to analog and displayed on an NTSC monitor.

A socket-based data link connects the Onyx2 and the PC in order to transmit position and event information to the application program. The system setup is shown in Figure 2.

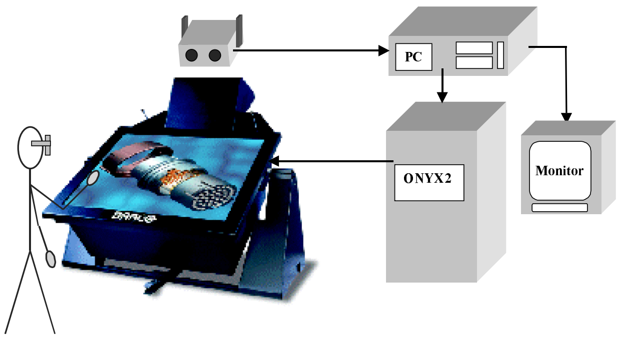

Figure 2 System setup

3. The gesture recognition process

The system is implemented following a framework for gesture recognition that we have developed. The task of gesture recognition can be thought of as two problems: determining the position and orientation of the hand or its parts; and identifying particular gestures or movements within the application context. Solving these problems involves a gesture recognition process that identifies the hand, calculates its 3D position and orientation, classifies its pose and updates the application. The process can be summarised as in Figure 3. Our system currently focuses primarily on the first problem of identifying the position and location of

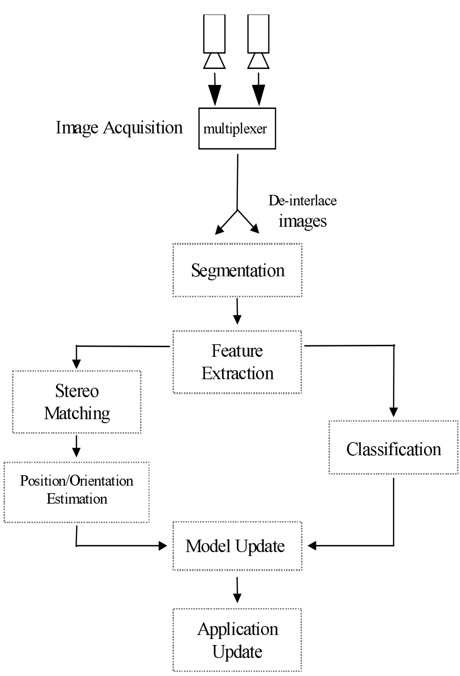

Figure 3 The gesture recognition process

the hand, although some progress has also been made in recognising particular gestures for object manipulation using moment and signature analysis. Discussion of the implementation of each stage in the process follows.

3.1. Segmentation

Normalised skin colour detection is used to find the hand within the images. A skin colour model is developed in YUV colourspace for skin detection. In each frame, those pixels within the volume of colourspace likely to be skin are extracted. Sensitivity to changes in lighting conditions is reduced by using a normalised model.

Following detection, several filtering steps are applied to determine the hand region. The largest connected region of extracted pixels is chosen as the hand, and any small holes within the region filled. This allows a mask to be created and applied to the original intensity image for extraction of the hand. Using the extracted intensity image provides more information than straight segmentation. Typical results of the segmentation process are shown in Figure 4.

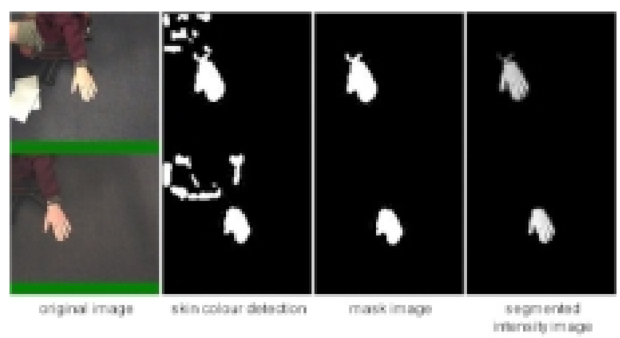

Figure 4 Image segmentation

3.2. Feature extraction

Following segmentation, further processing is applied to detect various features within the images. A combination of techniques based on geometric properties and template matching is used to determine the position of features such as the wrist, fingertips and base of fingers.

Geometric properties such as areas of high curvature can be used to indicate regions of interest such as the probable location of fingertips or wrist-hand junction. By examining the outline of the hand region, points of local minima and maxima curvature can be found. Templates can then be acquired in these areas of interest in the left image, and matched using image correlation techniques to the corresponding region in the right image as shown in Figure 5.

The templates are also used to match image features in subsequent frames. However because the visible portion of the hand changes as it is rotated, it is not possible to always match the same features in subsequent images. Also, due to the high similarity of various parts of the hand to each other (eg. the fingers all look the same), it is possible that between frames the templates will lock onto different features. For the navigation interface, tracking specific features is less important than ensuring that the features in each image are matched correctly so this problem can be disregarded.

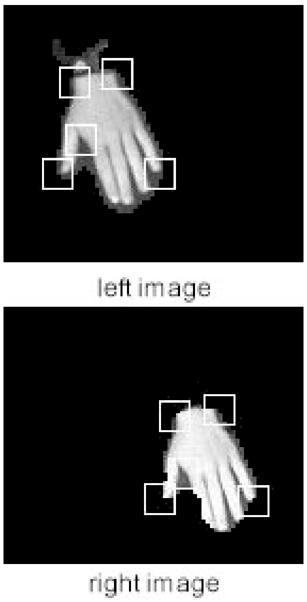

Figure 5 Template correlation

By matching multiple templates, a set of corresponding left and right image coordinates can be obtained, and through triangulation and knowledge of the camera parameters, the 3D position of the feature points determined. The 3D feature point locations are then used in updating the position and orientation of the hand model and subsequently the application.

3.3. Model update

The gesture recognition system maintains an underlying model of the hand based on the kinematic model of the hand in [25]. The model provides an estimate of the position, orientation and pose of the hand. Model parameters include the position and orientation of the hand as a whole, the position of fingertips and the base of the fingers, joint locations and wrist angles. Parameters are updated each frame from the feature values. Joint locations can be estimated from the locations of fingertips and the structural constraints of the human hand. A CAD representation of the model is shown in Figure 6.

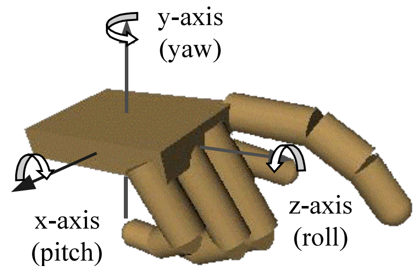

Figure 6 Kinematic hand model

For the implementation of gesture navigation, hand position and orientation parameters are used. Given the set of 3D feature points and the assumption that a flat hand is generally planar, the orientation of the hand in 3D can be estimated by fitting a plane to the points.

Packets containing the points are transmitted from the image processing PC to the SGI Onyx2 where they are used to estimate the hand plane. A least-squares approach is taken to find the best-fit plane. A minimum of three points is required, but in general at least five feature points are used. The equation of the plane, its normal vector, and the angle of the wrist with respect to the hand together describe the orientation of the hand in terms of rotations about the three axes as shown in Figure 6. The angles of pitch and roll are determined by projecting the normal vector onto the axes, while yaw is given by the angle between the principle axes of the wrist and hand. These angles can be used directly by the application, or can describe the “rate of change” of the rotation.

3.4. Application update

Having determined the position and orientation of the hand, it remains only to notify the application program of the changes and allow appropriate actions to be taken. For the navigation interface, this involves converting the hand data into an equivalent change in the view direction. Instead of using the angle of the hand directly, the viewpoint orientation can be thought of in terms of a continuous rotation, with the amount of angle indicating the rate of change in the orientation. This is similar to the method used to control flight simulators and results in an interface where the user can “fly” their hand over the terrain or dataset.

Transforming the pitch, roll and yaw angles into a corresponding rotation of the viewpoint is achieved by adding the angles to the current viewpoint rotation angles.

In future stages of development where multiple gestures are used, other information will be sent to the application to indicate specific events such as object selection or manipulation.

3.5. Gesture classification

Classification of hand gestures involves determining the particular pose of the hand and/or its motion, and deriving the meaning given a gesture set and the application context. In some situations, the pose of the hand along with its location in 2D or 3D space is sufficient information to determine the meaning and perform the appropriate action. In others, the temporal nature of gesture is required as well. In either case, a method of classifying various poses and movements is necessary. Several techniques have been used for classification in gesture recognition systems, including feature vectors [26], hidden Markov models [15, 27, 28], and neural networks [19, 29, 30, 31, 32].

The classification step in our gesture recognition system is still under development, but uses feature vectors to determine the user's gesture. For navigation, it is assumed that the user is showing a flat hand and so classification is not required. For manipulating objects, a classification system will be developed for the chosen gesture set. It should be noted that our system does not attempt to classify all movements and poses of the hand, but is constrained to the defined (but expandable) gesture set. The gesture set for object manipulation is described briefly below, and shown in Figure 7.

3.5.1. Selection. Without doubt the most important task in interaction is the ability to select - an object, menu item, tool from a tool bar or a viewpoint. Selection is also the task most demonstrated in gesture recognition systems. The gesture used for selection in these systems is typically that of a pointing finger, in the same way as we would naturally point to physical objects in the real world. Identifying the selection gesture is straightforward. The detection of a single finger outstretched, with the rest forming a fist clearly distinguishes this gesture from the others in the gesture set. The location of the "pointer" is defined to be the tip of the finger.

3.5.2. Object/scene translation. The standard "drag and drop" metaphor for moving objects can be extended to closely match the way we move objects in the real world. For example to move a wooden block we would pick it up, move our arm to the destination location and let the block go. In simulating this method of interaction we chose to let the user grip the virtual object and perform the translation as if the object were real. The thumb and fore-fingertip can be recognised and located in 3D space. This gives the points of contact on the object in question. Tracking the thumb and finger until they "release" the object or close into a fist allows the destination to be determined.

3.5.3. Object/scene rotation. In a similar manner to object translation, object rotation is usually achieved by picking up the object and then rotating the wrist. The points of contact between the fingertips and the object determine the axes of rotation. With large rotations, the object is often rotated a small amount, released and then grasped again to rotate further. This prevents the wrist from twisting further than is comfortable.

Again mimicking the approach chosen with object translation, object rotation is carried out by grasping the object - either in the same grip as the translation, or by a larger grip as in the rotation sequence in Figure 7. The axes of rotation can then be determined and rotations achieved by rotating the wrist.

3.5.4. Object resizing. The question of how an object should be resized in a "natural" way is an interesting one as there are few directly equivalent tasks in the physical world. We cannot, for example, change the size of a wooden block. The closest we come to this is to stretch or squeeze objects of certain materials to change their shape.

However, in virtual worlds, it is a common task to change the size of an object. A method for resizing objects that seems intuitive would be to simulate stretching the object in question by either grasping the object with both hands and moving the hands away from each other to enlarge, or towards each other to reduce. Alternatively, a one-handed approach could have the thumb and forefinger moving apart or towards each other as in the rotation sequence in Figure 7. An option for non-uniform scaling of objects requiring two-handed interaction could use one hand to grasp the object at one end and the movement of the other hand would stretch or compress the object at the second hand's point of contact.

Figure 7 Gesture set

3.5.5. Scene zoom. A common zooming technique for virtual environments mimics the action of reaching for an object and involves selecting a "zoom tool" (which effectively changes the mode of interaction) and then moving the input device towards or away from the screen to zoom in or out of the scene. This is easily replicated in a gesture interface by having a recognisable hand pose to set the mode to "zoom" and then tracking the motions of the hand until the mode (or hand pose) changes.

4. Gesture interface issues

There are several issues concerning the use of gesture as an interface. One of these is the idea of feedback to the user. A gesture-based interface requires the user to interact with objects that have no physical representation. This is in contrast to traditional interfaces where the user interacts with the objects indirectly through a physical device. Recent developments in haptics [33] provide a solution to the feedback problem by providing force-feedback to the user when they 'touch' a virtual object. In this way, the user can literally feel the physical properties of the object. However, current systems are limited. Most provide only point contact to the objects, rather like feeling an object with a pen. Additionally, many systems are not co-located with the virtual objects, so the user's hands aren't in the same physical space as the objects they are feeling.

Gesture, while not giving the user the benefit of touch, allows the user to interact with the objects in the exact space that they appear and to remain unconstrained by the physical restrictions of mechanical devices.

Another issue to consider in developing gesture-based interfaces is user fatigue. From observation, it is clear that manipulating objects in the volume in front of the user can quickly become tiring. This problem can be reduced by careful selection of the gesture set, and understanding of the function of the hand.

The effects of these issues cannot be properly evaluated until the system is fully implemented and useability testing conducted.

5. Conclusion & further work

We have investigated the use of gesture as an interface for virtual environments and in particular the use of computer vision techniques in developing such an interface. Gesture recognition offers a natural, unobtrusive method of interaction, especially if the gesture set is selected with some knowledge of the nature of human hand movements. Vision techniques eliminate the need for gloves or restraining cabling back to the computer and so provide an unobtrusive interface.

Examination of previous work in this area provided a basis for the development of a framework for a vision-based gesture interface, as well as identification of promising techniques. The recognition process can be broken into steps; namely segmentation, feature extraction (including stereo matching & 3D location determination), model update, classification and application update. Each of these steps can be implemented in many ways and experiments have been conducted to explore various techniques.

A system for navigating in 3D worlds within virtual environments has been developed. The user can "fly" through the scene by tilting their flat hand up or down, rotating it left or right and by twisting the wrist.

The hand is segmented from video images via normalised skin colour detection and is stable under varying lighting conditions. A combination of geometric properties and template correlation is used to find and match feature points in the two stereo images, and in subsequent image frames. The 3D coordinates of these feature points are then calculated. By fitting a plane to the points, the orientation of the hand can be estimated and passed to the application program to update the viewpoint of the user.

Future work will extend the interface to include manipulation of objects within the environment. A set of gestures has already been developed for object manipulations such as selection, translation, rotation, resizing and scene zoom. The classification system will be completed, allowing multiple gestures to be recognised. This will provide the user with an interface for navigating and manipulating objects in virtual environments in a new, natural and effective manner.

6. Acknowledgements

This work is supported by The Australian National University Department of Computer Science and Research School for Information Sciences and Engineering, and by the Co-operative Research Centre for Advanced Computational Systems. In particular, thanks goes to Yoshio Matsumoto, Sebastien Rougeaux and Tim Edwards for their technical knowledge and expertise.

7. References

[1] Figueiredo, M., Böhm, K., et al., "Advanced Interaction Techniques in Virtual Environments", *Compute & Graphics,* 17(6) 1993, pp. 655-661.

[2] Foley, J.D., van Dam, A., et al., *Computer Graphics, Principles and Practice*, Addison-Wesley Publishing Company, 1993.

[3] Fröhlich, B., Fischer, M., et al., "Collaborative Production Modeling and Planning", *IEEE Computer Graphics and Applications,* 17(4) 1997.

[4] Bowman, D., Wineman, J., et al., "Designing Animal Habitats within an Immersive VE." *IEEE Computer Graphics and Applications*, 18(5) 1998, pp. 9-13.

[5] Wesche, G., Wind, J., et al., "Visualization on the Responsive Workbench", *IEEE Computer Graphics and Applications,* 17(4) 1997.

[6] Schulz, M., Rending, T., et al., "Analysing Engineering Simulations in a Virtual Environment", *IEEE Computer Graphics and Applications*, 18(6) 1998.

[7] Hubbold, R.J. and McPhater, N.P., "The Use of Virtual Reality for Training Process Plant Operatives", *EPSRC Conference on Virtual Reality and Rapid Prototyping for Engineering,* 1995.

[8] Rosenblum, L., Durbin, J., et al., "Situational Awareness Using the Responsive Workbench", *IEEE Computer Graphics and Applications,* 17(4) 1997.

[9] Tate, D.L., Sibert, L., et al., "Using Virtual Environments to Train Firefighters", *IEEE Computer Graphics and Applications*, 17(6) 1997.

[10] Roussos, M., Johnson, A., et al., "Learning and Building Together in an Immersive Virtual World", *Presence*, 1998.

[11] Hand, C., "A Survey of 3D Interaction Techniques" *Computer Graphics Forum,* 16(5) 1997, pp. 269-281.

[12] Bowman, D. and Hodges, L., "User Interface Constraints for Immersive Virtual Environment Applications", Atlanta, Graphics, Visualization, and Usability Center, Georgia Institute of Technology, 1995.

[13] Bowman, D.A., "Interaction Techniques for Immersive Virtual Environments: Design, Evaluation, and Application", *Invited Talk*, Human Interface Technology Laboratory, University of Washington, 1997.

[14] Bowman, D.A. and Hodges, L.F., "Formalizing the Design, Evaluation, and Application of Interaction Techniques for Immersive Virtual Environments", *The Journal of Visual Languages and Computing*, 1999.

[15] Pavlovic, V.I., Sharma, R., et al., "Gestural Interface to a Visual Computing Environment for Molecular Biologists", *2nd International Conference on Automatic Face and Gesture Recognition*, 1996.

[16] Azarbayejani and Pentland, A., "Real-time self-calibrating stereo person tracking using 3D estimation from blob features", *International Conference on Pattern Recognition*, Vienna, Austria, 1996.

[17] Crowley, Berard, et al., "Finger Tracking as an Input Device for Augmented Reality", *International Workshop on Gesture and Face Recognition*, Zurich, June 1995.

[18] Crowley, J.L. and Coutaz, J., "Vision for Man Machine Interaction", EHCI, Grand Targhee, 1995.

[19] Väänänen and Böhm, "Gesture-driven interaction as a human factor in virtual environments - an approach with neural networks", *Virtual Reality Systems,* Earnshaw, R.A., Gigante, M.A., and Jones, H. (eds), Academic Press Ltd, London.

[20] Pavlovic, V., Sharma, R., et al., "Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review", University of Illinois at Urbana-Champaign, 1995.

[21] Napier, J.R., *The Human Hand*, Carolina Biology Readers, 1976.

[22] Segen, J., "Controlling computers with gloveless gestures", *Virtual Reality Systems,* 1993.

[23] Tsai, R., "An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision", *IEEE Conference on Computer Vision and Pattern Recognition*, Miami Beach, 1986.

[24] Tsai, R., "A versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses", *IEEE Journal of Robotics and Automation,*RA-3(4) 1987.

[25] Rehg, J.M., *Visual Analysis of High DOF Articulated Objects with Application to Hand Tracking*, School of Computer Science, Carnegie Mellon University, 1995.

[26] Clifton, M. and Pang, A., "Cutting Planes and Beyond", *Computers & Graphics,* 21(5) 1997, pp. 563-575.

[27] Becker, D.A., "Sensei: A Real-Time Recognition, Feedback and Training System for T'ai Chi Gestures", *MIT Media Laboratory Technical Report No. 426*, 1997.

[28] Starner, T. and Pentland, A.P., "Real-Time American Sign Language Recognition from Video Using Hidden Markov Models", *MIT Media Laboratory Technical Report No. 375.*

[29] Fels, S., "Neural Networks for Computer-Human Interfaces: Glove-TalkII", *International Conference on Neural Information Processing (ICONIP96)*, Hong Kong, 1996.

[30] Fels, S. and Hinton, G., "Glove-TalkII: A neural network interface which maps gestures to parallel formant speech synthesizer controls", *IEEE Transactions on Neural Networks*, 9(1) 1998, pp. 205-212.

[31] Fels, S. and Hinton, G., "Glove-Talk: A neural network interface between a data-glove and a speech synthesizer" *IEEE Transactions on Neural Networks,* 4(1) 1993.

[32] Böhm, K., Broll,W. et al., "Dynamic Gesture Recognition Using Neural Networks; A Fundament for Advanced Interaction Construction", *IS&T'94: SPIE's Symposium on Imaging Science & Technology*, San Jose, California, USA, 1994.

[33] Salisbury, K., "Making Graphics Physically Tangible", *Communications of the ACM*, 42(8) 1999, pp. 74-81.