A Gesture Driven 3D Interface

MING-CHUN CHANG   LUNGISA MATSHOBA   STEVEN PRESTON

mchang@cs.uct.ac.za   lmatshob@cs.uct.ac.za   spreston@cs.uct.ac.za

_____

3D visualisation systems have a wide variety of applications, particularly as teaching and simulation tools in a broad range of fields. Manipulation of 3D objects using current regular input devices can be tricky and impractical. While various less common interfaces and methods for the manipulation of 3D objects do exist, many of these are expensive and impractical for the common user.

We present a camera-based application where the user can use hand gestures to manipulate and translate objects in a 3D visualisation system in real time. We also provide a highly useable gesture vocabulary by conducting a Wizard of Oz usability experiment. We present a way of interacting with 3D objects by means of a simple and intuitively easy to use interface through the use of hand gestures.

_____

1. INTRODUCTION

Computer Scientists working in the field Human-computer interaction are constantly looking for more useable interfaces for users to interact with computer systems. One such interface that has been identified and only partially explored is the use of hand gestures as a means of input. The most appropriate use of hand gestures, the best gestures to use and the most effective way to capture and detect the gestures are just some of the issues that are still unsolved and have not been thoroughly researched. We present a system for interacting with 3D molecule objects through a simple and easy to use hand gesture interface.

The system captures hand gestures with a single, standard web camera device – the stream of images captured from the web camera is processed using a number of image processing techniques to extract a representation of the hand gesture performed. Gestures are then recognised and classified using a principal component analysis based approach. Finally, the relevant action is performed on the 3D objects according to what gesture has been recognised.

One factor regarding gesture-based interfaces is that no standard exists that states what different gestures mean. Thus, we also present a study into what the most intuitive gestures are for performing particular actions.

2. RELATED WORK

Many papers have been written on the use of the human hand as an interface to a multitude of systems ranging from 3D object manipulation, to typing programs using sign language recognition – or a variant thereof.

What makes our particular solution unique is the focus on evaluating what the most useable gesture is; and the focus on using the system as an interface into large visual modeling environments such as the display of molecules on large screens. Also, our approach proposes a method of capturing and extracting hand gesture features that is

noticeably different to the methods described in related work - such as that done by Y Sato et al in the paper titled 'Real-Time Input of 3D Pose and Gestures of a User's Hand and Its Application for HCI'[14]. In this paper two cameras where used to extract the hand gestures, and the movements of the hands. Other work uses a motion-capture glove for the purpose of capturing hand movements [18], or having the users wear special beads on there hands to aid in the gesture detection.

The majority of human-computer interfaces are driven by the use of a keyboard and mouse. Even though users have learned and grown accustomed to this kind of interface, this method of interaction does not come naturally to humans. It is an acquired skill learned from continuous use. This motivated the research into a hand gesture recognition interface as an alternative to that of the current norm. It is definitely possible to implement a viable interface; however whether it will replace the current norm of using the keyboard and mouse is in question [3].

There are many different methodologies that could be implemented to achieve this. At present the majority of these solutions use solely image processing techniques for the purpose of feature extraction and gestures, none of which present an optimum solution. The basic techniques include:

- Colour segmentation [7],[11]
- Edge-based segmentation
- Region-based segmentation
- Correlation [1],[10]
- Contours [8],[13],[16]
- Blob-models [6]
- Wavelets [17]

Most applications that make use of such an interface are bare-hand game control [2], [15], bare-hand television control [2] and finger driven drawing and mice applications [1], [5], [8], [10], [11]. These applications function under certain restrictions as to simplify the computer vision implementation procedure:

- Background condition restriction [16]
- Light condition restriction
- Hardware requirements [13], [15]
- Initialisation stage [1]
- Hand movement speed restriction [1], [5], [8], [10]

Due to the above mentioned restrictions, most of these applications are not robust enough to be viable as a replacement keyboard and mouse interfaces. This created motivation for the experimental design of our camera-based hand gesture recognition interface.

3. METHOD

The presented system works using a state-based approach – the system can be put into a number of states where only certain hand signals have an affect on the system depending on the state that the system is in. For a specific action to take place, the system needs to be in the correct state. State transitions occur through the performance of some appropriate hand signal.

Four different states are specified – the translation state, rotation state, selection state, and the free state (the initial or base state where no actions on the molecules can be performed). Table i indicates the different signals that are used for the different actions.

Once in a particular state, specific gestures can be performed that carry out actions on the molecules. Thus, for example, once in the translation state, the movement of the flat-hand left and right; up and down; or forward and back would allow for the molecules to be translated along the x, y or z axes, respectively.

HAND SIGNAL		SIGNAL'S PURPOSE
		Change to rotation state and perform rotation.
		Change to translation state and perform translation.
		Select objects.
		Go back one step.

Table i: The hand gestures used in the system.

The system was implemented in four separate phases. The first phase determined the most useable gesture vocabulary for the system; the second phase extracts the feature set from the hand images captured by the web camera; the third phase performs gesture recognition on the extracted features; and the fourth phase performs the visualisation of the actions according to what gesture has been performed. What follows is a description of how each of these phases was implemented.

3.1 Vocabulary Extraction Phase

For any form of communication to occur there needs to be an understandable and predefined vocabulary between the parties of the 'conversation'. In the case of a gesture driven interface this 'conversation' occurs between the human controller and the computer. Vocabulary extraction is the process of extracting a vocabulary from a set of users to be used in the final implementation of a gesture recognition system. For this implementation it was decided that a user centered approach to the creation of a vocabulary would be implemented, and it is hoped that the system is more usable because of this.

Nielsen produced a technical report in 2003 outlining two procedures for the extraction of gestures [9]. The first procedure focused on generating a vocabulary with a focus on technology, and making the gestures as easy for the computer to recognise as possible. This approach was evaluated as being able to produce a vocabulary that was simple for the machine to interpret, but the primary problem with this approach is that the semantics of the gesture and the actual gesture where not linked in such a manner that a user would be able to perform them naturally. The second approach focused on the use of users to create a vocabulary. The extraction methodology that was used for this implementation takes its cue from this second approach.

In order to extract gestures from users a Wizard of Oz experiment was conducted on a series of users. The results from these experiments where then classified and grouped to produce a number of gesture groupings. These gesture groupings where then used to create the final vocabulary for the system.

The final gestures chosen for implementation of the system where then tested in two separate experiments involving ten users each in order to evaluate the vocabulary. The first test was to see if users could understand what the gestures meant through visual inspection. The results from this test where not all together convincing, with certain gestures being easily recognised, while others proved tricky for users to recognise.

The next experiment was to test the memorability of the gestures. Test subjects

where shown a slideshow with instructions on actions to perform. For each incorrect gesture produced a point was deducted, and the vocabulary s memorability has been rated based on the number of points deducted. The vocabulary produced rated a 96% recognition average.

3.2 Image Processing Phase

Gestures are captured by the system through the use of a single web camera facing towards the user. Access to the video input images from the web camera was implemented using the DirectShow component of the DirectX 8 SDK. The video input images were sampled using the SampleGrabber class of DirectShow.

3.2.1 Colour Space Transformation

The RGB colour space does not provide sufficient information about skin objects in the image to significantly differentiate it from other objects and background. The colour space also has the issue of the luminance and chrominance properties not been separated. This makes it more difficult to distinguish skin from other skin-colour like objects.

A transformation is performed on the RGB values by applying a transform matrix onto the image set. The transformation changes the representation of the image to an YCbCr colour space.

Below is an illustration of the matrix transform.

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

There exist a correlation between the Cb and Cr component that uniquely differentiates a skin colour from other parts of the image, it also works on different types of skin colour.

3.2.2 Segmentation

The transformation to the YCbCr colour space provided the possibility of thresholding to locate the skin colour object in an image. An upper and lower bound for the Cb and Cr component was used to locate the desire skin colour pixels. The range decided upon was broad for the purpose of identifying all skin colours and to ensure no loss of any skin colour details. The Y component threshold was then applied to enhance the segmentation further by using the intensity properties of the image. Threshold segmentation was implemented as the first step to decrease the details in the image set greatly for efficient processing.

The values that were found to be skin colour from thresholding are then further divided into three separate groups. The values are assigned to groups depending on their pixel value. Pixel values within a group are all assigned the same colour. The three colours used were red, green and yellow. These are used to minimize the amount of detail received from the input image. The abundance of information is not necessary for the segmentation procedure to follow.

The colour assignment into three groups makes the erosion and dilation operations implemented more effective. The erosion operations are applied first followed by dilation. The procedure is a morphological opening operation that removes small unnecessary objects from the image.

For the erosion operation any pixel value with a neighbouring pixel value that is not of the same colour is changed to a grey value and considered to be a false pixel. The dilation operation then checks for all

false pixels in the image - if all the neighbouring pixels of a false pixel are the same value, the false pixel is then assigned to be true and given the colour of the neighbouring pixels. Afterwards any remaining false pixels are assigned to be true and changed to background colour.

Figure 1 shows an image sequence throughout the segmentation process, starting from YCbCr on the left, colour assignment in the middle and opening operation on the right.
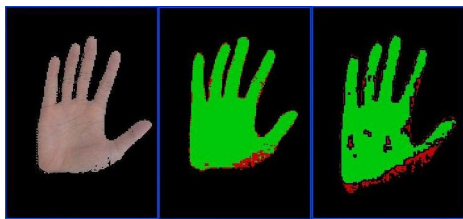

Fig 1: Left: threshold using YCbCr colour space. Middle: Colour Assignment to the different colour regions. Right: Morphological opening operation to eliminate small pixels.

3.2.3 Marching Squares

To find the boundary values of the hand, the marching squares algorithm has been implemented. The first step was to divide the image set to blocks of squares of dimension two pixels by two pixels. An iterative procedure of traversing through all the corner points of all the squares and calculating there weight was then implemented. The weight of a corner point is calculated based on the amount of colour pixels within the square blocks that its part of. Each time a colour pixel was detected, the weight of the corner point would be incremented by one. Once the weighting for all corner points have been found, corner points with a weighting of greater than zero are set to one. The value one is used to indicate that the corner point is part of the object. Any corner point with no weighting is set to zero indicating that the corner point belongs to the background. The result is a binary image of only values of one or zero. This makes it easier to process and determine the cases.

The method used to determine the case and draw the lines was implemented as follows, every single block are checked iteratively until a case has been found. The properties inherent in knowing the case that was found were used to locate the position of the next case. Thus a process of locating sequences of cases occurs until a complete object outline is drawn. The method then returns to the procedure of checking for a case match from where it left off. The above process is performed until every object has been found in the scene. Lines belonging to the same object are drawn with the same colour and outlines of different objects are given a different colour.

Once a block has been checked, it is marked to be false by given it the value of negative one. By doing so guarantees that the boundary lines of an object will not be drawn infinitely, because every single square block can only be checked once. Once all the objects have been found, a procedure of eliminating all objects other than the largest object occurs.

Figure 2 shows an image sequence throughout the marching square process. Left is the square grid representation, middle shows all the objects in the scene, and right shows the largest object in the scene.
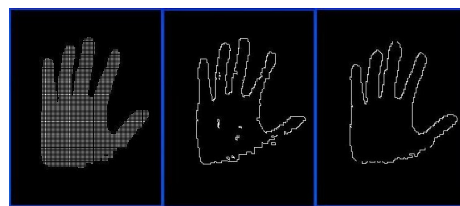

Fig 2: An image sequence throughout the marching square process. Left: the square grid representation. Middle: all the objects in the scene. Right: the largest object in the scene.

3.3 Gesture Recognition Phase

Once the captured image has been processed and the features of the hand extracted, gesture recognition can take place.

PCA has traditionally been described as suitable for static pattern recognition problems, but difficult to implement for dynamic pattern recognition [6]. This suggests that PCA is suitable for static gesture recognition but not for moving gestures. Dynamic gesture recognition is thus performed through the use of a combination of PCA and image processing techniques by breaking the gesture recognition down into the following two phases:

- A gesture can be defined as a series of postures where each frame of a gesture captured by the web camera is analysed and treated as separate posture. PCA is used to determine if this is the correct posture and determines what gesture is being performed and for how long the gesture is being performed.

- When the PCA has determined that a correct posture is being performed for a gesture to be carried out, orientation and movement detail of the hand is determined by analysing pixel intensity changes in the series of frames that the neural network and PCA deem to be part of the gesture.

In [12], Turk and Pentland describe a method where PCA is used for the purpose of face recognition. This algorithm is very similar to the traditional PCA algorithm – it does, however, include an important step in the training phase that allows eigenvectors of a much smaller matrix than the covariance matrix of the input data to be calculated. The PCA based system for posture recognition was based largely on their algorithm. What follows is a description of the adapted method used for implementing the PCA based approach to posture recognition.

Suppose we are working with images of size N x N pixels where M images are used for training. Suppose also that $A = [\Phi_1\ \Phi_2\ .\ .\ .\ \Phi_m]$ is the matrix of all normalised vector representations Φ_i of the M training images, and that C is the covariance matrix derived from A. Turk and Pentland suggest that we can solve for the N^2–dimensional eigenvectors of C by first solving for the eigenvectors of an M x M matrix and then by taking appropriate linear combinations of the hand images to form the eigenhands.

First they deduce that $C = AA^T$, and then go to consider the eigenvectors $\mathbf{v}_i$ of AA^T such that

$$A^TA\ \mathbf{v}_i\ =\ \mu_i\ \mathbf{v}_i,$$

where μ_i is the ith eigenvalue of AA^T.

This implies that

$$AA^TA\ \mathbf{v}_i\ =\ \mu_i A\mathbf{v}_i$$

from which it can bee seen that $A\mathbf{v}_i$ are the eigenvectors of $C = AA^T$. Now, if we construct the M x M matrix $L = A^TA$ and find the M eigenvectors, $\mathbf{v}_l$, of L, then with these eigenvectors we are able to determine linear combinations of the M training hand images to form the eigenhands, or principal components, $\mathbf{u}_l$.

$$\mathbf{u}_l\ =\ \sum_{k=1}^{M} \mathbf{v}_{lk}\ \Phi_k,$$

where $l = 1, \ldots, M.$ (1)

In (1) above, note that the eigenvectors of L, are ordered according to increasing value of the associated eigenvalues. Figure 3

gives eight of the eigenhands calculated from a training data set of 48 hand posture images.
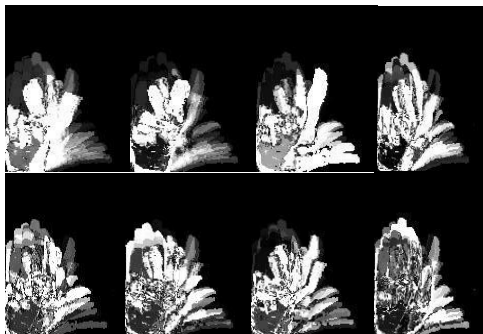

Fig 3: Eight of the eigenhands calculated from a training dataset of 48 images.

The eigenhand images calculated from the eigenvectors of L span a basis set of with which to describe hand images. Let Γ be a new hand image that we wish to classify. To classify Γ, it first needs to be transformed into its eigenhand components:

$$\omega_k = \mathbf{u}_k (\Gamma - \Psi),$$

for $k = 1, \ldots, M$ and where Ψ is the mean hand image calculated from training set of hand images.

These eigenhand components form a set of weights that together form the vector $\Omega = [\omega_1, \omega_2, \ldots, \omega_m]$ that describes the contribution of each principal component in representing Γ. This vector is what we are now able to use to find which of the training images the new image Γ resembles the closest. This is done by calculating the Euclidean distance between Ω and Ω_k, where Ω_k is the vector of weights that describes the kth hand image in the initial training set:

$$\epsilon_k = \|\Omega - \Omega_k\|.$$

If ϵ_k is the smallest element of $\{\epsilon_1, \epsilon_2, \ldots, \epsilon_m\}$ then Γ closest resembles the kth hand image in the initial training set. Finally, we need to check that the smallest ϵ_k is above a certain threshold value Θ to ensure that postures that do not resemble those in the training set enough are discarded.

3.4 3D Front-end Visualisation Phase

In order to make the system more than just a trivial test system a useful front-end visualisation needed to be chosen. To this end molecular visualisation was chosen. The main reason for this choice was that though at present a number of systems exist for representing molecular data in 3D, all these systems use a mouse-based interface to interact with 3D data. This results in complicated interfaces to convert this 2D input into 3D operations, and often programs have a learning curve that users need to overcome. As such molecular biologists could benefit from a gesture recognition driven molecular viewer to be used as both a teaching and research tool.

The visualisation system was created using Opengl, and is capable of opening molecule files stored in the .pdb file format.

4. RESULTS

The final implemented system is able to run in real time on a standard desktop PC and is able to detect all four of the gestures given in table i. The system does, however, have a number of constraints:

- Lighting plays a vital role in how effectively the hand's features are extracted from the web camera feed. If the light is not at the right intensity and not diffuse enough, feature detection is not successful enough for gesture classification to take place. Also, once the light has been suitably initialised, the range of skin

colour that it detects is extremely narrow. Thus, only hands of a very similar colour are detectable by the system.

- The system runs with on average a 1 second lag. This is due to the fact that together, the image processing and gesture recognition phases are not able to process sufficient frames per second. Thus, although the system does run in real time, the algorithms used need to be further streamlined to make the system more useable.

- While the system does not require a single colour background behind the hand, it is preferable for no other visual noise with a similar intensity to skin colour to occur behind the hand. This noise decreases the system's ability to extract hand gesture features.

5. CONCLUSION

A gesture recognition interface for 3D applications using only a single 2D camera as input can be created, and made to be relatively accurate. Furthermore, by following a proper vocabulary extraction methodology this system can be created such that it is intuitive and easy for users to learn to use.

Skin colour detection using only the YCbCr colour space produces reasonable results, but in order to increase the detection rate there is a need for a better method of skin colour extraction in order to segment the hand out of the image under all lighting conditions.

PCA is a suitable method for the purpose of gesture recognition, provided that it is used in combination with various image processing techniques. Hands must be orientated and scaled appropriately for PCA to be used effectively.

We are able to deduce that a useable hand gesture recognition system is possible to implement where gestures are captured by only a single web camera.

6. REFERENCES

[1] CROWLEY, J., BÉRARD, F., AND COUTAZ, J., 1995, Finger tacking as an input device for augmented reality, Automatic Face and Gesture Recognition, Zürich, 195-200.

[2] FREEMAN, W., ANDERSON, D. AND BEARDSLEY, P., 1998, Computer Vision for Interactive Computer Graphics, IEEE Computer Graphics and Applications, 42-53.

[3] F. C. M. KJELDSEN, 1997, Visual Interpretation of Hand Gestures as a Practical Interface Modality.

[4] M. HAJDINJAK AND F. MILHELIČ, Wizard of Oz Experiments" University of Ljubljana, Faculty of Electrical Engineering.

[5] LAPTEV, I. AND LINDEBERG, 2000, T. Tracking of Multi-State Hand Models Using Particle Filtering and a Hierarchy of Multi-Scale Image Features, Technical report ISRN KTH/NA/P-00/ 12-SE, September 2000.

[6] J.J. LA VIOLA, 1999, "A Survey of Hand Posture and Gesture Recognition Techniques and Technology", Technical Report: CS-99-11.

[7] LIEN, C. AND HUANG, 1998, C. Model-Based Articulated Hand Motion Tracking For Gesture Recognition, Image and Vision Computing, vol. 16, no. 2, 121-134, February 1998.

[8] MACCORMICK, J.M. AND ISARD, 2000, M. Partitioned sampling, articulated objects, and interface-quality hand tracking, European Conference on Computer Vision, Dublin, 2000.

[9] M. NIELSEN, M. STÖRRING, T. MOESLUND AND E. GRANUM, 2003, A Procedure for Developing Intuitive and Ergonomic Gesture Interfaces for Man-Machine Interaction, Technical Report, Aalborg University.

[10] O'HAGAN, R. AND ZELINSKY, 1997, A. Finger Track - A Robust and Real-Time Gesture Interface, Australian Joint Conference on Artificial Intelligence, Perth.

[11] QUEK, F., MYSLIWIEC, T. AND ZHAO, M. FINGER, 1995, A freehand pointing interface, International Workshop on Automatic Face- and Gesture-Recognition, Zürich.

[12] M.A. TURK AND A.P. PENTLAND, 1991, Eigenfaces for Recognition , Journal of Cognitive Neuroscience, 1991.

[13] REHG, J. AND KANADE, T. DIGITEYES, 1993, Vision-based human hand tracking, Technical Report CMU-CS-93-220, School of Computer Science, Carnegie Mellon University, 1993.

[14] SATO, Y., SAITO, M., KOIK, H., 2001, Real-Time Input of 3D Pose and Gestures of a User's Hand and Its Applications for HCI. IEEE Virtual Reality Conference 2001 (VR'01) p. 79, 2001

[15] SATO, Y., KOBAYASHI, Y. AND KOIKE, H. FAST, 2000, Tracking of Hands and Fingertips in Infrared Images for Augmented Desk Interface, International Conference on Automatic Face and Gesture Recognition, Grenoble,.

[16] SEGEN, J., 1998, GestureVR: Vision-Based 3D Hand Interface for Spatial Interaction, ACM Multimedia Conference, Bristol, 1998.

[17] TRIESCH, J. AND MALSBURG, C., 1996, Robust Classification of Hand Postures against Complex Background, International Conference on Automatic Face and Gesture Recognition, Killington, 1996.

[18] ZIMMERMAN, T. G., LANIER, J., BLANCHARD, C., BRYSON, S., HARVILL, Y., 1987, A Hand Gesture Interface Device. ACM SIGCHI Bulletin , Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface, Volume 17 Issue SI, 1987.