Geometric Modelling in a Three-Dimensional Environment

G J Clapworthy, R A Noble
Department of Computer & Information Sciences,
De Montfort University
Hammerwood Gate
Kents Hill
Milton Keynes, MK7 6HP
email: gclapwor@dmu.ac.uk

Abstract

Computer animators have long had difficulties in defining and controlling the shape of deformable objects during motion. In the context of human figure animation, a new form of surface description has been developed, Murugaiyan & Clapworthy [1], which allows the surface to be manipulated by the adjustment of a single parameter. The form of the surface deformations produced by the operations is under the control of the user. The mathematical structures used are extensions of NURBS, but have a greater flexibility at the cost of marginally-increased overheads.

The development of this tool enables the surface modelling of a deformable object to be speeded up considerably, and the paper describes a geometric modeller currently under construction which will use this facility to produce the desired object by 'virtual sculpting', a process analogous to the production of real objects from clay. In this, the user is able to pull and push the surface shape to produce either rounded or sharp surface indentations or protrusions by direct interaction with the surface.

The object is viewed stereoscopically and at present shutter glasses are used. These have the attendant problem of ghosting, but this can be removed by a subtract-from-background algorithm, Noble [2]. Ongoing work at De Montfort University [3, 4] will eventually allow enhancements using autostereoscopic displays and eye tracking.

The object is oriented in three dimensions using a spaceball and the surface operations are controlled using a mouse. The reduction in the number of parameters employed in the surface description enables the operations on the surface to be performed by use of the mouse buttons. Thus, the shape of the object is defined by simple 'point and click' operations providing an extremely rapid, interactive method of surface construction.

It is anticipated that this modelling system could be used in a number of areas of work, including Virtual Reality.

While the tool is still at the developmental stage, the paper lays out the complete conceptual basis for its construction and implementation.

1. Introduction

For a long time, computer animators have found it difficult to define the shape of computer-generated deformable objects. Equally, it is difficult to control them whilst they are being reshaped. The final stage of animating the objects, in which they move from one defined shape to another presents further problems.

While large-scale deformations can be created by the use of established techniques such as Free-Form Deformations (FFDs) [5], tools which give the animator detailed control of the shape of a deformable object throughout a deformation are not widely available. In the context of key-frame animation, defining in detail the shape at each key frame may not provide good control of the shape at the in-between frames.

Free-form objects are often modelled using surfaces which are parameterised by a two parameter system. These surfaces are defined by an array of points, known as "control points", which influence where the surface lies. The simple topologically-rectangular description which is often used is limited in the sorts of shapes it can generate. Cylinders can be generated, but not forking ones. Surface holes

may result which are difficult to fill. Varying the surface shape in a controlled manner is not very easy.

Conventionally, the variation of shape is created by movement of control points. However, unless care is taken to ensure that this manipulation is performed similarly at each key frame, inconsistencies in the underlying structure can produce unforeseen effects at the in-between frames.

A computer system for animating objects should therefore provide several facilities. Firstly, the user must be able easily to describe to the computer the changes required. Secondly, the system must be able to provide changes of the type desired, preferably changing limited areas of the object at a time. Thirdly, there must be some underlying consistency between different shapes of the object, which allows the one to be deformed into the other for animation.

Given such an animation system, the user will interact with it through the Human Computer Interface, rotating objects, selecting options and describing the required changes. Traditionally, this interface has included a mouse, a keyboard and a 2-D screen. These are not the ideal means of interfacing to a 3-D system, whilst trying to interpret a complex 3-D surface from a series of 2-D views requires experience.

There is a need, therefore, for a sculpting tool which is quick, flexible, easy to use and which maintains consistency of the object between consecutive poses. This paper discusses the design of such a modelling system, which forms the basis of an on-going research project. It is designed to run on a computer graphics workstation using a mouse and a 6 degree of freedom (dof) input device, which in our case is a SpaceBall. To allow easy visualisation of the object being modelled it has a stereoscopic viewing option to display the objects in 3-D.

It is considered that such a tool has considerable potential for use in Virtual Reality applications, where deformable objects such as human organs, which require considerable detail and well-defined deformations, are required.

2. Background

The whole purpose of the sculpting tool is that it should be easy and intuitive to use, require only 2 hands for its operation and not require detailed mathematical understanding on the part of the user. We therefore propose the use of the following:

-the stereoscopic (3-D) viewing system for easy interpretation of the model.
-a SpaceBall 6-degree-of-freedom controller to rotate and move the model.
-a simple "point and click" metaphor using a mouse to interface to the surface.

For users not used to reading multiple-view engineering drawings, it is not easy to understand a complex 3-D object from a series of 2-D views. Using a stereoscopic display allows the user to see a natural 3-D object as in normal viewing.

The SpaceBall will allow the selected object, a group of objects or the whole world to be translated or rotated, allowing the user to view objects from different angles and to access the part of an object that requires changing. The changes can either be global ones, to generate the correct general shape, or local ones to adjust particular detail. The user may switch between global and local control by pointing the mouse at appropriate icons.

The "point and click" metaphor, which is described in detail later, allows the user to push and pull the surface simply by pointing the mouse at the part of the surface at which changes are required and pressing one of the mouse buttons. These changes affect only small regions of the object and in this way local detail can be built in. To explain this fully, we must first outline the mathematics of the surface being used.

2.1 The Mathematics of the Surface

Flexible, deformable surfaces are often modelled using B-Splines, or their more general form, NURBS (Non Uniform Rational B-Spline). Such a surface is defined by selecting a set of points in 3-D known as "control points". These points control the surface in two ways. First, their position defines the surface, although the surface does not generally pass through them. If the user moves a control point,

the surface moves in response. This movement is predictable. It is also local, in the sense that only the part of the surface close to the control point changes.

The surface may also be changed by altering the "weight" of a control point. This weight is a measure of the attraction between the point and the surface. The greater the weight, the more the surface is pulled towards it. Reducing the weight will cause the surface to move away from the control point. This allows the user to shape the surface local to a control point by adjusting its weight. This localisation occurs because a control point influences only a restricted part of the surface.

These techniques can be applied to the modelling of limb behaviour, where they lead to special conditions. The work by Murugaiyan & Clapworthy [1] was aimed at producing a method of modelling and adjusting limb cross sections, such that the limb shape would change in a realistic way as its joints bent. In this application it is important that the control points are defined by the system and are not moved by the user, as discrepancies in the underlying structure between joint positions can create unpredictable distortions in the surface shape during motion. By ensuring that the positions of the control points are consistent for different angles of the joint, predictable behaviour of the surface shape is obtained as the joint bends.

The types of changes being modelled were skin protuberance and depressions caused by muscles flexing and relaxing and skin protrusions and depressions caused by bones and joints. These changes must be modelled whilst maintaining the basic cylindrical shape of the limb. Given the requirement that the user should not move the control points, the use of NURBS to define the surface did not allow the inclusion of the required amount of surface detail, because of their Convex Hull property, which keeps the surface within the volume enclosed by the local control points. To counteract this, a special extension of the NURBS surface known as Contours in a Spline Surface (CISS) was developed which allowed local detail to be included by varying only the weight parameters.

The CISS surface has a mathematical structure similar to a NURBS surface and can thus be varied by changing the weights of the associated control points. Each CISS curve effectively defines a cross section of the limb, which can then be shaped by weight variations.

To understand how the CISS works, consider the 2-D version in Fig. 1. This shows a planar array of control points P_{ut} in a rectangular mesh, with lines of constant parameter t highlighted. The figure is drawn in physical space (i.e. x, y space), but since the weights are all set to 1.0 and the control points are all on a physically rectangular grid, the lines of constant t or u must be parallel to the x or y axes.

The control points shown in Fig. 1 are doubled along the bottom and top. This means:

$$P_{u0} = P_{u1}$$

$$P_{u2} = P_{u3}$$

The contours, shown bold, are for different values of t = constant. Assuming the physical spacing between the points are equal, the contours start and end as shown.

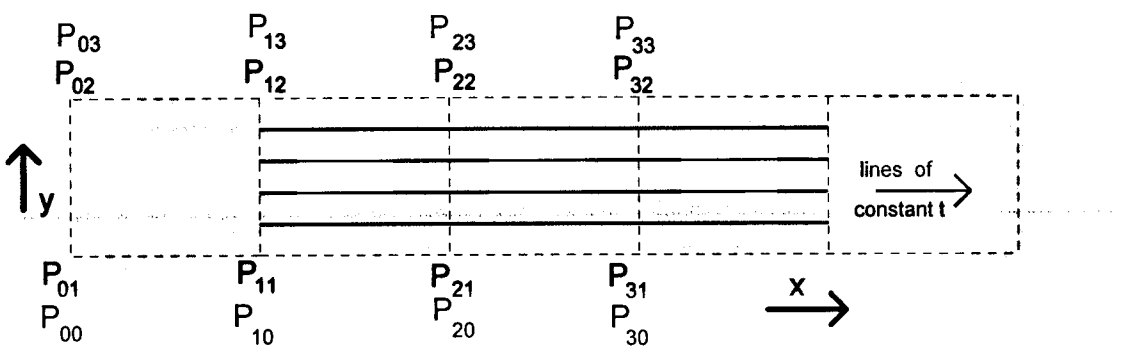

Fig. 1 Doubled row of control points and constant contours

Suppose now that the control points are arranged circularly as in Fig. 2, the control points still being doubled. The contours of constant t are now circles, provided that all of the control point weights are the same. Note that this figure is still shown in physical space. In parametric space, the points lie on a rectangular (u, t) grid.

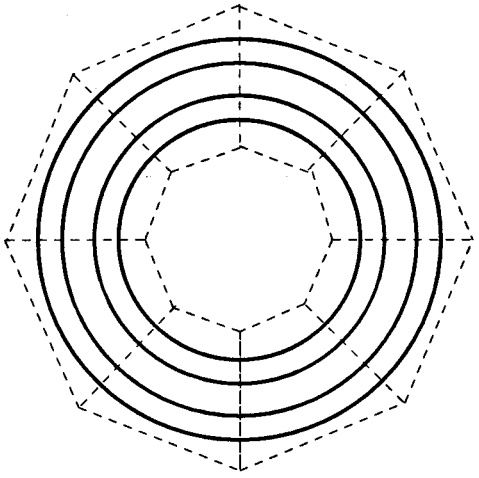

Fig. 2 Lines of constant parameter for a double ring of control points

Since these are NURBS curves, they posses the Convex Hull property (for positive weights), that is, the curves must lie within the Convex Hull of the control points, which is approximately the area marked by the inner and outer dotted polygons. Thus, the curves lie between the inner and outer dotted polygons. As the weights of P_{u1} or P_{u2} vary, the curves move towards the inner or outer polygon.

Experiments in varying the shape of this contour showed that local control by adjustment of the weight parameters produced the sorts of shapes required for limb modelling (Fig. 3 and 4) when the midpoint contour of t was used. Generally, pushing by lowering a weight value produces a rounded change in the shape of the contour (hence in Fig. 3 the inner ring weight is reduced) while pulling towards a point by increasing the weight creates a more peaked effect (so in Fig. 4 the outer weight is increased).

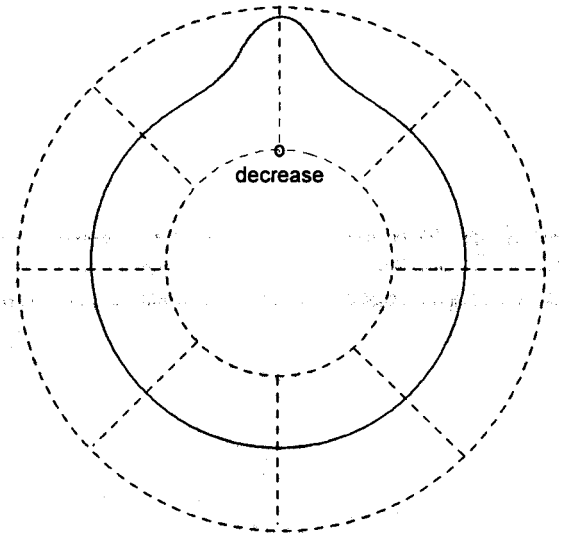

Fig. 3 A smooth bump

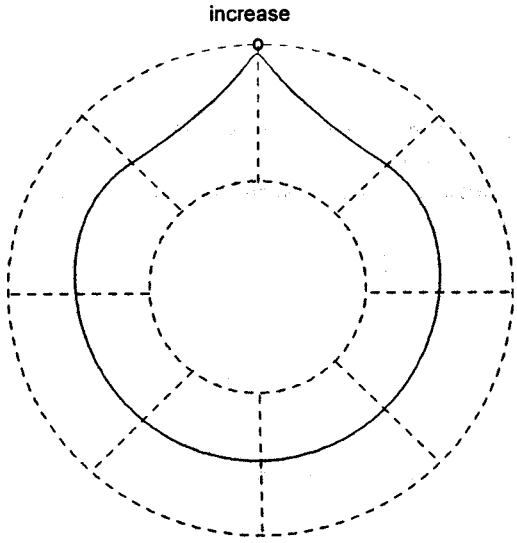

Fig. 4 A sharp bump

To build a surface, a series of CISSes must be put together and the common t = constant surface used. Mathematically this works as described in Appendix 1, which shows that the CISS surface is equivalent to a NURBS surface (equation 1), but that it allows the user to shape the cross section in an easy and meaningful way. The extra overheads introduced by using CISSes rather than NURBS are considered in Appendix 2.

The form of NURBS surface described by equation 1 is known as the "tensor product" form. This is easy to use, but has shortcomings. For example, when modelling the human body, several such surfaces must be joined together. At the point where the legs meet the trunk, gaps will be left. This has been addressed, Savva [6], in the context of CISSes.

3. Techniques for Interacting with the Surface

Having outlined how the surfaces are described in terms of control points, we can now consider how the user interacts with them. Starting from an initial shape, the user may wish to make global changes to produce the general form of the required object. The user then needs to effect local manipulation and add local detail. We also discuss the stereoscopic interface by which the user sees the object and consider a few broader issues.

3.1 Changing the Surface Globally

The CISS surface can be sculpted by changing the weight of a single control point. This is good for shaping the surface locally. There are times however when the user wishes to make more global changes to the surface. This will be achieved in two ways. Firstly, by applying global functions such as scale, twist and shear. Secondly by the method of Free Form Deformations (FFDs), Sederberg & Parry [5]. These consist of embedding the surface in a 3-D Bézier volume which can be manipulated in the usual way, by moving the Bézier control points. As the Bézier volume deforms, so does the surface embedded in it. Other possibilities are discussed in Section 5.1.

3.2 The "Point and Click" Metaphor

The user does not wish to keep switching hands between different devices. Modelling will therefore use only a mouse and SpaceBall, one dedicated to each hand. "Virtual Sculpting" will then be achieved using a "point and click" metaphor. The basic idea is that as the user clicks the left or right mouse button, the weight of the selected control point (as described below) is changed and hence the surface is sculpted.

The SpaceBall is used to orient the object being manipulated, to give access to the localised area at which detailed modelling will take place. The operations on the surface are then performed as required by direct use of the mouse at the appropriate position.

For this, the point on the surface at which the mouse is pointing is found and the nearest control point in either the inner or outer control surface (see below) is identified. To identify the point at which the mouse is pointing we proceed as follows.

Firstly, being a stereoscopic system, the depth of the mouse-controlled cursor must be known, as well as its horizontal and vertical position. Since the cursor is only a 2-D object (and only actually occurs in one eye's view) it has no defined depth. Since however it is always drawn in front of all the objects and since the objects are cut at the near clipping plane, any parts in front of the plane not being displayed, the cursor is defined to be on the near clipping plane. This, combined with its screen co-ordinates, defines its position in 3-D model space, allowing the ray from the user's eye to be defined.

Secondly, the intersection of the ray and the surface must be found. Since the surface is a function of 2 parameters (u and v, Appendix 1), two sets of curves can be generated on the surface as u = constant and v = constant. These curves intersect in a topologically rectangular fashion and allow a set of triangles to be defined from which the surface can be drawn. The intersection of the ray from the eye can now be found with these triangles and the closest one taken as an approximation of the required surface point.

Since the surface is drawn using the triangles, the derived intersection is the correct intersection with the object as seen, although this is actually an approximation to the mathematical surface. A closer approximation can be generated by using a finer mesh, but will necessarily take longer to compute.

In this "point and click" mode of operation the three mouse buttons have the following functionality.

Left: Increment weight of selected control point.
Middle: Toggle select inner / outer control point.
Right: Decrement weight of selected control point.

The four possible operations of increase or decrease the weight of the inner or outer control point result in the four required types of change. These are a smooth or sharp bump (as in Fig. 3 and 4) or a smooth or sharp indentation.

In this way, the object can be oriented speedily and, with a few rapid clicks of the mouse buttons at appropriate positions, the required surface manipulations can easily be performed.

An alternative configuration using two mice is also being investigated, but this will require a slightly amended form of interface to allow for the reduction in the flexibility of the device used for the orientation of the object.

3.3 Issues Relating to the Stereoscopic Interface

Properly designed stereoscopic viewing systems provide a very attractive environment for viewing complex 3-D objects. There are, however, various potential pitfalls which need to be avoided if an acceptable system is to be achieved. These relate both to the technology and also to the implementation.

Consider the following points which we discuss in turn:

1. The depth scale of the image must be designed to match its width scale.
2. The depth of field of the 3-D image must be within the user's viewing tolerance.
3. The meaning of pan and zoom in a 3-D context must be carefully considered.
4. A switched shutter system is prone to producing ghost images.
5. A stereo image of an object with no surface texture is very hard to interpret.

The depth scale of a stereo image depends upon the separation of the left and right images and is thus independent of the horizontal scale of the pictures. It is also generally non-linear. This latter point is most readily appreciated by realising that any point in the picture for which the separation is equal to the viewer's eye separation will appear to be at infinity. Furthermore, the depth scale is dependent upon the distance of the viewer from the VDU and the system should therefore be designed with an ideal viewer distance in mind.

The depth of field of a stereo image is the 3-D display equivalent to the front and back clipping planes used in a 2-D display, 3-D modelling system. The picture as seen will extend in front and behind the screen from the near to the far clipping plane. However, the human vision system can only fuse two images into one solid image if the separation between the images is within specific limits. Thus, the picture depth must be restricted if the viewer is to use the system comfortably. This means the user will only be able to see the stereo image if the clipping planes are close enough to the screen. We return to this issue in Section 5.2.

Zooming essentially means reducing the angle of view of the camera, giving the impression of moving towards the object in view. This is different from actually moving the camera towards the object. The latter moves the point of convergence (zero disparity on the screen) and the clipping planes. The former, however, magnifies the objects in view, including the depth of the picture. Care must be taken to adjust the clipping planes to keep the depth seen by the viewer within limits. In effect, the clipping planes are now defined in viewer space and not model space.

Ghosting is caused by the right image reaching the left eye and vice-versa. The viewer sees the required object with ghost images either side. For a switched shutter system it is generally caused by the fact that the screen phosphor is still glowing when the alternate shutter opens. The effect can be very distracting to the user. One method of reducing the problem is to introduce a low intensity grey background and to subtract the ghost from it. This has been demonstrated, but is very specific, not only to the VDU used but also to the vertical position of the ghost on the screen. This latter is simply because the picture is written from the top of the screen downwards, so the ghost at the top is less bright.

Stereo works because the eye can relate positions on one image to those on the other. In a flat shaded image, with no variations, the stereo effect reduces, or fails. This can be shown by a simple experiment. Draw a vertical cylinder, shaded by a light at infinity, behind the viewer. The light intensity varies horizontally across it and its depth can be clearly seen. Now rotate it to become a horizontal cylinder, across the screen. There is now no horizontal variation in intensity and its depth can no longer be estimated. It is therefore important that the shading varies horizontally across the surface.

Having set up a stereo viewing environment, we will need to investigate how users adapt to it. Natural methods of interaction in a stereoscopic environment then need to be investigated. It should be particularly noted that the normal mouse cursor is only a 2-D display and when the workstation is in stereo mode, will only appear in one eye.

3.4 Further User Requirements

The following typical operations will be required by the user:

1. Choosing one or more objects for manipulation.
2. Rotating the world or rotating object(s).
3. Selecting combinations of SpaceBall drives (enable pan, zoom, rotate etc.)
4. Selecting control points for manipulation
5. Moving a control point or vary its weight.
6. Selecting the surface deformation method

Various methods of performing such operations will be investigated to find the most suitable approach in the current context.

4. System Development

The proposed system raises questions in the areas of user interface and surface modelling. The next sub-sections will consider a number of these.

4.1 The 6 dof Interface

An object which can move in 3-D and rotate about 3 independent axes has 6 degrees of freedom (dof). Trying to control all 6 dof simultaneously is not easy. Until the general availability of the SpaceBall and related devices, the user was more or less bound to control a maximum of 3 dof with each hand, simply because 6-dof interfaces were not available. Even the 3-dof joysticks which were available were not easy to use. The vertical motion presented a problem. If the joystick responded to vertical movement, the weight of the user's hand caused drift. If it responded to twist, it made it very difficult to combine horizontal moves with vertical ones.

The SpaceBall has overcome the problems of the previous 3-dof joysticks to an extent. All translates are equivalent (being a push in that direction), the ball has no preferred direction of movement and the user rests his hand on the support, reducing vertical drift. There is still however some cross-talk between the drives, particularly if rotate and translate are enabled together.

These problems need to be addressed in the context of model handling. Some specific areas which need to be addressed can be listed as follows:

- *Which combinations of drives should be enabled
- *Methods of allowing the user to switch easily between combinations
- *Methods of displaying to the user the current set of enabled drives
- *How to implement more advanced methods of control, such as enabling a rotation about a specific line in space

Another issue to note is that when rotating the whole of model space, the user may wish to perform the rotation relative to himself. When rotating a single object he may wish to rotate relative to the object's axis (e.g. rotating a wheel about its axis).

4.2 The Stereo Display

Some of the problems associated with stereo displays were mentioned in Section 3.1. Ongoing work at De Montfort University into using autostereoscopic displays will overcome some of these difficulties, particularly the ghosting caused by crosstalk.

Others will be overcome by suitable understanding of the user requirements and careful design. In particular, the stereo controls will need to be designed to keep the display within the user's ability to see it comfortably.

An important part of the development cycle is the evaluation of the system by potential users. Particular factors are how acceptable the stereo is to the user, how easily the stereo pictures can be interpreted, how accurately the depth dimension can be used and whether it can in fact replace the traditional 2-D views.

4.3 Extending the CISS Concept

The CISS structure was developed and used specifically for limb modelling, where it proved to be most useful. In this context, the control points initially lay on a circle as in Fig. 2 and the surface was shaped from that. Later, some experiments were made with ellipses and similar shapes. In the modeller being developed, the shapes of the control polygons will be related much more closely to the shape of the final object. Because it is constrained by the Convex Hull property, there is no reason to expect that the surface will exhibit anything other than reasonable behaviour in the new configuration.

5. Future Development Issues

5.1 Manipulating the Surface

Gross variations to the object may be made by manipulating groups of control points using FFD techniques. This is useful during the initial modelling stage when the general shape of an object is being defined.

For CISSes and NURBS, the basic action is to move a control point or to change its weight. This is useful for local control and adding detail. Broader local changes can be applied by manipulating a group of control points together. Such methods need to be investigated.

One possibility is to manipulate an area of the surface by adjusting multiple control points. This movement could be weighted to change the outlying ones less than the central ones. Changes can also be effected by changing the underlying parametric knots, although the results of such changes are not easily predicted.

To increase detail on the surface, local refinement can be used to increase the number of control points. A mechanism would need to be defined whereby the user indicates the requirement for increased detail and the system responds. Given that the user is not expected to know about control points and NURBS theory, this requires an interface which translates the user's sculpting requirement into a mathematical surface-deformation function.

5.2 Improving the Stereo Interface

A badly-designed stereo interface can cause the user to suffer eye strain. The underlying reason in assumed here to be the fact that the real image lies on the VDU screen, whilst the perceived image is off the screen. The conflict between converging the eyes off the screen whilst focusing on the screen causes strain. Two possible ways of reducing this conflict are considered here, focal blurring with picture depth and eye tracking.

There are well known techniques for generating computer pictures that have a finite depth of field. In such pictures, only objects in the defined focal plane are truly in focus. Objects in front or behind this plane blur out. This is how our eyes see the world, since they have a finite depth of focus. Objects not in the plane of convergence of the eyes (that is, in front or behind the object we are looking at) blur out and are not fused into a single image. We are so used to this that we rarely notice it.

In a computer generated stereo picture however, the whole image is on the screen and therefore the whole image is in focus. In effect, the stereo image has unlimited depth of field. We suggest that if this depth is greater than the eye's depth of field, eye strain will result. An obvious cure therefore is to use the known techniques to add focal depth to the stereo picture. Unfortunately, this is a very computationally intensive process and cannot be done in a sensible time on our current workstation.

A second method of keeping the image (i.e. the screen) within the focal range of the eye is to move the part of the image the user is looking at on to the screen. This implies the computer can tell where the user is looking and for this we need to use eye tracking.

Eye tracking is currently becoming a reality. Systems that track one eye already exist and systems to track both eyes simultaneously are being researched. The following possibility therefore presents itself. Suppose a two-eye tracking system had developed to the point where it was accurate, non-intrusive to the user, had a fast enough update rate and was not too expensive. At the user's request, the software could move the point he was looking at to the plane of the screen. Providing that the movement was graceful he would be able to follow it. The resulting system would enable the user to keep the point of interest close to the screen, causing less eye strain.

The above eye tracking system also suggests a means of pan and rotate control, by which the user looks at an object and the program moves it to the screen centre. Tilting the head could allow rotations.

6. Conclusion

This paper has outlined a research project in which we will develop a deformable object modelling system, taking advantage of useful properties of CISSes, which are an extension to conventional NURBS. The method of operation of the system and its user interface have been described in detail and further issues which will have to be resolved once it is operational have been indicated.

References

1. E Murugaiyan, G J Clapworthy *Using CISSes for Detailed Modelling of Cylinders.* Proc GraphiCon '93, St Petersburg (Russia), Sept 1993

2. R A Noble *Removing Ghosts from the Silicon Graphics Workstation when Displaying in Stereo.* RN94-4, De Montfort University, Milton Keynes.

3. I Sexton *Parallax Barrier Displays.* IEE Colloquium on Stereoscopic Television, London, Oct. 1992

4. T. Bardsley & I. Sexton *Task Performance using 3-D Displays.* IFIP TC WG 5.10 Workshop on Virtual Environments, Coimbra (Portugal), 1994.

5. TW Sederberg, SR Parry *Free-form Deformation of Solid Geometric Models.* ACM Computer Graphics, Proc. SIGGRAPH '86, Vol. 20, No. 4, pp. 151-160, 1986.

6. A. Savva (Private Communication).

Appendix 1 Mathematical Formulation of a CISS Surface

Non Uniform Rational B-Splines (NURBS) came about by adding an extra dimension to normal B-Splines and projecting the resulting 4-D structure back into 3-D. A 2-parameter surface can then be represented as in equation (1), with "w" being, in effect, the fourth co-ordinate. This value w can then be interpreted as a "weight" in the 3-D representation, allowing the surface to be manipulated without the control points being moved.

In a similar way, a CISS surface is defined by constructing a particular form of NURBS volume (that is, a function of the 3 parameters t, u and v, as in equation 2) and converting back to a surface by setting the parameter t equal to a constant, as in equation 3. This again allows the surface to be manipulated by varying the weights without moving the control points. In this case however, the variations produced are constrained by the original CISS rings to suit the modelling purposes.

Given a set of CISS weights and control points defining a surface (equation 3), a set of NURBS weights and control points can be found (equations 4 and 5) which define the same surface. However, the CISS definition cannot be then forgotten and the NURBS weights manipulated directly, since changing a CISS weight moves some of the equivalent NURBS control points (equation 5). Thus, the CISS form has to be maintained and equations 4 and 5 tell us the associated overheads for using this more complex form of description. Provided the CISS is properly defined, this overhead need not be excessive (Appendix 2).

Let us now consider the mathematics. The standard form of a NURBS surface generated from a 2-D mesh P_{vu} is given by:

$$\mathbf{Q}(v,u) = \frac{\sum_v \sum_u \mathbf{P}_{vu}\, w_{vu}.B_v^k(v).B_u^k(u)}{\sum_v \sum_u w_{vu}.B_v^k(v).B_u^k(u)} \qquad (1)$$

This can be extended to a 3-D volume generated by a 3-D mesh $\mathbf{P}_{vut}$ as:

$$\mathbf{Q}(v,u,t) = \frac{\sum_{v=0}^{N_v-1}\sum_{u=0}^{N_u-1}\sum_{t=0}^{N_t-1}\mathbf{P}_{vut}.w_{vut}.B_t^k(t).B_u^k(u).B_v^k(v)}{\sum_{v=0}^{N_v-1}\sum_{u=0}^{N_u-1}\sum_{t=0}^{N_t-1}w_{vut}.B_t^k(t).B_u^k(u).B_v^k(v)} \tag{2}$$

where:

v, u, t as subscripts are indices
(v), (u), (t) are function parameterisations
$\mathbf{P}_{vut}$ is a set of 3-D control points
$B_t^k(t)$ is a k^{th} order B-Spline
w_{vut} is a set of weights associated with the control points
N_t is the number of control points in variable t
$\mathbf{Q}$(v, u, t) is a vector function of the parameters v, u and t.

By putting parameter $t = t_0$, a constant value, a CISS surface embedded in the 3-D volume is generated. As there is usually only one patch in the radial direction, $N_t = k$ and the range of t in the CISS volume will be between 0 and 1, making a choice of $t_0 = 0.5$ the most convenient.

Given that t is a constant, equation (2) can be transformed into the usual form of a 2-D NURBS surface, (1) as follows. Considering (2) with $t = t_0$:

$$\mathbf{Q}(v,u) = \frac{\sum_{v}\sum_{u}\left(\sum_{t=0}^{k-1}\mathbf{P}_{vut}.w_{vut}.B_t^k(t_0)\right)B_u^k(u).B_v^k(v)}{\sum_{v}\sum_{u}\left(\sum_{t=0}^{k-1}w_{vut}.B_t^k(t_0)\right)B_u^k(u).B_v^k(v)} \tag{3}$$

The terms in brackets are now constants. Now, define a new set of points and weights as follows, noting that the new $\overline{\mathbf{P}}$ and $\overline{w}$ are different from the original $\mathbf{P}$ and w.

$$\overline{w}_{vu} = \sum_{t=0}^{k-1} w_{vut}.B_t^k(t_0) \tag{4}$$

$$\overline{\mathbf{P}}_{vu} = \frac{\sum_{t=0}^{k-1}\mathbf{P}_{vut}.w_{vut}.B_t^k(t_0)}{\overline{w}_{vu}} \tag{5}$$

The resulting equation is in the form of a NURBS surface in 2 parameters u and v. Thus, the CISS surface is equivalent to the NURBS surface generated from (4), (5).

Appendix 2 Overheads of Using CISSes Rather than NURBS

In the context of the proposed system, an object may be rotated without being changed, may be changed by having one of the CISS weights altered, or may be changed by having new parts added to it. We consider the three cases separately. For the sake of clarity we assume that cubic Splines (k = 4) are being used.

In the first case the equivalent NURBS values (equations 4 and 5) are already known and there is no overhead.

Whilst a user is using the system for modelling, he will be changing weights by clicking the mouse buttons. For each button click the surface needs to be redrawn to show the effect. This response needs to be fast for the system to be acceptable. In the second case then, when weights are being altered, overheads must be low.

Equations 4 says that the NURBS weight $\overline{w}_{vu}$ depends only on the four CISS weights w_{vut} which lie on the radial line of the CISS ring (Fig. 1 and 2). Thus, varying a CISS weight only changes one NURBS weight, so equation 4 must be evaluated once.

Equation 5 shows that each NURBS point is a weighted sum of the 4 CISS points along a radial line of the CISS ring. The weights here are functions of the CISS weights on that radial line. Since the CISS points do not move, only changes to the weights will affect the NURBS points. Changing a CISS weight only changes one NURBS point, so equation 5 need only be evaluated for one point. Note that it is a vector equation with 3 components.

The overhead in the second case then is that for each button click to change a weight, equation 4 and 5 are evaluated once each.

In the third case, where the object is being built up and new rings are being added, the NURBS points for the whole ring must be calculated. This requires multiple applications of equations 4 and 5.

Consider now the evaluation of equations 4 and 5. By design the four points are in two pairs, with the first two and the last two on a radial CISS line being equal. Furthermore, only the second and third weight can be changed from unity. Further simplification arises from the fact that the value of t_0 and the knots are constant. This allows the factors $B_t^k(t_0)$ to be precalculated. As a result the required calculation is very straight forward.