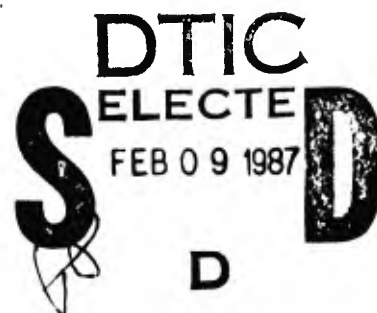


AD-A176 537

Free-Form Deformation as a General
Purpose Interactive Modeling Tool



A Thesis
Presented to the
Department of Computer Science
Brigham Young University



In Partial Fulfillment
of the Requirements for the Degree
Master of Science

DTIC FILE COPY

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

by
Rodney O. Davis
December 1986

87 2 6 056

FREE-FORM DEFORMATION AS A GENERAL
PURPOSE INTERACTIVE MODELING TOOL

Rodney O. Davis

Department of Computer Science

M.S. Degree, December 1986

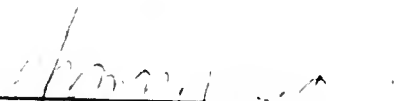
ABSTRACT

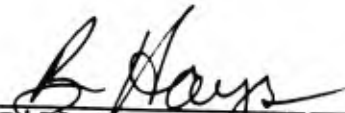
The geometric modeling technique known as Free-form Deformation (FFD) was conceived by Sederberg and set forth in a Ph.D. dissertation by Parry. Parry's dissertation deals with the feasibility of using FFD to accomplish free-form sculpting in a non-interactive constructive solid geometry modeling system and does not address any high-level operations.

FFD can be used as a general purpose technique for manipulating geometric models in a "standard" interactive 3-D world coordinate space, i.e. most high-level modeling operations can be accomplished using FFD as the interactive manipulative medium. The rational basis function is explored along with free hand and algorithmic techniques for single and grouped control point manipulation. Operations from the viewpoint of the user of an interactive graphics environment are identified. High-level operations are demonstrated. The characteristic of model continuity is explored.

COMMITTEE APPROVAL:

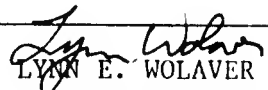

Robert P. Burton, Committee Chairman


Thomas W. Sederberg, Committee Member


Bill Hays, Department Chairman

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/CI/NR 87-13T	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Free-Form Deformation as a General Purpose Interactive Modeling Tool		5. TYPE OF REPORT & PERIOD COVERED THESIS/DISSERTATION
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Rodney O. Davis		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: Brigham Young Univ		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS AFIT/NR WPAFB OH 45433-6583		12. REPORT DATE 1986
		13. NUMBER OF PAGES 52
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES APPROVED FOR PUBLIC RELEASE: IAW AFR 190-1  LYNN E. WOLAVER 14 JAN 89 Dean for Research and Professional Development AFIT/NR		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) ATTACHED		

This thesis, by Rodney O. Davis is accepted in its present form by the Department of Computer Science of Brigham Young University as satisfying the thesis requirement for the degree of Master of Science.

RP Burton

Robert P. Burton, Committee Chairman

Thomas W. Sederberg

Thomas W. Sederberg, Committee Member

Nov 26, 1986
Date

B Hays

Bill Hays, Department Chairman



Accession For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

TABLE OF CONTENTS

	Page
List of Figures	iv
Chapter	
1. Introduction	1
Free-form Deformation	1
Computer Graphics Concepts	1
Mathematical Concepts	4
Context of this Research	6
2. Free-form Deformation Application	8
Basic Concepts	8
Initial FFD Region Definition	9
Arbitrary Size & Orientation	10
Object-tied Size & Arbitrary Orientation	11
Object-tied Size & Orientation	12
Stored FFDs	13
Predefined Objects	13
Predefined FFD Regions	16
Simple Deformation Concepts	17
Free-hand Sculpting	17
Group Manipulation	20
Planar Groupings	21
Logical Groupings	22

	Groupings of Symmetry	23
	Rigid Body Motions and Regular Deformations	24
	Scaling, Rotation & Translation	25
	Tapering	25
	Bending	27
	Twisting	34
	Continuity Control and Manipulation	38
	Assuring Order of Continuity	38
	Creating Specific Discontinuities	41
3.	Analysis and Summary	42
	Basic Concepts	42
	The FFD Region	43
	Simple Manipulations	45
	Rigid Body Motions & Regular Deformations	46
	Continuity	48
4.	Conclusion and Future Directions	49
	Conclusion	49
	Future Directions	49
	User Interface	50
	Specific Applications	51
	Hardware Architectures	51

FIGURES

Figure	Page
1. Initial FFD Configuration	3
2. Application of a Stored FFD	15
3. Example of Control Point Weight Variation	19
4. Orientation of FFD Region for Deriving a Bend Deformation	29
5. Control Point Placement for Deriving Cir- cular Arcs with a Quadratic Basis Function .	31
6. Control Point Placement for Deriving Cir- cular Arcs with a Cubic Basis Function . .	32
7. The Steps of the Bend Process	35
8. Orientation of the FFD Region for Deriving a Twist Deformation	37
9. Configuration of a Twisting FFD	39
10. Example of Differing Control Point Influences	44
11. Attempt at Inverting the Twist	47

ACKNOWLEDGMENTS

I gratefully acknowledge the help of Dr. Thomas Sederberg for his aid in teaching me the finer points of basis function manipulation and in helping me to formulate some of the equations used in this thesis. Dr. Robert P. Burton is acknowledged as a mentor who aided me in formulating the problem space of this research.

I acknowledge the sacrifice of my wife and children who gave up valuable time with husband and daddy. Without their understanding I would not have been able to concentrate my time and mental efforts on this research.

I very gratefully acknowledge the help of the Lord in aiding me from day to day in overcoming problems and sharpening my mental faculties.

CHAPTER 1

Introduction

This thesis explores the versatility of Free-form Deformation (FFD) as a general purpose geometric modeling tool in an interactive 3-D graphics environment. FFD is a geometric model manipulation method conceived by Dr. Thomas Sederberg (1986) and set forth in a Ph.D. Dissertation by Scott Parry (1986). This thesis explores the general purpose interactive modeling capabilities of FFD using Parry's work as a starting point. The concepts explored by this research are of a general nature and are not tied to any specific application.

Free-form Deformation

The solids manipulation technique known as Free-form Deformation (FFD) was conceived by Dr. Thomas Sederberg in 1985. The technique is a new approach to modeling solid objects bounded by free form surfaces.

Computer Graphics Concepts

FFD is defined within a sub-space of the world coordinate space. This sub-space is bounded by a set of six clipping planes forming a parallelepiped assumed to be

rectangular. This sub-space may be oriented in any position in the world coordinate space and has its own coordinate axes, referred to as the STU axes. Four world coordinate points are required to identify the box, as follows:

P_0 at $(s_{\min}, t_{\min}, u_{\min})_{STU}$

P_1 at $(s_{\max}, t_{\min}, u_{\min})_{STU}$

P_2 at $(s_{\min}, t_{\max}, u_{\min})_{STU}$

P_3 at $(s_{\min}, t_{\min}, u_{\max})_{STU}$

The minimum and maximum values are dependent upon the actual basis function used.

A lattice of control points is defined within the parallelepiped by independently and uniformly dividing each of the S, T & U axes. Each division defines a plane of constant value in the axis it divides. Control points are established at each intersection of three perpendicular planes. This provides the initial configuration of any FFD. One such configuration is shown in figure 1. The control point density is defined by the values of l, m & n, each being one less than the number of control points along the s, t & u axes, respectively. Hereafter, the density values l, m & n often are denoted as $l \times m \times n$.

Moving any of the control points deforms, or alters, the coordinates according to some predefined tri-variate basis function. Movement of control points affects each coordinate within the STU box in a predictable manner. The

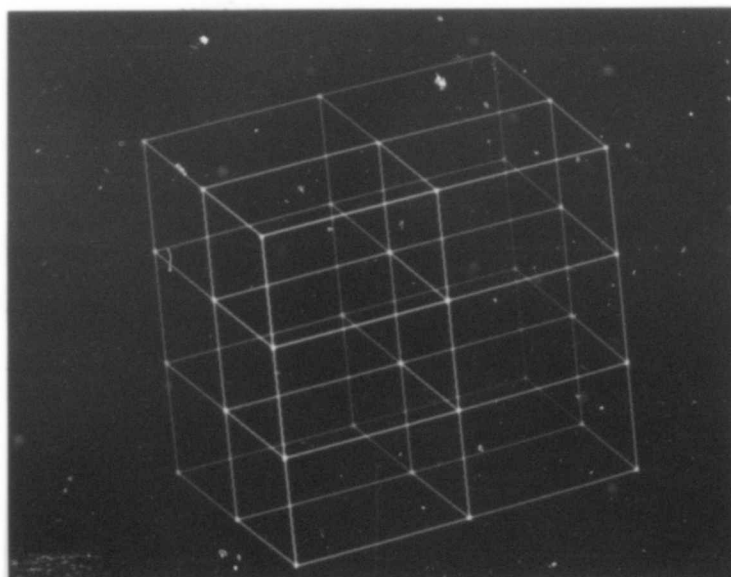


Figure 1.

Initial FFD Configuration

This is the initial configuration of a $2 \times 3 \times 2$ FFD. Intersections of any two dividing planes are designated by yellow lines. Intersections of any three dividing planes designated by green dots.

identity configuration is defined as having the control points in the initial configuration.

Usually, FFD is defined to alter only the coordinates within its parallelepiped. In this case, point clipping must be applied to all of the coordinates in world coordinate space. Any point which has at least one of its s , t or u values outside the range of the identified minimum and maximum values does not undergo FFD. No matter how the configuration of the set of control points changes, clipping is still performed against the original six clipping planes. In other words, once the clipping decision has been made for a given coordinate value, that decision holds for the "life" of the FFD.

This method of deforming, or altering, the world coordinates within a specific sub-space in a predictable manner provides a powerful and versatile geometric modeling tool. In his dissertation, Parry (1986) demonstrates the feasibility of this tool in a constructive solid geometry modeling system. This thesis explores the utility of FFD as a tool for performing higher level operations that relieve the user of the tedium of moving each control point.

Mathematical Concepts

FFD is a mapping from R^3 to R^3 as follows. First, the original coordinate X (coordinate at the time of FFD definition) is converted to the STU coordinate system by

solving the following vector equation for s , t & u (see Parry 1986):

$$X = P_0 + sS + tT + uU$$

Here, P_0 is the world coordinate point corresponding to the origin of the STU axes and S , T & U are the vectors which comprise these axes. The variables s , t & u are the corresponding STU coordinate values for the original point X and are found by applying Cramer's rule to the above equation (see Parry 1986).

The coordinates are mapped to their deformed location using a rational trivariate tensor product Bernstein polynomial:

$$\frac{\sum_{ijk} w_{ijk} P_{ijk} B_i^l(s) B_j^m(t) B_k^n(u)}{\sum_{ijk} w_{ijk} B_i^l(s) B_j^m(t) B_k^n(u)}$$

$$(0 \leq i \leq l, 0 \leq j \leq m, 0 \leq k \leq n)$$

Where $B_i^l(s)$ is the univariate Bernstein polynomial $\binom{l}{i} (1-s)^{l-i} s^i$ and likewise B_j^m & B_k^n . To increase computation speed, the Bernstein polynomials are converted to standard power basis polynomials which can be evaluated using Horner's algorithm.

The FFD which uses a rational basis function, allowing weights to be assigned to control points, was not explored by Parry.

Context of This Research

A simple 3-D interactive polygonal based modeling system has been developed for use in this research. This system is capable of providing four different basic graphical objects: spheres, cylinders, boxes and pre-defined MOVIE.BYU geometry files. The use of parametric surfaces and the combining of simple geometric forms via Boolean operations are covered in Parry (1986); their inclusion in this system is not essential to proving the thesis of this research. The system also includes a cursor with movement in either the XY or XZ planes. All objects, including the cursor, are depth cued to aid in depth perception.

Graphical objects are considered to be specific instances of graphical groups or items similar to those discussed by Newman & Sproull (1978). In this representation scheme, objects are defined in their own coordinate system and are placed in the world coordinate system by a specific instance transformation, represented by the matrix I_i .

As with Parry's research, the basis function used is the Bernstein basis. No necessary relationship exists between the validity of these concepts and the basis

function selected. As with Parry's research, the Bernstein basis is chosen for simplicity.

The role of the software implementation is to test and exercise the concepts. The intent of this research is not to develop a comprehensive interactive graphics environment based upon the FFD, but rather to derive and implement basic functional entities which can be included in a comprehensive system.

CHAPTER 2

Free-form Deformation Application

The objective of this research is to enhance the utility and versatility of FFD. Several basic concepts are identified and several methods of applying FFD are explored.

Basic Concepts

Since the FFD region has its own STU axes, it can be viewed as a graphical item with the STU axes as the defining coordinate axes. Using this notion, the region can be placed into the world space by a FFD instance transformation. Since every graphics transformation matrix has an inverse, the STU axes can be normalized to, and aligned with, the XYZ axes using the inverse instance transformation. This facilitates easy manipulation of the FFD control points. Once the STU axes are normalized to and aligned with the XYZ axes, the application of transformation matrices can be applied to manipulate selected control points. When manipulation is complete, the STU axes can be returned to their originally defined position via the instance transformation.

The origin of the STU axes is in the lower-left-rear corner for a right-handed coordinate system and in the lower-left-front corner for a left-handed coordinate system. The use of a STU coordinate systems which is opposite-handed from the world coordinate system has the effect of scaling the U axis by -1.0. For ease of manipulation it is helpful to align the STU axes with the world coordinate axes as an intermediate step. For the intermediate step in some operations it is useful to place the STU axes coincident with the world coordinate axes. For the intermediate step in other operations it is useful to align the parallelepiped with the world coordinate axes, placing the geometric center of at least one parallelepiped face coincident with one of the of the world coordinate axes.

Where practical, transformation matrices are used to define manipulation of FFD control points; this is also how control point manipulations are implemented.

Initial FFD Region Definition

Two sets of information are needed to define a FFD region: 1) the bounding points P_0 , P_1 , P_2 & P_3 , and 2) the control point densities l , m & n . Since the initial configuration is the identity configuration, all weights are assigned the value of 1.0. The assigned values in each set are dependent upon the type of operation being performed.

There are several methods for determining the bounding points P_0 through P_3 in an interactive environment.

Except in the special case of stored FFDs, to be discussed later, the initial FFD region is always a box with uniform weights of 1.0. The box and the region of the work space which it contains may be defined in at least three ways. The selection of a specific FFD region defining method depends on the goal of the particular FFD application. Three different methods are used for defining a FFD region: 1) arbitrary size & orientation, 2) object-tied size & arbitrary orientation and 3) object-tied size and orientation.

Arbitrary Size & Orientation

A box is defined in world coordinates by specifying two diagonally opposite vertices, P_{min} and P_{max} . The two points are translated by the transformation matrix T_t so that P_{min} lies at the origin. The matrix R , which represents the desired rotation, is also determined. The matrix R^{-1} is applied to P_{max} , yielding P_{max}' . The point P_{max}' is one end of a diagonal from the origin. This diagonal defines a box aligned with the XYZ axes which is the parallelepiped defining the FFD region. The previously identified values P_0 , P_1 , P_2 and P_3 are defined as follows: P_0 corresponding to $(0,0,0)$, P_1 corresponding to $(box_x_{max}, 0, 0)$, P_2 corresponding to $(0, box_y_{max}, 0)$ and P_3 corresponding to $(0, 0,$

$\text{box_z}_{\text{max}})$. These four values are then transformed by the composition RT_i^{-1} , the FFD instance transformation.

A capability allowed by the groups & items concept, which must be facilitated by the user interface, is the exclusion of non-selected groups and/or items. Though a group or item may exist within a FFD region, deselecting it causes its coordinate values to be excluded from the deformation. This adds the additional FFD region clipping criterion of coordinate selection. Exclusion can also be applied to either of the other two methods for defining a FFD region.

Object-tied Size & Arbitrary Orientation

A FFD region is defined to be tied to an object if it uses the coordinates and/or defining axes of that object in determining its four defining coordinates, i.e. P_0 through P_3 . A region which is tied to a modeled object's size is a parallelepiped bounding the object.

The transformation matrix T_i is derived which translates the origin of the object's defining axes to the origin of world coordinates and a rotational transformation R is defined to orient the STU axes in the world coordinate space. The defining points of the object are translated by the composition $T_i R^{-1}$ and the maximum and minimum values are determined. These maximum and minimum values are used to define a box aligned with the XYZ axes. The four bounding

coordinates, P_0 through P_3 , which are derived from the box and then transformed by the composition RT_t^{-1} .

A variation on the object-tied size concept is the local FFD, a concept identified by Parry (1986). In this case, only a specified part of the modeled object is in the FFD region. The local area is defined by specifying floor and ceiling points which specify a box oriented in the same manner as the parallelepiped. The minimum and maximum searches for determining the bounding parallelepiped will not go beyond this box. Before they are used, these floor and ceiling coordinates must be transformed by the composition $T_t R^{-1}$ along with the objects coordinates. The local FFD can also be applied to the third method of FFD region definition, object-tied size & orientation.

Object-tied Size & Orientation

This method of FFD region definition aligns the STU axes with an objects local defining xyz axes. Since it is also tied to size, it is the minimum size necessary to completely enclose the object.

To define a region which is tied to both object size and orientation, the points which define the object are transformed by the inverse of the objects instance transformation, I_t^{-1} , and the maximum and minimum values are found. These maximum and minimum values are used to define a box aligned with the XYZ axes from which the defining points P_0

through P_3 are derived. These defining points are then transformed by the instance transformation I_t . A local FFD is calculated in a similar manner, with the floor and ceiling values transformed by the matrix I_t^{-1} prior to use.

Once the parallelepiped has been defined the remaining FFD descriptive information is associated to it, i.e. control point densities, control point locations, etc.. This descriptive information may follow the previously described initial configuration, or it may be associated with a stored FFD. A stored FFD is the saved set of descriptive information associated with an FFD which will allow it to be associated with a new parallelepiped. Associating this information with a new parallelepiped is the process of restoring an FFD.

Stored FFDs

A stored FFD is the set of FFD descriptive information which has at least one control point out of its original lattice position and/or which has at least one control point with a weight other than 1.0. When the FFD descriptive information is associated with a new parallelepiped, every vertex within the new parallelepiped is deformed in accordance with the stored FFD.

Predefined Objects. If the stored FFD is used in a system with predefined primitives, i.e. cylinders, spheres, boxes, etc., it can be used to define new objects. These

new objects can be added to the list of available predefined objects. The FFD in figure 2a deforms a sphere into an egg shape, as shown in figure 2b, by associating it with the new parallelepiped which encompasses the sphere.

There is no difficulty in associating stored FFD control points with a new parallelepiped when the control points are saved in STU coordinates because the STU coordinates always range from a constant minimum to a constant maximum. The inverse instance transformation is applied and the STU coordinates of the stored FFD are scaled to match the size of the new parallelepiped. Any size difference which is not a uniform scaling in all three STU axes will introduce distortion into the output shape.

The original FFD region is defined -- in an object-tied size and orientation manner -- about some predefined object. The control points are manipulated to derive the model shape and the FFD is stored. In order to store a FFD to be used in defining new objects, the following values need to be saved: 1) control point densities l , m & n , 2) control points, in STU coordinates and 3) control point weights.

The FFD is restored by the following process: First, a FFD region is created about some predefined object, tied to its size and orientation, and translated by I_t^{-1} so as to coincide with the XYZ axes. This region is given the control point densities -- l , m & n values -- of the stored FFD. Second, the length of each of the STU axes is

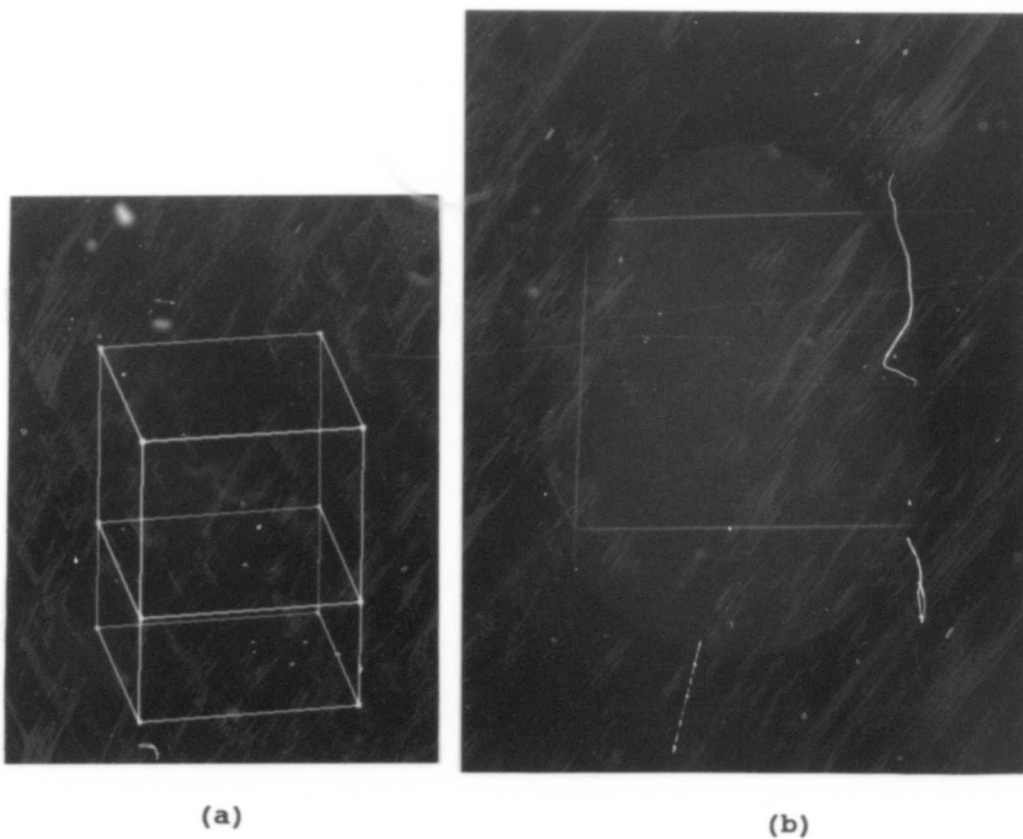


Figure 2

Application of a Stored FFD

When the stored FFD in photograph (a) is placed around a predefined sphere, the result is the egg shape of photograph (b).

calculated in world coordinate units. Third, the control points are restored and multiplied by the calculated S, T & U lengths. This works because each of the STU axes ranges from 0.0 to 1.0 with the Bernstein polynomial basis function. Fourth the control points are transformed by the new FFD's instance transformation. Fifth, and last, the stored weights are assigned to their corresponding control points. The deformation is calculated in the standard manner.

Predefined FFD Regions. In addition to predefined objects, stored FFDs have an additional use. A stored FFD region can be recalled and a modeled object moved within it to provide the desired deformation. This is useful in the case where the intent is to deform an object so that it matches a previously deformed object. In such a case, the object selection method of clipping to the FFD region might be necessary, depending upon the actual system implementation.

The original FFD region is defined -- in an object-tied size and orientation manner -- about some predefined object. The control points are manipulated to derive the model shape and the FFD is stored. In order to store a FFD for this purpose, the following values need to be stored: 1) the four values P_0 through P_1 , 2) the control point densities l , m & n , 3) the control points, in world coordinates, 4) the control point weights and 5) the FFD instance transformation.

The FFD is restored by the following process. First, a FFD region is created using the stored parameters P_0 through P_3 and the stored control point densities. Second, the control points of this newly created FFD are assigned the stored coordinate values and weights. Third, and last, the FFD instance transformation is assigned the stored instance transformation. Since this creates a FFD region in its original position, any objects which were originally defined by this FFD are doubly deformed. To prevent double deformation, the previously deformed objects must be deselected.

Simple Deformation Concepts

The most basic, and simplest, types of deformation are: 1) free-hand sculpting and 2) group manipulation. As demonstrated by Parry (1986), when using the FFD for simple manipulations, it is helpful to have the control point lattice visible.

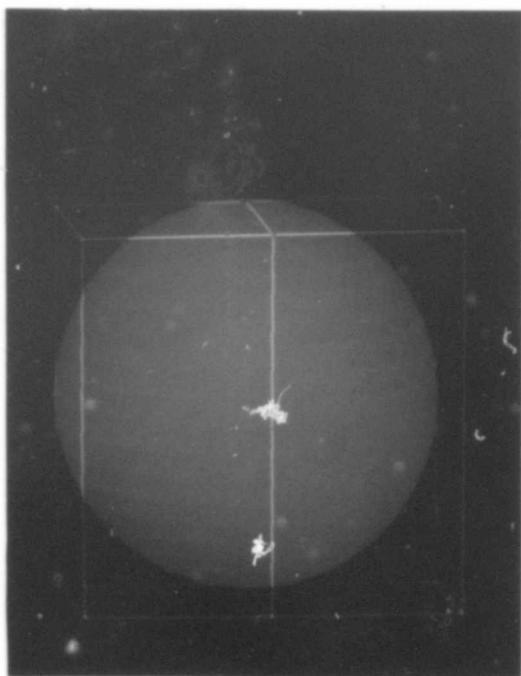
Free-hand Sculpting

Parry's research achieved FFD by sculpting with individual control point movement in a non-interactive environment. This is the simplest method and easy to accomplish in an interactive environment. The desired control point is selected with the interactive locator device and placed in a new position with that same device. This is the most general FFD. Parry used this method in a non-interactive

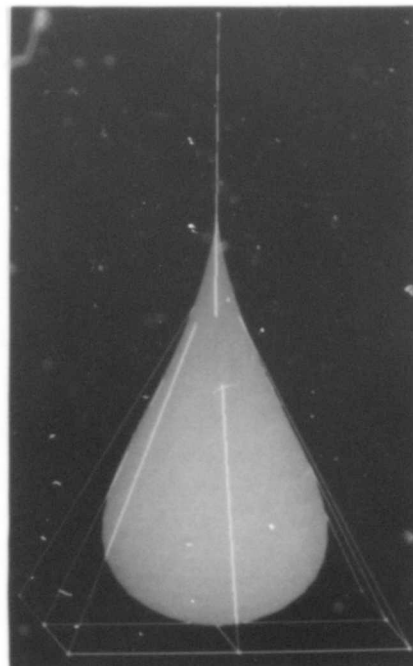
environment to design a telephone handset with "... only a few hours of experience [and] ... in a single design iteration!"

Each control point exerts some degree of influence on the deformation, depending on its relative position, its assigned weight and the basis function used. In the case of uniform weights, the greater the density of control points (the values of l , m and n above) the less the effect on the model in altering a single control point. On the other hand, the lower the density of control points, the lower the amount of curvature which the deformation can create. Experience appears to be the determining factor in selecting the control point densities.

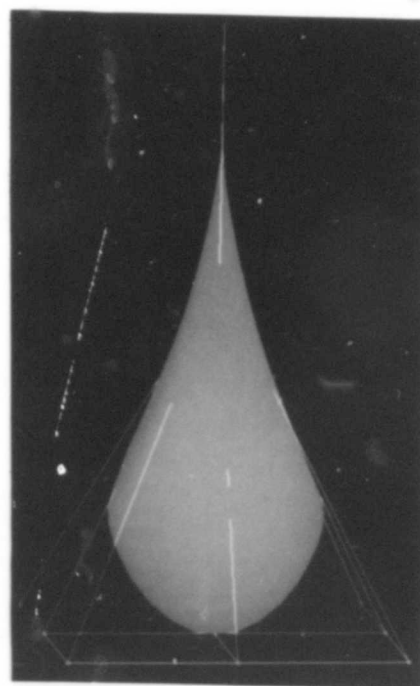
Control point weight provides another means of sculpting the shape of an object which was not explored by Parry (1986). Increasing the weight of a control point increases its influence on the deformation. This is demonstrated by the example in figure 3. Figure 3b shows a teardrop shape created from the sphere and FFD of figure 3a. The deformation is made by moving the center point up, along a line parallel to the t axis, and moving all of the adjacent edge control points into the center control point's previous position. The control points in the lattice of figures 3a & 3b all have weights of 1.0. By increasing the weight of the raised center control point, the teardrop is stretched toward it as shown in figure 3c.



(a)



(b)



(c)

Figure 3

Example of Control
Point Weight Variation

The sphere in photograph (a) is made into a teardrop by control point movement, shown in photograph (b). The teardrop is altered by varying the weight of the control point in red, shown in photograph (c).

Group Manipulation

The grouping of control points provides a basic building block for accomplishing higher level manipulations. A group is a set of control points which all move when any one of the set is moved.

Group manipulation requires the identification of a functional relationship between control points. A strict functional grouping identifies which control points are affected, and by how much, when a single control point is moved. For example, a control point can be placed into the group of all control points which moves by $f(x,y,z) \rightarrow (x',y',z')$ when a specific control point moves. One possible functional relationship $f(x,y,z)$ is $x'=x+2dx'$, $y'=y+dy'$ and $z'=z+.3dz'$ (where dx' , dy' & dz' represent the difference between the original position of the specified control point and its new position). The inverse relationship -- that is, now that control point B is declared to move when control point A moves, control point A can be declared to move when control point B moves -- may be declared by forming a group on the second control point which includes the first, but it is not necessary and depends upon the application. The actual function used may also form subgroups, i.e. some of the control points in a group move by $f(x,y,z)$ while others move by $g(x,y,z)$.

In the preceding example, the differences in x , y & z are specified as dx' , dy' and dz' because the functional

grouping is calculated in world coordinate units, but is relative to the STU axes orientation. The control point values must be transformed by I_t^{-1} before the difference values can be calculated. Once the control points are inversely transformed, the difference values dx' , dy' & dz' are calculated and applied to the control points according to the groups functional specification. Once the control points have been functionally altered, they are transformed by I_t and the deformation is calculated.

This scheme allows for a single control point to belong to several groups, allowing the user to specify if a relationship always holds. If the user desires to have control point A always move when control point B moves, then control point A must be included in every group which contains control point B. This can be identified in a matrix with a column and a row for each control point. When the control point associated with a column is moved, the functions in that column are applied to the control points corresponding to the rows of the function entries.

There are three basic types of groupings: 1) planar groupings, 2) logical groupings and 3) groupings of symmetry.

Planar Groupings. In an initially defined FFD region, the control points lie on sets of planes which are perpendicular to the S, T & U axes. These planes can be manipulated by manipulating the contained control points as a

single group. The intersection of these planes with the modeled objects within the FFD region can be viewed as cross sections. These cross sections are affected by every control point in the FFD lattice, not just those which lie on the plane.

Many functions can be performed upon the points of a plane. The plane can be moved in the 3-D world coordinate space, adding a constant (dx', dy', dz') to all the control points which lie on the plane. The planes can be rotated in three-space about an axis which lies on the plane. The standard two-dimensional transformations of scaling, translation and rotation can be applied to the control points which lie in a single plane. A plane can even be divided by a line of symmetry, where the movement of a control point on one side of the line causes the movement of a corresponding control point on the other side.

The planes themselves can be manipulated as single entities. One example is the distribution of planes along a curve. A free-hand curve is drawn in three-space and the planes are evenly distributed along the curve, oriented perpendicular to its tangent. This provides a powerful free-hand sculpting method.

Logical Groupings. In addition to the inherent groupings of planes of control points, other groups can be defined. The other groups are defined according to user needs and represent some desired model deformation. For

example, some control points are manipulated to provide a desired degree of curvature. When these control points are placed into their proper relative positions, they are grouped together. This group of control points can be moved as a unit, a logical group.

The control points of a logical group are not necessarily planar, but can be manipulated using three-dimensional transformation techniques.

Groupings of Symmetry. As an aid to the free-hand sculpting of models, groupings of symmetry provide the means of creating symmetrical models. Symmetrical grouping is viewed as dividing the STU box with mirrors as planes of symmetry. There are three types of symmetry: 1) single plane, 2) double plane and 3) triple plane.

In single plane symmetry every control point is grouped with a single corresponding control point. This correspondence is as if a mirror is used to divide the STU box. The basic functional relationship $f(x,y,z)$ for a symmetrical grouping, where the plane of symmetry is transformed to be the plane $x=\text{constant}$, is:

$$x'=x-dx', y'=y+dy' \text{ \& } z'=z+dz'$$

In double plane symmetry every control point is grouped with three corresponding control points. The correspondence is as if two perpendicular mirrors are used to divide the STU box. The three functional relationships for this symmetrical grouping, where the planes of symmetry are

transformed to be the planes $x=\text{constant}$ & $y=\text{constant}$, are:

$$\begin{aligned} x' &= x - dx', \quad y' = y + dy' \quad \& \quad z' = z + dz' \\ x' &= x - dx', \quad y' = y - dy' \quad \& \quad z' = z + dz' \\ x' &= x + dx', \quad y' = y - dy' \quad \& \quad z' = z + dz' \end{aligned}$$

In triple plane symmetry every control point is grouped with seven corresponding control points. The correspondence is as if three perpendicular mirrors are used to divide the STU box. The seven function relationships for this symmetrical grouping are:

$$\begin{aligned} x' &= x - dx', \quad y' = y + dy' \quad \& \quad z' = z + dz' \\ x' &= x - dx', \quad y' = y - dy' \quad \& \quad z' = z + dz' \\ x' &= x + dx', \quad y' = y - dy' \quad \& \quad z' = z + dz' \\ x' &= x - dx', \quad y' = y - dy' \quad \& \quad z' = z - dz' \\ x' &= x - dx', \quad y' = y + dy' \quad \& \quad z' = z - dz' \\ x' &= x - dx', \quad y' = y - dy' \quad \& \quad z' = z - dz' \\ x' &= x + dx', \quad y' = y - dy' \quad \& \quad z' = z - dz' \end{aligned}$$

Rigid Body Motions and Regular Deformations

Rigid body motions and regular deformations as described by Barr (1984) can be accomplished using FFD. The reason for using FFD to accomplish these deformations is to provide a homogenous deformation environment. Such deformations are achieved by treating the control point lattice as the object to be manipulated. These manipulations produce a deformed region which is used to deform geometrical models in a consistent manner. For high level manipulations, displaying the control point lattice satisfies no need and provides no utility. The FFD is used as an underlying manipulative method; visibility of the lattice serves no useful function.

Scaling, Rotation & Translation

The basic graphics transformations of scaling, rotation and translation are the fundamental means of achieving the regular body motions & regular deformations. When any of these transformations, or any composite transformation, is applied to every control point in the FFD lattice, it causes the same transformation to be performed on each coordinate in the FFD region during deformation calculation. The ability of the deformation to perform the transformation is known as the transformation inheritance property. For example, to rotate the coordinates within a FFD region, the rotation transformation is applied to the control point lattice. After this rotation, the deformation produces rotated coordinates.

Barr, 1984, identifies three high-level deforming operations: 1) tapering, 2) bending and 3) twisting. These are easily formulated as special use FFDs.

Tapering

Tapering is the scaling of two of the coordinate axis values based upon the value of the third. For example, a tapering commonly used in computer graphics is the perspective transformation. In the perspective transformation, the X and Y values of a coordinate are scaled based upon that coordinate's Z value, while the Z value becomes normalized.

The problem with using the standard perspective transformation to taper a model is that the Z coordinate value becomes normalized. In the tapering of a model, the Z coordinates must not change. So far, the presentation has centered around tapering on the Z axis because of the perspective transformation example. As a general tool in geometric modeling, it may be desirable to taper about an arbitrary axis. Since STU axes of FFD regions are oriented arbitrarily, such tapering can occur along any of the STU axes.

To perform the simple tapering of a geometric model, a $1 \times 1 \times 1$ FFD region is created in the region of world coordinate space to be tapered. When the tapering is associated with a specific modeled object, the region should be tied to that object's size. The orientation of the FFD is such that one of the axes, the U axis for example, is parallel to the axis of taper. The inverse FFD instance transformation I_t^{-1} is used to align the STU axes with the XYZ axes. An additional translation, represented by T, is needed to put the axis of taper coincident with the Z axis. The X and Y values of the four control points at the extreme of the Z axis are scaled. The scaling of the X and Y coordinate values can be by some specified percentage, or by some function of the Z coordinate value. Finally, the control points are transformed to their original orientation by the composite transformation $T^{-1}I_t$, and the model is deformed.

Bending

Bending is a basic function. Objects are bent according to three specific parameters: 1) axis, 2) radius and 3) degree. The bend can be viewed as taking a modeled object and bending it around a pipe. The axis of bend is the centerline of the pipe, the radius of bend is the pipe's radius and the degree of bend is the angle of the arc where the pipe is in contact with the object.

The first problem to be solved is how to define the FFD region. For the purposes of discussion, the bend provides curvature about the u-axis. The length of the s-axis, in world coordinate units, is related to the radius and degree of bend. The formula for the length of the s-axis below is derived from the formula of the circumference of a circle:

$$\text{len}_s = \frac{\text{angle} \cdot \text{radius} \cdot \pi}{180}$$

The FFD region is oriented so that the axis of bend is parallel with the u axis. The region is size-tied to the object in the t & u axes with the s axis having the length len_s . The control point densities along the t & u axes, m & n, are each 1, while the control point density along the s-axis places an upper limit on the angle of bend. For bend angles less than or equal to 90 degrees, the s-axis density value does not need to be greater than two. For bend angles less than or equal to 180 degrees, the s-axis density does

not need to be greater than three. (Although this last case will handle bends greater than 180 degrees, distortion can be introduced because of arc length parameterization.)

The simplest way to calculate the control point values is to calculate them for the case where the axis of bend is the z axis and then apply a transformation to the calculated control point values. To do this, the transformation matrix B_t is derived which puts the FFD region in the position shown in figure 4. The control point values themselves are not transformed, but are replaced with new values.

In the case where the angle of bend is less than or equal to 90 degrees, a $2 \times 1 \times 1$ control point density, the twelve control point values are assigned as follows:

```

theta := angle/2
x1 := radius*sin(theta)
x2 := (radius+lent)*sin(theta)
y1 := radius*cos(theta)
y2 := (radius+lent)*cos(theta)
y3 := radius/cos(theta)
y4 := (radius+lent)/cos(theta)
CP0,0,0 := (-x1, y1, 0)
CP0,0,1 := (-x1, y1, lent)
CP0,1,0 := (-x2, y2, 0)
CP0,1,1 := (-x2, y2, lent)
CP1,0,0 := (0, y3, 0)
CP1,0,1 := (0, y3, lent)
CP1,1,0 := (0, y4, 0)
CP1,1,1 := (0, y4, lent)
CP2,0,0 := (x1, y1, 0)
CP2,0,1 := (x1, y1, lent)
CP2,1,0 := (x2, y2, 0)
CP2,1,1 := (x2, y2, lent)

```

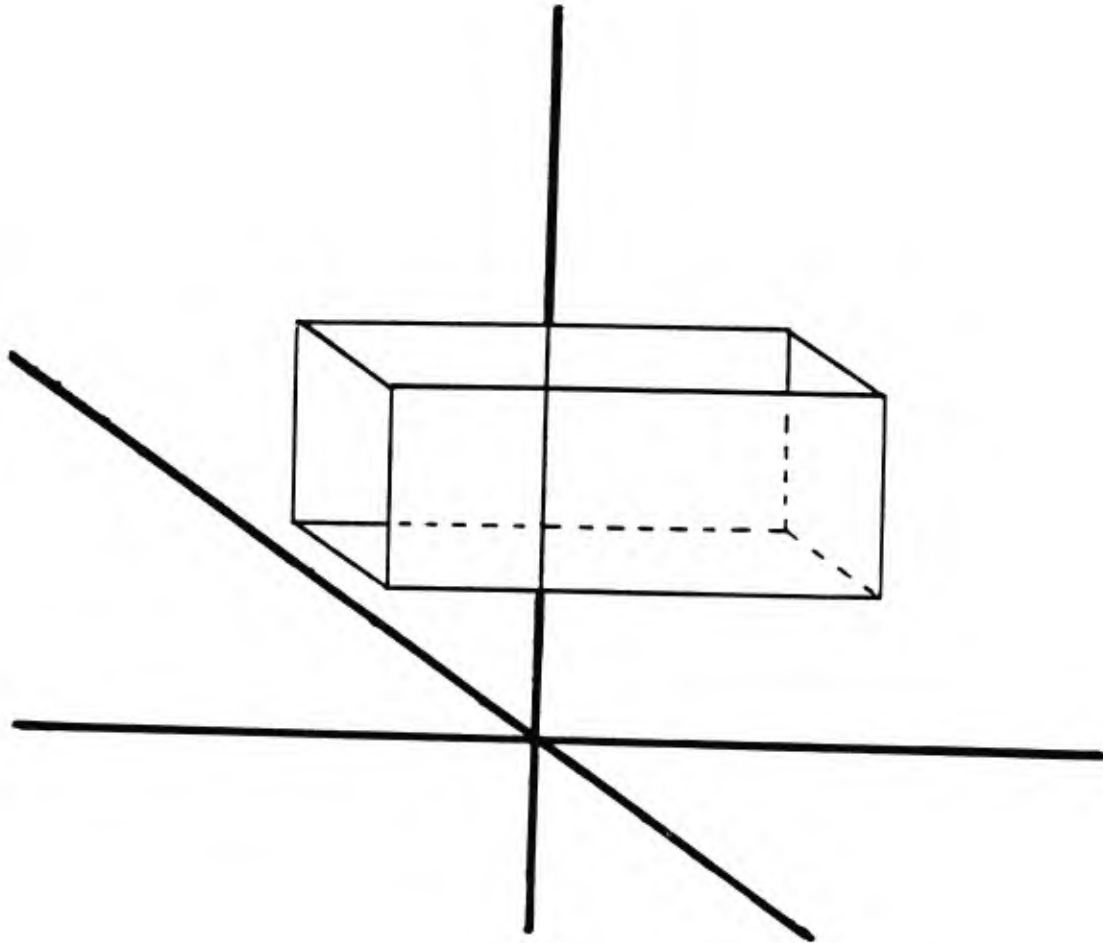


Figure 4

**Orientation of FFD Region
For Deriving a Bend Deformation**

The orientation of the FFD region used to calculate control point placement for creating a bend deformation is depicted. The XYZ axes in this example form a right-handed coordinate system. The far Z plane is at $Z=0$; the near plane is at $Z=\text{len}_z$. The bottom plane is at $Y=\text{Radius}$; the top plane is at $Y=\text{Radius}+\text{len}_y$. The left plane is at $-(\text{len}_x/2)$; the right plane is at $\text{len}_x/2$.

The weights of the control points at each of the two ends, i.e. $CP_{0,j,k}$ & $CP_{2,j,k}$, are assigned the value 1.0, and the weights of the control points in the center, i.e. $CP_{1,j,k}$, are assigned the value $\cos(\theta)$. This provides the control point relationships necessary to derive a circular arc from the Bernstein polynomial basis. The geometry of this relationship is shown in figure 5. Once the control points are set to these values, they are transformed by B_i^{-1} , the deformation is calculated and can then be removed.

The geometric relationship shown in figure 5, for calculating circular arcs in the case where the angle is less than or equal to 90 degrees (a quadratic basis function), is used to derive the geometrical relationship for calculating circular arcs where the angle is less than or equal to 180 degrees (a cubic basis function), as depicted in figure 6.

The case where the angle to be calculated is less than or equal to 180 degrees is calculated as follows: The matrix B_i is derived as in the previous case. The sixteen control points are assigned the following values:

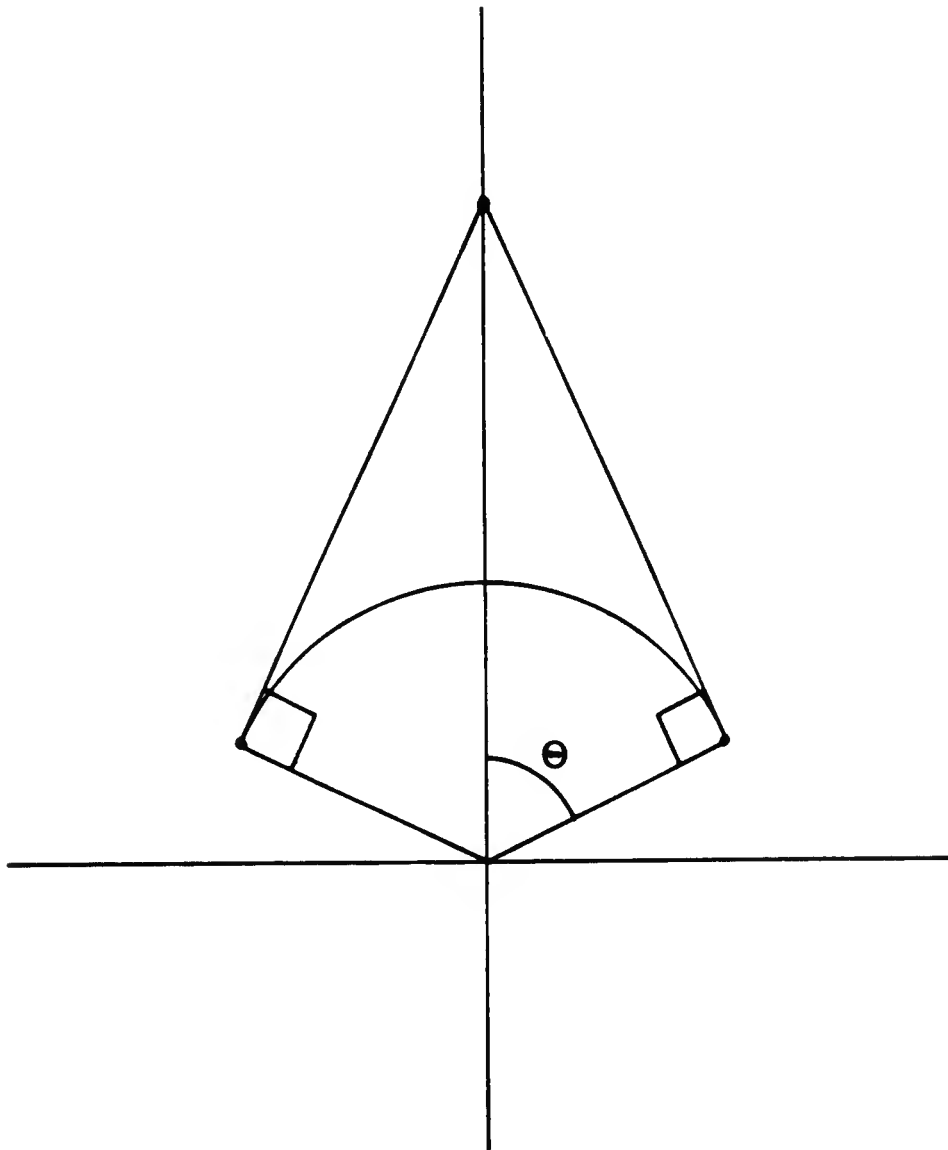


Figure 5

**Control Point Placement for Deriving
Circular Arcs with a Quadratic Basis Function**

Theta represents one-half of the bend angle.

Control Point Placement for Deriving Circular Arcs with a Cubic Basis Function

Theta represents one-half of the bend angle. In this configuration, only the circled control point position is calculated. The same method is used to derive the end coordinates in the cubic case and the quadratic case. The coordinate of the circled control point is used as an offset from the two end control points, taking advantage of the of the convex hull's symmetry.

```

theta := angle/2
d := (2*cos(theta))+1
x1 := radius*sin(theta)
x2 := (radius+lent)*sin(theta)
dx1 := 2*radius*((sin(theta)*cos(theta))/d)
dx2 := 2*(radius+lent)*((sin(theta)*cos(theta))/2)
dy1 := 2*radius*(sin2(theta)/1)
dy2 := 2*(radius+lent)*(sin2(theta)/d)
y1 := radius*cos(theta)
y2 := (radius+lent)*cos(theta)
CP0,0,0 := (-x1, y1, 0)
CP0,0,1 := (-x1, y1, lent)
CP0,1,0 := (-x2, y2, 0)
CP0,1,1 := (-x2, y2, lent)
CP1,0,0 := (dx1-x1, y1+dy1, 0)
CP1,0,1 := (dx1-x1, y1+dy1, lent)
CP1,1,0 := (dx2-x2, y2+dy2, 0)
CP1,1,1 := (dx2-x2, y2+dy2, lent)
CP2,0,0 := (x1-dx1, y1+dy1, 0)
CP2,0,1 := (x1-dx1, y1+dy1, lent)
CP2,1,0 := (x2-dx2, y2+dy2, 0)
CP2,1,1 := (x2-dx2, y2+dy2, lent)
CP3,0,0 := (x1, y1, 0)
CP3,0,1 := (x1, y1, lent)
CP3,1,0 := (x2, y2, 0)
CP3,1,1 := (x2, y2, lent)

```

As before, weights of the control points at the ends, i.e. CP_{0,j,k} & CP_{3,j,k}, are given the value of 1.0. Each of the eight control points in the center, i.e. CP_{1,j,k} & CP_{2,j,k}, are given a weight value of ((2*cos(theta))+1)/3. Also as before, all control points are transformed by B_t⁻¹, the deformation is calculated and can then be removed.

FFD regions can be used to bend models using this procedure. If the application is to bend part of a model, such as is done in creating a pipe with a series of bends in it, there must be some means provided by the software system to divide a model into separate pieces and then to rejoin them. In bending a pipe, the length and location of the FFD region are determined by the three bend parameters. The

pipe is broken into three pieces: 1) left of the region ($s\text{-value} < 0$), 2) within the region ($0 \leq s\text{-value} \leq 1$) and 3) right of the region ($s\text{-value} > 1$). The FFD region is bent, as described above, the left and right pieces are rotated into position using $1 \times 1 \times 1$ FFDs and the three pieces are joined back into a single piece. This procedure is shown in figure 7.

Twisting

Twisting is the rotation of coordinates according to those coordinates distance along the axis of twist. In order to define the twisting of a model two basic parameters must be identified: 1) the axis of twist and 2) the degree of twist rotation.

Since the rotations must be circular, the same methods are used to define circular arcs in the twist as are used in the bend. Although both the quadratic and cubic versions of control point placement and weighting can be used for twisting, the following discussion is concerned only with the cubic version. For example purposes, the axis of rotation is taken to be the t axis. A $1 \times 3 \times 1$ FFD is defined on the modeled object, with the line $s=u=0.5$ as the axis of twist. A transformation matrix T is defined which trans-

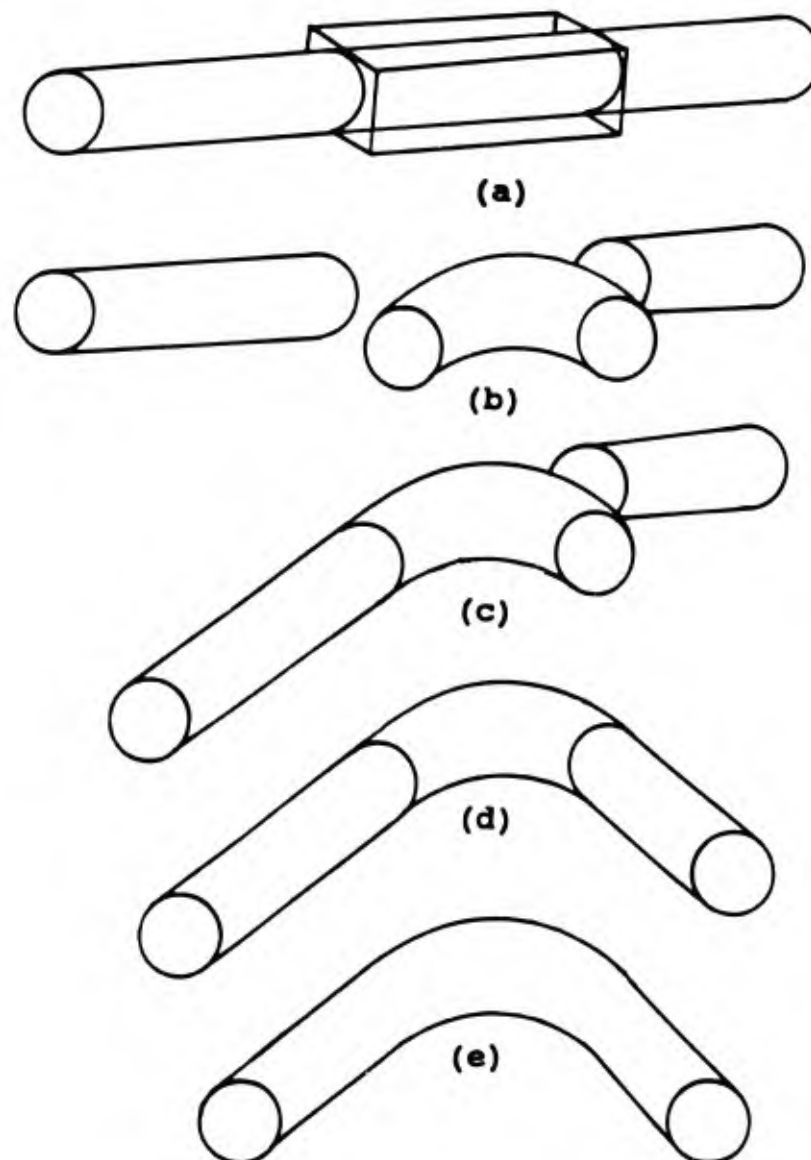


Figure 7

The Steps of the Bend Process

This illustration show the five steps of the bend process: a) divide the object into three pieces along the end boundaries of the FFD region, b) bend the center piece using the FFD region, c) rotate the left end into position (by use of a 1x1x1 FFD), d) rotate the right end into position (by use of a 1x1x1 FFD) and e) join the pieces into a single object.

forms the FFD region into the configuration shown in figure 8. The control points are not transformed, but are replaced as follows:

```

theta := degrees/2
d := (2*cos(theta))+1
x := lens/2
z := lenu/2
phi1 := sin-1(x/sqrt(x2+z2))-theta
phi2 := 2*(90-(phi1+theta))
x1 := x*cos(phi1)-z*sin(phi1)
z1 := -z*cos(phi1)-x*sin(phi1)
xprime := x1*sin(degrees)+z1*sin(degrees)
zprime := z1*sin(degrees)-x1*sin(degrees)
dx := 2*sqrt(x2+z2)*((sin(theta)*cos(theta))/d)
dz := 2*sqrt(x2+z2)*(sin2(theta)/d)
CP1,0,0 := YROTATE((x1,0,z1),-phi1)
CP1,1,0 := YROTATE((x1-dx,lent/3,z1-dz),-phi1)
CP1,2,0 := YROTATE((xprime+dx,(2*lent)/3,zprime-dz),
                    -phi1)
CP1,3,0 := YROTATE((xprime,lent,zprime),-phi1)
CP0,0,0 := YROTATE(CP1,0,0,180-phi2)
CP0,1,0 := YROTATE(CP1,1,0,180-phi2)
CP0,2,0 := YROTATE(CP1,2,0,180-phi2)
CP0,3,0 := YROTATE(CP1,3,0,180-phi2)
CP0,0,1 := YROTATE(CP1,0,0,180)
CP0,1,1 := YROTATE(CP1,1,0,180)
CP0,2,1 := YROTATE(CP1,2,0,180)
CP0,3,1 := YROTATE(CP1,3,0,180)
CP1,0,1 := YROTATE(CP1,0,0,-phi2)
CP1,1,1 := YROTATE(CP1,1,0,-phi2)
CP1,2,1 := YROTATE(CP1,2,0,-phi2)
CP1,3,1 := YROTATE(CP1,3,0,-phi2)

```

The weights of the top and bottom control points, i.e. CP_{i,0,k} & CP_{i,3,k}, are 1.0 and the weights of the middle control points, i.e. CP_{i,1,k} & CP_{i,2,k}, are ((2*cos(theta))+1)/3.

Because the formulas used in calculating the arcs were derived about the center of arc, the easiest way to use them is to calculate the convex hull for one set of corner points and then to define the others by different rotations. The

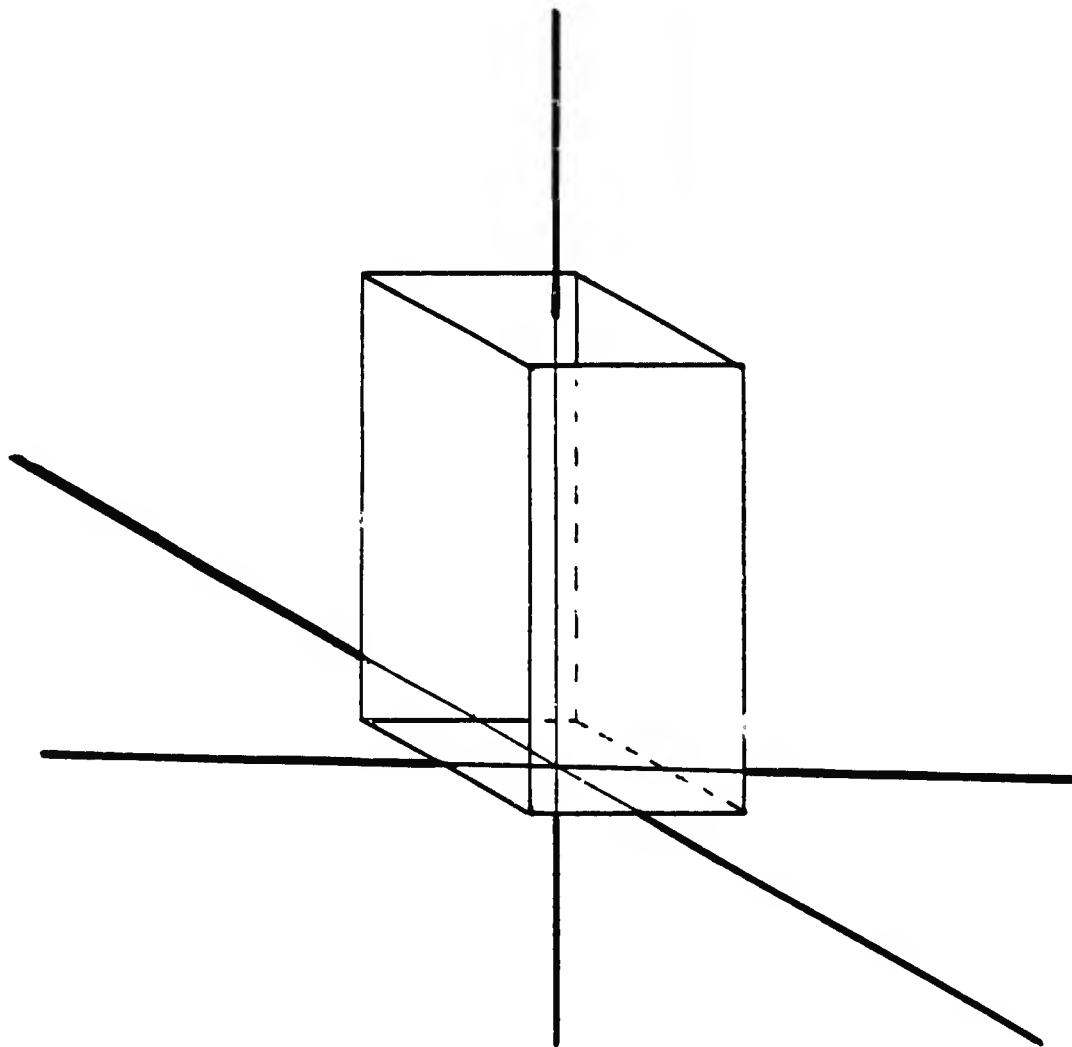


Figure 8

**Orientation of the FFD Region
for Deriving a Twist Deformation**

The twist is derived by placing the FFD region in the position show in the illustration. The Y-axis extends through the center of the top and bottom planes. The bottom plane is at $Y=0$, while the top plane is at $Y=len_t$.

function YROTATE above returns the rotation of a point about the Y axis. After derivation, the new control points are transformed by T^{-1} , the model is deformed and the FFD can then be removed.

Although the geometric relationship for creating circular arcs is the same for both twisting and bending, the difference lies with which control points are combined to create the arcs. The four control points forming a line parallel with the axis of twist in each corner of the box are given x and z values and weights so as to form a circular arc when projected onto the plane $y=0$. The y values of these control points evenly divide the FFD region along the y-axis. This is depicted in figure 9.

Continuity Control and Manipulation

As demonstrated by Parry (1986), when a FFD region is local, i.e. applied to only part of an object, discontinuities can occur at the region boundaries. The user of FFD in an interactive system desires control over continuity while manipulating the geometrical model.

Assuring Order of Continuity

As demonstrated by Parry (1986), any order of continuity can be imposed between deformed and undeformed regions. This is accomplished by the number of control points along the outside of the region which are kept in their original lattice positions. Only those control points along the

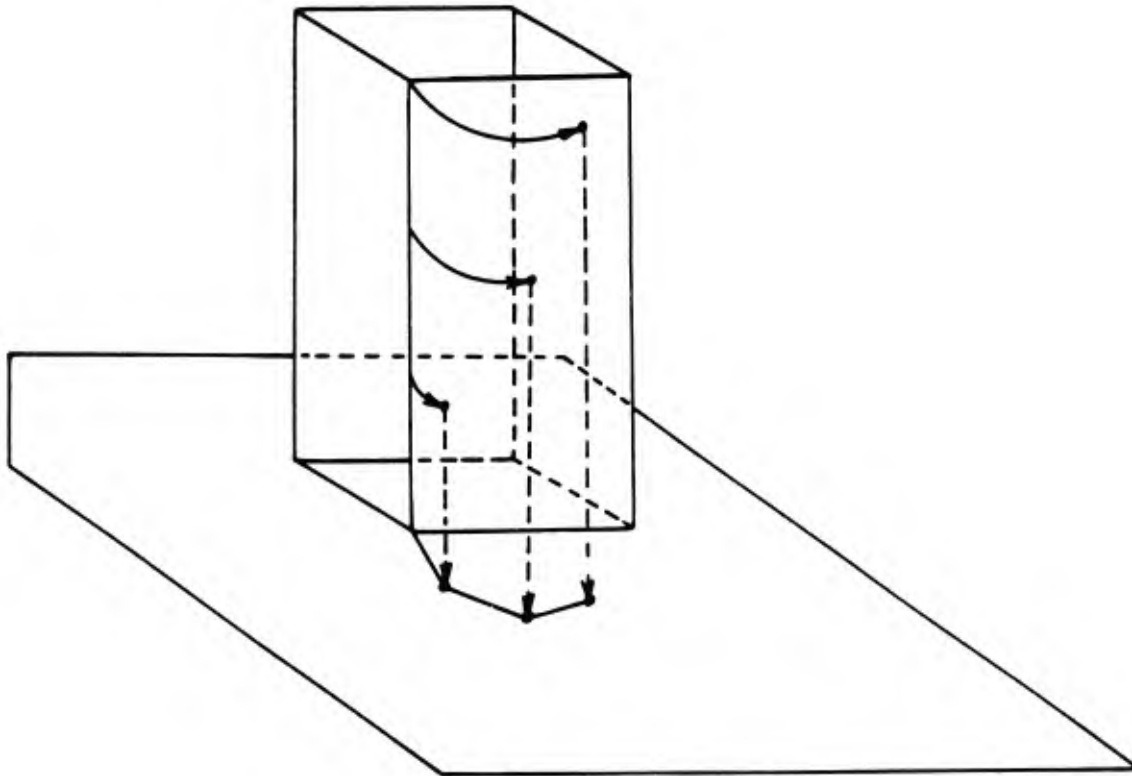


Figure 9
Configuration of a Twisting FFD

The illustration above shows how the four control points along a corner edge of the FFD are transformed. The projection of these points onto the plane $Y=0$ should be compared with the control point placement shown in figure 6.

dividing boundary plane(s) where continuity control is desired are fixed -- those planes which the modeled object pierce -- as well as some neighboring parallel planes. The number of planes to be fixed is dependent upon the order of continuity of desired at the associated boundary.

To maintain a specific order of continuity above G^1 , the control points along the outside boundary of the region are fixed at their original lattice positions with a fixed weight of 1.0. When the order of continuity is specified at the time of FFD region creation, control point density can be increased to assure the desired level of continuity.

To assure a G^0 continuity, the control points which lie on the six original clipping planes are fixed in both position and weight. To assure a G^1 continuity, both the control points which lie on the six original clipping planes and the adjacent control points are fixed in both position and weight. To assure a G^2 continuity, the control points which lie on the six original clipping planes, the adjacent control points and the control points adjacent to them are fixed in both position and weight.

Continuity control can be applied selectively along either end of any of the STU axes. For example, the G^2 continuity constraint can be applied to the left end of the s-axis. When the FFD is defined with a control point density of $2 \times 1 \times 2$, the system increases the density to $5 \times 1 \times 2$

and prevents the user from altering the positions or weights of the control points $CP_{0,j,k}$, $CP_{1,j,k}$ and $CP_{2,j,k}$.

Creating Discontinuities

In some design applications it is desirable to create discontinuities, such as the discontinuity crease in the hood of an automobile. Although discontinuities can be created by the boundary of a local FFD, the FFD itself can be used to create discontinuities.

To accomplish this, a FFD is created which provides for G^2 continuity at each end of the s-axis. The FFD has a control point density of $8 \times 1 \times 1$; three at each end of the s-axis to assure the G^2 continuity and three more in the middle. The three middle control points are placed coincident with the center control point and are then placed into a group. The group causes all of the three center control points to move with the same dx , dy & dz . Raising this group along the t-axis creates a discontinuity crease.

CHAPTER 3

Analysis and Summary

Eight basic methods for interactive model manipulation have been explored. These methods demonstrate the versatility of FFD as a general tool for use in the geometric modeling of free-form objects.

Basic Concepts

The versatility of FFD is expanded by the use of a rational basis function. The rational basis function enhances free-form sculpting. It makes possible the implementation of the higher level functions bending and twisting.

Treating the FFD control point lattice and STU axes in a graphical groups-and-items fashion facilitates the implementation of operations which aid the interactive user and extend the capability of FFD. This concept, when coupled with FFD's transformation inheritance property, allows higher level operations to be defined easily and implemented.

The FFD Region

The three possible methods of initially defining a FFD region (arbitrary size & orientation, object-tied size & arbitrary orientation and object-tied size & orientation) along with their two variations (selection/deselection and local regions) provide a complete set of FFD definition capabilities.

A fundamental problem with FFD, amplified when the initial region is tied to an object's size, is varying control point influence. Since the FFD region is always a parallelepiped, the distance from the side plane of the FFD region to the side of the contained model may vary due to curvature. This causes varying degrees of influence between a control point and the surface of the model. This is demonstrated in figure 10. In this figure, a FFD region is created object-tied in size and orientation to a sphere. The density is 2x1x2, which places an evenly distributed plane of control points on top of the sphere. When the weight of the top-center control point is increased to 1000, parts of the sphere are influenced only slightly while others are influenced to the verge coinciding with the top-center control point.

Because the deformation occurs in a predictable manner, this problem should not be a hinderance to a user who has gained experience with FFD. The interactive designer of even two-dimensional curves must gain an intuitive feel for

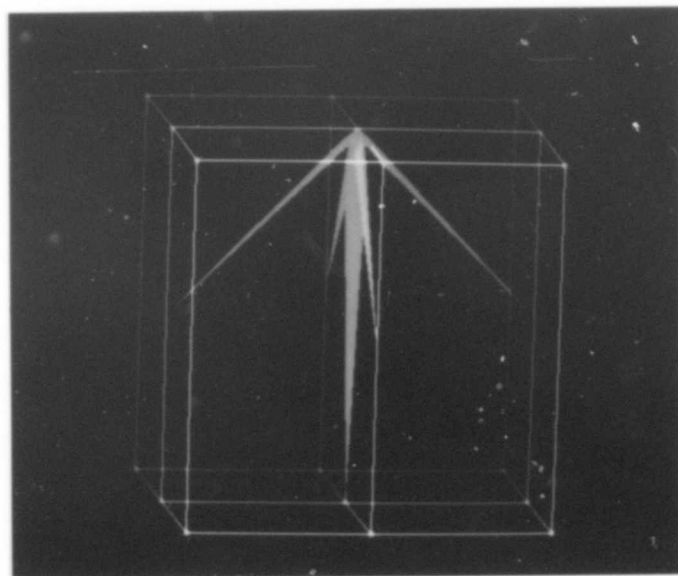


Figure 10

Example of Differing Control Point Influences

A sphere is deformed by changing the weight of the top-center control point to one thousand. This causes almost every point to collapse into the highly weighted control point. Places where the control point influence is naturally high are highlighted by the small amount of change caused by the top-center control point.

control point manipulations and their effect on the curve. No fundamental difference appears between developing an intuitive feel for manipulating two-dimensional curves and developing an intuitive feel for FFD manipulation.

Stored FFDs do not appear to pose any unique problems, but depend upon system implementation. Just as the designer of a standard graphics system can choose to have objects defined in a groups and items hierarchy with each displayed object being a specific instance, the FFD can be implemented using an instance approach. FFD instance is the deformation which must be performed on the object before it can be displayed. Alternatively, the FFD can be used to deform the object permanently and can then be removed. Both extremes are viable options for the system designer.

Simple Manipulations

Free-hand sculpting is a capability which is intuitively implemented in an interactive environment. With appropriate hardware to speed-up the deformation calculation, it could become almost as simple as working with clay.

The notion of groups is a fundamental computer graphics concept. The problem with groups in an interactive environment is how to identify the functional relationships in an interactive manner. Although symmetrical groupings can require many functional definitions, they can be determined algorithmically; this is not the case with logical group-

ings. It might be valuable to store logical grouping information so that it can be recalled and applied to different FFDs.

Rigid Body Motions & Regular Deformations

The transformation inheritance property of FFD is a valuable aid in accomplishing the rigid body motions and regular deformations.

The operations of bending and twisting utilize the rational basis function in order to derive circular arc deformations. Another fundamental problem with FFD when using rational basis functions is that arc length parameterization is not attained. This is a problem when performing circular arc deformations. This property prevents the even distribution of the original space along the circular arc deformation. For this reason, the twist operator cannot be inverted by performing a twist of negative degree. Figure 11a shows a box twisted by 180 degrees; the twist is tighter at the top and bottom than in the middle. When this twisted box is twisted by negative 180 degrees using a new FFD the result is not the original box, as can be seen in figure 11b.

Although the cubic form of the circular arc described in the section on bending is capable of creating bends greater than 180 degrees, arc length parameterization begins

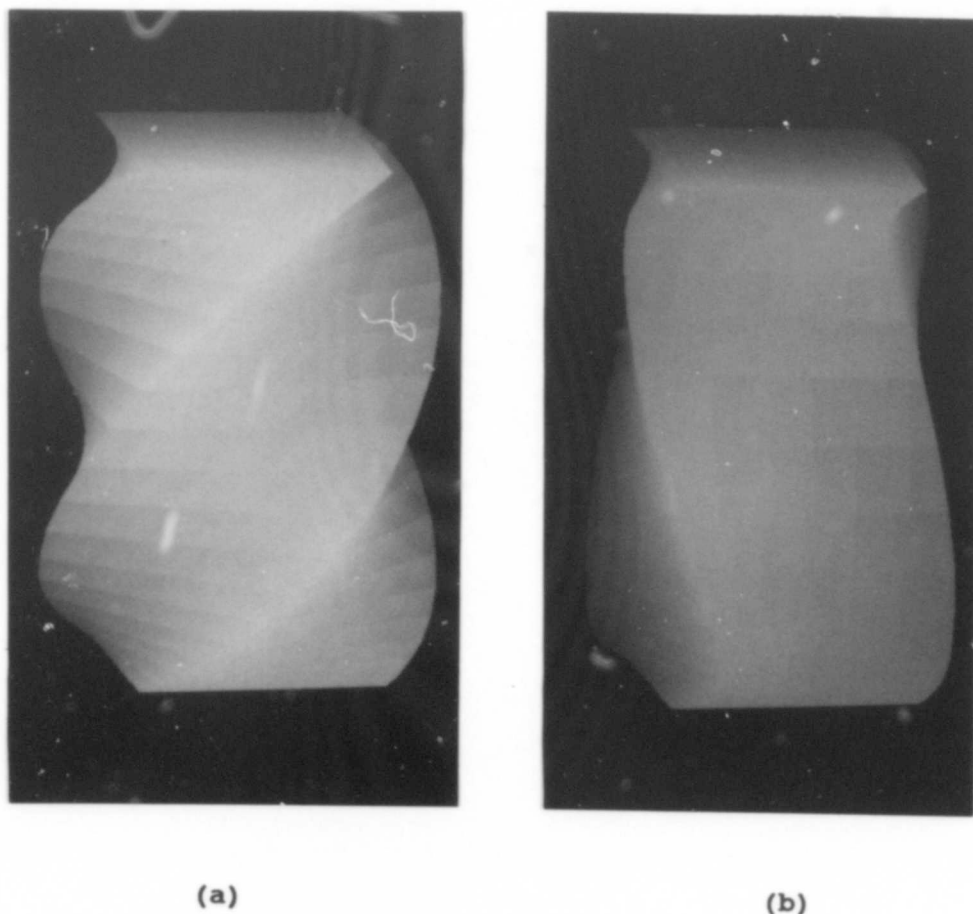


Figure 11

Attempt at Inverting the Twist

The box in photograph (a) was twisted 180 degrees by the twist deformation. The FFD was removed and a new one placed on the twisted box. Photograph (b) shows the result of using the new FFD to twist the box by -180 degrees. The result is a distortion of the original box caused by the absence of arc length parameterization.

to introduce greater distortions along the center of arc as the angle increases.

Continuity

Controlling discontinuities along the edges of local FFDs is a simple technique to implement in an interactive environment.

CHAPTER 4

Conclusion and Future Directions

FFD is a relatively new technique and shows promise as a general purpose method of manipulating geometric models.

Conclusion

An initial FFD region can be defined by three basic methods which can include two additional variations. The simple and high-level manipulations provide a rich, basic set of operations. Because of the inherent properties of FFD, the implementation of both region definition and manipulative operations is a simple task.

The FFD is a versatile tool. It supports a broad range of geometric model manipulations. Its implementation is simple and based upon well founded techniques: basis function with control point movement and standard transformation techniques for control point manipulation. Even though the implementation is simple, the result is a sophisticated tool.

Future Directions

As this research follows on the heels of Parry's work, so additional research is expected to follow. The follow-on

research falls into three categories: 1) user interface, 2) specific applications and 3) hardware architectures.

User Interface

This thesis concentrated on using the FFD as a means of implementing high-level operations on geometric models. The implementation of such high-level operations into a single interactive modeling package poses some interesting problems in user interface design. For example, what is the best way to specify the functional relationships of logical groups in an interactive environment? These groupings could be intricate and tedious to specify. What is the best method of specifying continuity control on the boundaries of a local FFD region? There are six sides which can each be specified to have a continuity ranging from G^1 through G^2 . How can the user be relieved of the tedium of specifying all six continuities every time a new region is created? What is the best way to specify the parameters of the bend and twist operations in an interactive environment?

The concept which led to this thesis research is an interactive 3-D image-space geometric modeling system; a graphics "clay" modeler. FFD appears to be well suited to this type of application. Further research into an appropriate user interface system should include this type of system.

Specific Applications

The FFD can be explored as a means of accomplishing the design task of some specific applications. For what applications might FFD best be suited? Specific applications in automotive design is one candidate. The research needs to identify which design characteristics are well suited to FFD and which design applications have these characteristics. This research will require an in-depth understanding of design processes.

Hardware Architectures

Because of the versatile nature of FFD, it may be valuable to develop a hardware FFD engine. Different architectures can be explored which exploit some of the concepts of this thesis as well as Parry's work. It may also be valuable to explore how such an engine could be incorporated into existing geometry pipeline architectures

BIBLIOGRAPHY

- Barr, A., "Global and Local Deformations of Solid Primitives," Computer Graphics, 18 (July 1984), 21-30.
- Foley, J.D. and A. VanDam, Fundamentals of Interactive Computer Graphics, Addison-Wesley, Reading, MA, 1984.
- Newman, W.M. and R.F. Sproull, Principles of Interactive Computer Graphics, Second Edition, McGraw-Hill, New York, NY (1979).
- Parry, S.R., "Free-form Deformations in a Constructive Solid Geometry Modeling System," Ph.D. Dissertation, Dept. of Civil Engineering, Brigham Young University, Provo, UT (1986).
- Sederberg, T.W. and S.R. Parry, "Free-form Deformation of Solid Geometric Models," Computer Graphics, 20 (August 1986).