Interactive Construction of Smoothly Blended Star Solids

Ergun Akleman
Visualization Sciences
Department of Architecture
Texas A&M University

Abstract

We introduce a computationally efficient method for interactive construction of implicitly represented star solids. These solids smoothly approximate control shapes that are defined by exact union and intersections over half-spaces containing the origin. Based on our algorithm, computation of a new solid shape when a new half-space is added or when the position of an existing half-space is changed can be performed in constant time and in space linear in the number of half-spaces.

Our implicit shape construction is based on a family of non-polynomials called *ray-linears* [Akl93]. Computation of an implicitly represented shape is a root finding process and in general can be extremely difficult. However since ray-linear implicit representations can easily be parameterized, the computation of any ray-linearly represented shape simplifies to evaluation of a parametric equation instead of root finding. But the related parametric equations are non-polynomials and their complexity increases as the number of building blocks (in this case half-spaces) increases. Our algorithm makes the computation of this parametric equation independent of the number of half-spaces. We develop an interactive platform based on our algorithm with which we are able to construct star solids that resemble human faces.

CR Categories and Subject Descriptors: I.3.7 [Computer Graphics]: Three Dimensional Graphics and Realism, I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling.

Additional Keywords and Phrases: Interactive Sculpting, Implicit and Parametric Representations, Solid Modeling.

1 Introduction

Implicit and parametric representations possess distinct modeling qualities.[1] Generally speaking, implicit representations provide control of blobby appearance and set operations; while, parametric representations provide control of detail and fast computation.

Implicit representations inherently provide a simple implementation of geometric operations, such as union and intersections by composition of functions [BI92]. This property of implicit representations makes them suitable for construction of solid shapes. Implicit representations also provide blobby appearances and create bulges. Since flesh depends on underlying fine details such as bone structure, the blobby appearances and bulges can represent flesh better than low frequency details which are independent of finer details. In fact, the implicitly based modeling tools that give control of blobbyness such as Wyvill's soft objects [Wyv90] or Blinn's exponential functions [Bli82] have long been succesfull in modeling organic looking shapes. There are two problems with implicit representations. The first problem is that so far no implicit represention based method exists that provides interactive construction and manipulation with *control shapes*[2]. In fact, control shapes are essential to describe fine details. The other problem is the computation of the shape of an implicitly represented solid. This computation is a root finding process in three-dimensions. Unless the functions have special properties, finding roots in three-dimensions can be extremely hard.

An extensive literature is available for parametric representations [BBB87]. Parametric representations have two properties that are useful for modeling. First, they use control shapes that allow control of detail. Second, computation of a parameterically defined shape is fast since it is simply an evaluation of either a polynomial or a rational polynomial. However, non-hierachical parametric representations can not provide blobby appearances. Forsey and Bartels introduced a hierachical method, hierachical B-splines, that provides control of different levels of details [FB88]. By viewing flesh as a low frequency

[1] In this paper, the word *representation* is used to include both equalities and inequalities.

[2] In order to provide an alternative control, Witkin and Heckbert used particles [WH94]

detail component that can be shaped independently, such hierarchical approaches can be successful in facial modeling and, in fact, Forsey used hierarchical B-splines to model faces.

1.1 A Dual Representation: Ray-Linears

Earlier we introduced a dual (both implicit and parametric) representation, ray-linear function family [Akl93]. Ray-linear representations, because of their dual nature, provide control of detail, control of blobby apperance, fast computation and union and intersection operations. Ray-linearly represented shapes are smoothly blended stars which are explained in section 2.

We have developed a modeling tool based on ray-linears. This modeling tool provides interactive construction and manipulation of smoothly blended solids described by star shaped control polyhedra.

In our modeling tool, each detail of the shape is represented by one convex polyhedron and control of details is accomplished by interactively changing these convex polyhedra. Modifiying each convex polyhedron results in a different solid shape.

Control of blobby apperance is accomplished by changing blending parameters. These blending parameters smooth out the sharp edges and corners of the star shaped control polyhedron. There are two types of blending parameters: global and local. The global blending parameter smooths out sharp edges resulting from the exact union of convex polyhedra. Blobby effects come mostly from this global blending parameter. Local blending parameters smooth out sharp edges and corners of convex polyhedra that result from the intersection of half-spaces. Different combinations of global and local blending parameters create different looks. Generally speaking, we can say that more blended shapes look fleshier, whereas less blended shapes look more robotic and less organic.

1.2 Interactive Computation of Ray-Linearly Represented Solids

Ray-linear implicit representations are easily parameterizable. Therefore, once the related parametric equations are obtained, computing the shapes is simply the evaluation of the related parametric equations. However, these parametric equations are non-polynomials and their complexity increases as the number of building blocks (in this case half-spaces) increases.

In this paper, we present a computationally efficient method which we develop in order to guarantee the interactive construction of ray-linearly represented solids. Based on our method, computation of a new solid shape when a new half-space is added or when the position of an existing half-space is changed can be performed in constant time and in space linear in the number of half-spaces.

The method we present here, by reusing the previous computation, makes the computation of these parametric equations independent of the number of half-spaces. As a result of this independency, when a user adds a new convex polyhedron or changes the position of a vertex of a convex polygon, a new solid shape can always be computed in constant time.

1.3 Organization of the Paper

In section 2, we introduce ray-linear implicit representations and explain how to parameterize them. In section 3, we show how to construct general ray-linear formulas from ray-linear formulas that result in symmetric stripes and half-spaces by using approximate and exact union and intersection operators. In section 4, we describe this interactive construction algorithm and implementation thereof. Section 5 concludes the paper.

In the appendices, we give the formal definitions of ray-linear functions, ray-linear implicit representations and ray-linearly represented shapes. We also formally explain parameterization of ray-linear implicit representations.

2 Ray-Linears

Ricci [Ric73] and later Requicha [RV82] independently showed that maximum and minimum operators over functions correspond with exact set operations over implicitly represented shapes. Maximum and minimum operators are easy to compute and they are widely used in solid modeling. In his paper Ricci also introduced approximate set operators that smooth out the sharp edges and corners resulting from exact set operations. In other words, solid shapes that are constructed by Ricci's approximate set operations can be viewed as smooth approximations of *control shapes* that are constructed by exact set operations. Ricci's approximate set operators therefore had the potential to be extremely useful for solid modeling. However, the formulas of Ricci's approximate operators include powers and roots. Successive application of these operators results in nested expressions with powers and roots. Since finding the roots of non-polynomials is especially hard, it is not feasible to implement Ricci's operators in an interactive modeler.

A subset of ray-linears, the non-negative ray-linears are closed under Ricci's exact and approxi-

mate union and intersection operators. Since ray-linear implicit representations can be easily parameterized [Akl93], the computation of ray-linearly represented shapes reduces to the evaluation of a parametric equation. This property of ray-linears yields fast computation. Therefore, a modeling tool based on non-negative ray-linears can provide interactive modeling with control shapes. We provide formal descriptions of ray-linear functions, ray-linear implicit representations and ray-linearly represented shapes in the appendix. Here, we give an informal description of ray-linears.

Ray-linears are functions that become linears when projected onto any ray starting from the origin. For instance, the implicit equation of Steiner's Roman surface which was formulated by Weierstrass is $x^2y^2+y^2z^2+z^2x^2+xyz=0$. This formula can be easily converted to a ray-linear formula plus a constant: $(x^2y^2+y^2z^2+z^2x^2)/(xyz)+1=0$. In this equation, $F(x,y,z)=(x^2y^2+y^2z^2+z^2x^2)/(xyz)$ is a ray-linear function. If we replace x with xt, y with yt and z with zt in the formula for $F(x,y,z)$, the result will be $F(x,y,z)$ times t. In other words, this formula is linear in t. Note that (xt,yt,zt) is actually the equation of the ray starting from the origin in the direction of the vector (x,y,z), therefore $F(x,y,z)$ is, in fact, a ray-linear function. If a function $F(x,y,z)$ is ray-linear, then it is straightforward to solve $F(x,y,z)+1=0$. As explained in the appendix, the solutions are actually parametric equations. For instance, if the vector (x,y,z) is given by a parametric equation, such as the parametric equation of a sphere, $x=\sin\theta\sin\psi$, $y=\cos\theta\sin\psi$, and $z=\cos\psi$, then a parametric equation of the Steiner surface is obtained:

$$x=-\sin^2\theta\cos\theta\sin\psi\cos\psi/G(\theta,\psi),$$

$$y=-\sin\theta\cos^2\theta\sin\psi\cos\psi/G(\theta,\psi),$$

$$z=-\sin\theta\cos\theta\cos^2\psi/G(\theta,\psi),$$

where $G(\theta,\psi)=\sin^2\theta\cos^2\theta\sin^2\psi+\sin^2\theta\cos^2\psi+\cos^2\theta\cos^2\psi$. Moreover, as explained in the appendix, it is possible to write simpler parametric equations that give the same surface.

Steiner's function belongs to a family of functions called monoids which are known to be easily parameterizable, therefore this parameterization is actually expected. In fact, the formulas of monoids can be converted to formulas in the form of a ray-linear plus a constant. The same goes for quadrics and we strongly believe that it may even be correct for cubics. In this paper, we do not deal with ray-linear rational polynomials since they are not closed under approximate set operations. In addition to polynomial ones above, there are many different classes of non-polynomial ray-linears. In fact, the non-negative ray-linears that we use in this paper are non-polynomials.

The ray-linear property gives a constraint over the ray-linearly represented shapes: each ray starting from the origin can intersect a ray-linearly represented shape at at most one point. Some examples of ray-linearly represented curves are shown in Figure 1. These shapes are known as star shapes. This restriction is not acceptable for modeling general curves. However, this is not such a serious restriction for interactive solid modeling since, unlike the two-dimensional case, in three-dimensions many interesting solid shapes are stars. For example, we have noticed that when ears are excluded, most human faces are star solids.

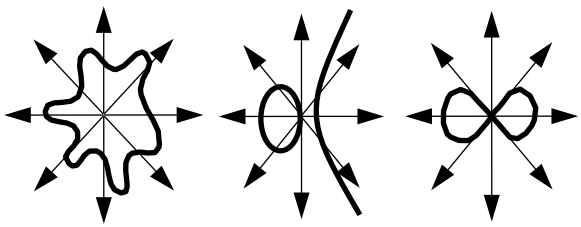

Figure 1: Examples of curves that can be generated by ray-linears.

3 Construction of Ray-Linear Implicit Representations

We use non-negative ray-linears in order to develop the modeling tool. As mentioned before and explained in appendix, non-negative ray-linears are closed under Ricci's exact and approximate union and intersection operators. For ease of reading, we always refer to non-negative ray-linears as ray-linears. Therefore, in order to define a control shape we can apply exact set operations iteratively to generate a structure that can be expressed as nested exact unions or exact intersections over a starting set of ray-linears. By replacing each exact union operator ($min(.,.)$) by an approximate union operator ($\sqrt[-p]{(.)^{-p}+(.)^{-p}}$) and each exact intersection operator ($max(.,.)$) by an approximate intersection operator ($\sqrt[p]{(.)^p+(.)^p}$), we obtain ray-linear implicit representations of smooth approximations of the ray-linearly represented control shape constructed by exact set operations. The starting set of ray-linear building blocks can be any ray-linear function. In the next section, we introduce some building blocks. An example of exact and approximate unions of convex

shapes is shown in Figure 2.

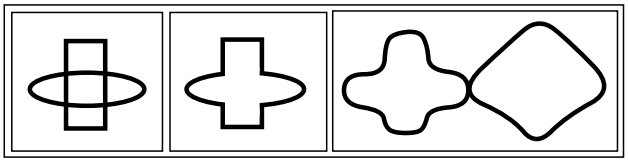

Figure 2: Two convex shapes, their union and two smooth approximations of the union.

4 Building Blocks

As is widely known, the distance function $F(x,y)=\sqrt[p]{|x|^p+|y|^p}$ leads to the following implicit inequality which provides a large number of shapes depending on the blending parameter p: $S=\{[x,y] \mid \sqrt[p]{|x|^p+|y|^p}\leq 1\}$. Note that the distance function is a ray-linear function. When p goes to ∞, S becomes square: $\blacksquare=\{[x,y]|max(|x|,|y|)\leq 1\}$. The related shapes are shown in Figure 3.

4.1 Symmetric Stripes

Since the maximum operator is an intersection operator over the implicitly represented shapes this square shape can be viewed as a control shape that is constructed by the intersection of two infinite symmetric stripes that are given by following implicit inequalities: $\blacksquare=\{[x,y] \mid |x|\leq 1\}$ and $\blacksquare=\{[x,y] \mid |y|\leq 1\}$. Note that both x and y are linears but are not always non-negative. On the other hand, absolute values of these linear functions, $|x|$ and $|y|$, are ray-linears. As a result, in order to describe a family of simple ray-linear building blocks we simply take the absolute values of linears. Let $L_i(\mathbf{v})$ be a linear function, then $|L_i(\mathbf{v})|$ is ray-linear. Note that the implicit inequality $|L_i(\mathbf{v})|\leq 1$ describes an n-dimensional infinite symmetric stripe. Therefore, $S=\{\mathbf{v}|max(|L_1(\mathbf{v})|,\ldots,|L_n(\mathbf{v})|)\leq 1\}$ is an n-dimensional control shape generated by the intersection of such n-dimensional infinite symmetric stripes. For instance, the following equation gives an hexagonal control shape: $S=\{[x,y] \mid max(|\sqrt{2}x|,|\sqrt{2}y|,|x+y|)\leq 1\}$. As explained in the previous section, the inequality $\sqrt[p]{|\sqrt{2}x|^p+|\sqrt{2}y|^p+|x+y|^p}\leq 1$ gives a smooth approximation of a hexagonal control shape. An example of the minimum operator is a star octagon shape given as the union of two squares: $S=\{[x,y] \mid min(max(|x|,|y|),max(|x+y|/\sqrt{2},|x-y|/\sqrt{2})\leq 1\}$ and the shape that can smoothly approximate this star octagon is given by the inequality: $\sqrt[-p_3]{(\sqrt[p_1]{|x|^{p_1}+|y|^{p_1}})^{-p_3}+(\sqrt[p_2]{|x+y|^{p_2}+|x-y|^{p_2}})^{-p_3}}\leq 1$, where p_1, p_2 and $-p_3$ are blending parameters.

Since infinite stripes are symmetric around the origin, their intersection can only generate symmetric shapes: *symmetric* polygons in two-dimensions or *symmetric* polyhedra in three-dimensions. Even-sided star or regular polygons with parallel edges are *symmetric* polygons in two-dimensions. Some examples of *symmetric* polyhedra in three-dimensions are the cube, octahedron, icosahedron, dodecahedron, cuboctahedron and great dodecahedron.

Another example of the use of symmetric stripes are non-toroidal super-quadrics [SP91] which are given by the inequality: $\sqrt[p^2]{(\sqrt[p^1]{|x|^{p^1}+|y|^{p^1}})^{p^2}+|z|^{p^2}}\leq 1$. The cube is the control shape for non-toroidal super-quadrics.

Symmetry is not always a desired feature for solid modeling. For instance, in two-dimensions not only irregular polygons but even regular polygons with an odd number of edges such as regular triangles or regular pentagons cannot be used as a control shape. In three-dimensions the regular tetrahedron cannot be used as control shape. In the next section we introduce ray-linear implicit representations of half-spaces to be able to describe more general control shapes.

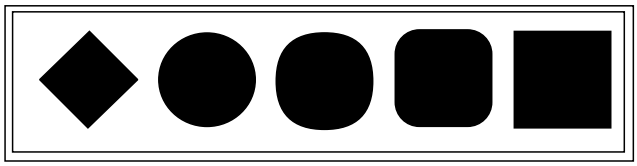

Figure 3: The shapes generated by distance functions for p values of $1,2,4,8,\infty$.

4.2 Half-Spaces

Since any convex or star polyhedron can be described as a combination of exact intersections and unions of half-spaces, we investigate the possibility of ray-linear representation of half-spaces. Half-spaces can be described by linear inequalities of the form $L(\mathbf{v})\leq 1$, however, this representation is not useful for our purposes since L is not always non-negative. Recall that since Ricci's approximate operators include roots, functions that are not always non-negative cannot be used. In order to find an alternative representation, we assume that there exists an operation $H(L(\mathbf{v}))$ such that $H(L(\mathbf{v}))$ is a ray-linear and describes the same half-space as linear function L:

$$\{\mathbf{v}|L(\mathbf{v})\leq 1\}=\{\mathbf{v}|H(L(\mathbf{v}))\leq 1\}. \qquad (1)$$

We can show that such an operation $H(L(\mathbf{v}))$ indeed exists.[3]

First note that $|L(\mathbf{v})| \leq 1$ will create two inequalities: **1:** $L(\mathbf{v}) \leq 1$ if $L(\mathbf{v}) \geq 0$ and **2:** $-L(\mathbf{v}) \leq 1$ if $L(\mathbf{v}) < 0$. The second inequality is an unwanted one that creates symmetric stripes. This unwanted inequality can be eliminated by simply equating the negative part to zero. This operation does not affect the shape since $L(\mathbf{v}) - 1$ will still be negative. In other words, if we can ensure $H(L(\mathbf{v})) = 0$ when $L(\mathbf{v}) < 0$ and $H(L(\mathbf{v})) = L(\mathbf{v})$ when $L(\mathbf{v}) \geq 0$, $H(L(\mathbf{v}))$ will be a ray-linear that satisfies the condition given in equation 1. One function that qualifies is

$$H(L(\mathbf{v})) = 0.5L(\mathbf{v}) + 0.5|L(\mathbf{v})|. \tag{2}$$

5 Interactive Construction

Construction of a control shape is not unique. In other words, the same control shape can be constructed many different ways. Each one of them will provide different control of blending. For instance, all three of the following implicit representations give the same control shape, a square:

$max(H(x),H(y),H(-x),H(-y)) \leq 1$

$max(max(H(x),H(y),H(-x)),H(-y)) \leq 1$

$max(max(H(x),H(y)),max(H(-x),H(-y))) \leq 1$

However, when the maximum operators are changed by approximate intersection operators, each one of them provides a different type of blending:

$\sqrt[p]{H(x)^p + H(y)^p + H(-x)^p + H(-y)^p} \leq 1$

$\sqrt[p2]{(\sqrt[p1]{H(x)^{p1} + H(y)^{p1} + H(-x)^{p1}})^{p2} + H(-y)^{p2}} \leq 1$

$\sqrt[p3]{(\sqrt[p1]{H(x)^{p1} + H(y)^{p1}})^{p3} + (\sqrt[p2]{H(-x)^{p2} + H(-y)^{p2}})^{p3}} \leq 1$

The first one has only one blending parameter, the second one has two blending parameters, and the last one has three blending parameters. The square is one of the simplest control shapes. For a complicated control shape the number of representations will be extremely high.

For interactive construction we need a simple and unique representation, so we have chosen the following construction methodology. First, we describe convex polyhedra as intersections of half-spaces. Then, we construct a control polyhedron as a union of convex polyhedra. In other words, the formulas of the ray-linears will be strictly of the form

$$F(\mathbf{v}) = \sqrt[-p_G]{\sum_{i=1}^{n-1} F_i(\mathbf{v})^{-p_G}}, \tag{3}$$

where,

$$F_i(\mathbf{v}) = \sqrt[p_i]{\sum_{j=1}^{m_i} H((L_{i,j}\mathbf{v}))^{p_i}},$$

where $n-1$ is the number of convex polyhedra and m_i is the number of half-spaces in the i^{th} convex polyhedron. Let $\mathcal{N}$ denote the total number of half-spaces then the $\mathcal{N} = \sum_{i=1}^{n-1} m_i$ We call p_G the global blending parameter and p_i's the local blending parameters. The global blending parameter is for smoothing out edges and corners resulting from the union operator. Some examples of the effect of changing the global blending parameter are shown in Figure 4. Ricci's approximate union operations generate bulges [Blo91]. Since they give a fleshy appearance, bulges are not a problem in our application. The local blending parameters are for smoothing out the sharp edges and corners of the convex polyhedra which are described by the intersection of half-spaces. Some examples of the effect of changing local blending parameters p_i are shown in Figure 5.

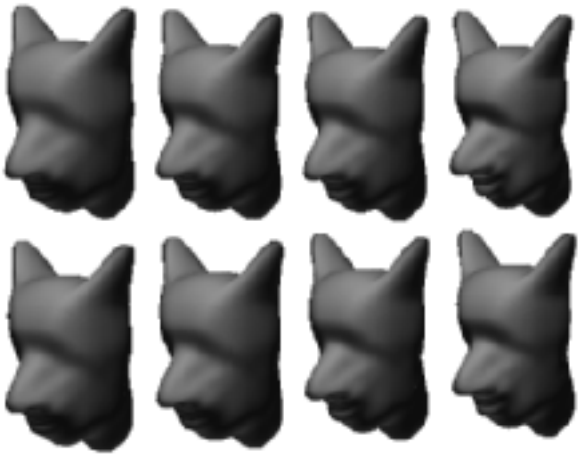

Figure 4: The effect of global blending parameter.

5.1 Description and Manipulation of Control Shape

In order to simplify the description of convex polyhedra, we restrict the user to those that can be represented by the union of two infinite prisms. The user

[3] Ricci solved this problem by using exponentials. We cannot use $e^{L(v)}$'s, since these exponential building blocks do not result in ray-linears. However, we observe that these exponential building blocks result in ray-exponentials which seem to be useful for modeling and need a further investigation. In addition, note that we cannot use affine building blocks since they do not result in ray-linears.

simply draws one convex polygon on each of two perpendicular planar surfaces.[4] Each polygon defines an infinite prism whose axis is perpendicular to the planar surface on which the polygon is drawn. (i.e., the edges of the polygons describe half-spaces perpendicular to the planar surface on which the polygon is drawn.) The intersection of the two prisms describes a convex polyhedron. The control polyhedron is obtained from the union of the polyhedra that have been described in this manner.

After a control polyhedron has been described, the user can change the smoothly blended solid by manipulating the half-spaces that describe the control polyhedron. In our program this manipulation is accomplished by simply moving the vertices of the polygons.

Figure 5: Evolution of human beings from robots: the effect of local blending parameters.

5.2 Interactive Alterations are Cheap

In order to compute the smoothly blended solid shape we have to compute equation 7 in the appendix, $\mathbf{v}_{i*} = \frac{\mathbf{v}_i}{F(\mathbf{v}_i)}$, for each vector $\mathbf{v}_i$ defined by the guide shape. Note that the computation of this equation mainly involves computation of $F(\mathbf{v}_i)$, where $\mathbf{v}_i$ is a vector given by the guide shape. Since $F(\mathbf{v}_i)$ consists of the ray-linear functions that correspond to half-spaces (in equation 3), the price of the computation is linearly dependent on the total number of half-spaces $\mathcal{N}$. This implies that each alteration takes computation time linear in $\mathcal{N}$. To effectively support interactivity, alterations must be cheap. This linear dependency therefore is not acceptable. In this section, we show that it is possible to make the computation of any $F(\mathbf{v}_1)$ result from an alteration independently from $\mathcal{N}$.

[4] These perpendicular planar surfaces are either the x=0 and y=0 planes or the x=0 and z=0 planes.

Recall that when we change a control shape, we actually change only one half-space. The formulas for the other half-spaces are unchanged. Therefore, we do not have to compute all these formulas again. Using this information equation 3 can be simplified.

First let us assume that the user changes the half-space i,j that is represented by the formula $H(L_{i,j}(\mathbf{v}))$. Let the old values of functions in equation 3 be denoted as $F_{old}(\mathbf{v}_1)$, $F_{old_i}(\mathbf{v}_1)$ and $H(L_{old_{i,j}}(\mathbf{v}_1))$. In addition, let the new values to be computed be denoted as $F_{new}(\mathbf{v}_1)$, $F_{new_i}(\mathbf{v}_1)$ and $H(L_{new_{i,j}}(\mathbf{v}_1))$. Then equation 3 can be rewritten:

$$F_{new}(\mathbf{v}_1) = \sqrt[-p_G]{F_{old}(\mathbf{v}_1)^{-p_G} - F_{old_i}(\mathbf{v}_1)^{-p_G} + F_{new_i}(\mathbf{v}_1)^{-p_G}}, \quad (4)$$

where,

$$F_{new_i}(\mathbf{v}_1) = \sqrt[p_i]{F_{old_i}(\mathbf{v}_1)^{p_i} - H(L_{old_{i,j}}(\mathbf{v}_1))^{p_i} + H(L_{new_{i,j}}(\mathbf{v}_1))^{p_i}},$$

and $H(L_{new_{i,j}}(\mathbf{v}_1))$ is given by equation 2.

Note that equation 4 is independent of $\mathcal{N}$. In other words, when a control shape is manipulated the speed of computation is independent of $\mathcal{N}$.

In the process of construction, the user not only can change half-spaces but can also add new convex polyhedra. In this case we would again like the computation to be independent of $\mathcal{N}$. Let the ray-linear representation of the new convex polyhedron be denoted as $F_n(\mathbf{v})$. Then equation 3 can be rewritten for each vector $\mathbf{v}_1$ as

$$F_{new}(\mathbf{v}_1) = \sqrt[-p_G]{F_{old}(\mathbf{v}_1)^{-p_G} + F_n(\mathbf{v}_1)^{-p_G}}, \quad (5)$$

where,

$$F_n(\mathbf{v}) = \sqrt[p_n]{\sum_{j=1}^{m_n} H((L_{n,j}(\mathbf{v}_1))^{p_n}}.$$

Computation of equation 5 is still independent of $\mathcal{N}$ but depends on m_n. Since the value of m_n in general is between 6 and 10, this dependency does not create any problem.

Based on equations 4 and 5, we develop a platform for interactively constructing smoothly blended solids. All smoothly blended solids that are shown in Figure 6 are constructed in approximately fifteen minutes. A set of simpler ray-linearly represented human faces are shown in figure 7 by adding skin textures, eyes and teeth.

Figure 6: All these faces constructed in approximately fifteen minutes by changing parts such as the nose, chin or lips.

6 Discussion

In order to simplify the user interface we provide only a certain type of control polyhedra: those that are given as a union of convex polyhedra that are constructed by the union of two perpendicular prisms. We are planning to remove this restriction by using virtual reality tools to manipulate the half-spaces freely in three-dimensions.

Since we use half-spaces as building blocks the faces of the control shape have to be planar. However, in general this does not have to be the case. For instance, a cylinder can be a control solid since it is the exact intersection of an infinite cylinder and two half-spaces, and, both infinite cylinders and half-spaces can be described by ray-linear implicit representations.

By using ray-linears we are limited to star shapes. Due to this restriction over the control solid we cannot interactively construct all solid shapes. However, ray-linear implicit representations of the parts of a solid shape can be obtained by interactive construction. Once the ray-linear formulas for each part are obtained, we can combine these formulas by using Ricci's set operations.

It is also possible to use these formulas in other implicit representation based solid modeling tools such as constructive solid geometry [RV82], soft objects [Wyv90] or Blinn's exponential equations [Bli82]. Since it is also possible to deform them [Bar84, SP91], ray-linears can even be used as building blocks for applications that require shapes other than stars.

The operations over ray-linear representations of half-spaces are closely related to the super-elliptic local blending of Rockwood [RO87] and convex superquadrics [Bar81, SP91] are a subset of ray-linears that are obtained by the approximate union operations over ray-linear representations of infinite symmetric stripes [Akl93]. Ray-linears are different than hyperquadrics [Han88] which are a superset of superquadrics [Akl94]. Whereas hyperquadrics generalize superquadrics by going to higher-dimensions, ray-linears arise by introducing non-homogeneous functions.

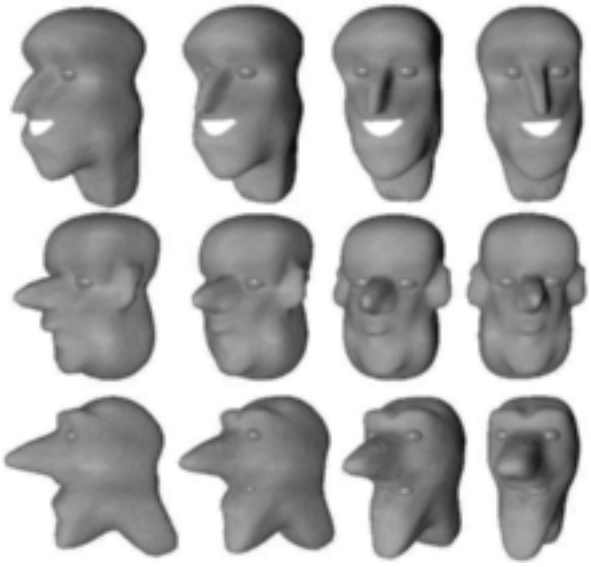

Figure 7: Ray-linearly represented human faces by adding skin textures, eyes and teeth.

Another property of our modeling tool is that the data requested for representation of one face are limited. Less than fifteen convex polyhedra are

enough for one face. Each convex polyhedron can be given by intersection of approximately ten half spaces. Each half-space can be described by two coefficients. These coefficients can be represented with just one byte since these solid shapes are quite robust (small changes in coeeficients do not change the shape).

7 Acknowledgments

We would like to thank to Graphics, Visualization and User interface Center at Georgia Tech and Computer Science Department at Yildiz Technical University. We also would like to thank Lambert Meertens, Steve Capell, Rimli Sengupta, James D. Foley, Larry F. Hodges, Ahmet Kuyumcu Donald House and Karen Hillier-Woodfin for their helpful suggestions.

References

[Akl93] E. Akleman. Construction of convex and star shapes with yontsal functions (always positive ray-linears). *International Symposium on Computer and Information Sciences*, 8, November, 1993.

[Akl94] E. Akleman. Ray-quadrics: A building block for implicit representations of solids. *Proceedings of IX. International Symposium on Computer and Information Sciences*, IX., November 1994.

[Bar81] A. H. Barr. Superquadrics and angle-preserving transformations. *IEEE Computer Graphics and Applications*, 1(1), 1981.

[Bar84] A. H. Barr. Global and local deformations of solid primitives. *Computer Graphics*, 18(3), 1984.

[BBB87] R. H. Bartels, J.C. Beaty, and B. A. Barsky. *An Introduction to Splines for use in Computer Graphics and Geometric Modeling*. Morgan Kaufmann, Los Altos, California, 1987.

[BI92] C. Bajaj and I. Ihm. Smoothing polyhedra using implicit algebraic splines. *Computer Graphics*, 26(2), 1992.

[Bli82] J. I. Blinn. A generalization of algebraic surface drawing. *ACM transaction on Graphics*, 1(3), 1982.

[Blo91] J. Bloomenthal. Convolution surfaces. *Computer Graphics*, 25(4), July 1991.

[FB88] D. Forsey and R. Bartels. Hierarchical b-spline refinement. *Computer Graphics*, 22(4), 1988.

[Han88] A. J. Hansen. Hyperquadrics: Smoothly deformable shapes with convex polyhedral bounds. *Computer Vision, Graphics and Image Processing*, 44, 1988.

[Ric73] A. Ricci. A constructive geometry for computer graphics. *The Computer Journal*, 16(2), May 1973.

[RO87] A. P. Rockwood and J. Owen. Blending surfaces in solid geometrical modeling. In G. Farin, editor, *Geometric Modeling*. SIAM, Philadelphia, 1987.

[RV82] A. A. G. Requicha and H.B. Voelcker. Solid modeling: A historical summary and contemparay assessment. *IEEE Computer Graphics and Applications*, 2(2), March 1982.

[SP91] S. Sclaroff and A. Pentland. Generalized implicit functions for computer graphics. *Computer Graphics*, 25(4), July 1991.

[WH94] A. P. Witkin and P. S. Heckbert. Using particles to sample and control implicit surfaces. *Computer Graphics*, 27(3), 1994.

[Wyv90] B. L. M. Wyvill. Ray tracing soft objects. In B. L. M. Wyvill and J Bloomenthal, editors, *Modeling and Animating with Implicit Surfaces*. ACM SIGGRAPH tutorial, 1990.

A Definition and Parameterization of Ray-Linears

Let $\mathcal{V}$ be a vector space over reals. We use the letter $\mathbf{v}$ to stand for a vector from $\mathcal{V}$ and $\mathbf{0}$ for the origin. In two-dimensional space $\mathbf{v}=[x,y]$, in three-dimensional space $\mathbf{v}=[x,y,z]$. Further, let $\Re^{\oplus}=\Re^{+}\cup\{0\}$ denote the set of non-negative real numbers. We use the variable t to stand for an element of $\Re^{\oplus}$.

Definition 1 A function $F:\mathcal{V}\longrightarrow\Re^{\oplus}$ is a non-negative *ray-linear* iff $F(\mathbf{v}t)=F(\mathbf{v})\ t$, $\forall\mathbf{v}\in\mathcal{V}$ and $\forall t\in\Re^{\oplus}$.

For ease of reading, we always refer to non-negative ray-linears as ray-linears. Recall that a function F is homogeneous if $F(\alpha\mathbf{v})=\alpha^m F(\mathbf{v})$ for every real α. At first ray-linears may appear to be a special case of homogeneous functions because it is required that $F(\alpha\mathbf{v})=\alpha F(\mathbf{v})$. However, for ray-linears the condition

does not need to hold for negative α's. Thus the class of homogeneous functions is incomparable with the class of ray-linears. For instance absolute value function is ray linear but not homogeneous, [5] Since Ricci's approximate operations can only be applied to non-negative functions, the inclusion of absolute value is crucial for the construction of formulas using his approximate set operations [Ric73].

Definition 2 A *ray-linearly represented shape* is the one that is described by the following *ray-linear implicit representation*:

$$S = \{\mathbf{v} \mid F(\mathbf{v}) \leq 1\}, \tag{6}$$

where $F(\mathbf{v})$ is ray-linear.

Let $S(\mathbf{v}_1)$ be the intersection of a ray-linearly represented shape with a ray starting from the origin $\mathbf{0}$ in the direction of $\mathbf{v}_1$. The parametric equation of this ray is $\mathbf{v}=\mathbf{v}_1 t$ and therefore $S(\mathbf{v}_1)=\{\mathbf{v}=\mathbf{v}_1\ t \mid F(\mathbf{v})\leq 1\}$. Because of the ray-linear property of $F(\mathbf{v})$ the inequality in the representation of $S(\mathbf{v}_1)$ simplifies to a univariate linear inequality: $S(\mathbf{v}_1)=\{\mathbf{v}=\mathbf{v}_1\ t \mid F(\mathbf{v}_1)\ t\leq 1\}$. Thus, the following linear equation $F(\mathbf{v}_1)\ t=1$ gives the border of $S(\mathbf{v}_1)$. The solution in t of this equation is: $t_*=\frac{1}{F(\mathbf{v}_1)}$. Note that since t is chosen a non-negative number if $t_*<0$ there is no intersection with the ray and the shape. And if $t_*\geq 0$ then the ray intersects with the shape and the intersection point is

$$\mathbf{v}_* = \frac{\mathbf{v}_1}{F(\mathbf{v}_1)}. \tag{7}$$

Recall that a shape is called a star if there exists a vector $\mathbf{v}_0$ such that every ray originating from this vector $\mathbf{v}_0$ intersects the boundary of the shape *at most at one point*. Since each ray starting from the origin intersects the border of a ray-linearly represented shape at most at one point, ray-linearly represented shapes are star shapes with center point origin, $\mathbf{0}$.

It is easy to show that the position of the intersection points depends only on the directions of the vectors. Therefore, in order to find intersection points in any given direction it is enough to provide only one vector. If we can describe a parametric equation of a vector family that gives only one vector for each direction, we can convert ray-linear implicit representations into parametric representations. Then, computation of any ray-linearly represented shape simplifies from finding roots in three-dimensions to evaluation of a parametric equation.

[5] Let $\alpha=-1$, then any homogeneous function F with $m=1$ has to satisfy $F(-1\times\mathbf{V})=-1\times F(\mathbf{V})$. But absolute value function does not satisfy this property.

Definition 3 A parametrically represented shape $S=\cup_{i=1}^{n} S_i$ is the one that is represented by by a set of parametric equations, $\mathbf{f}_i:\mathcal{P}\longrightarrow\mathcal{V}$, where

$$S_i = \{\mathbf{v} = \mathbf{f}_i(\mathbf{s}) \mid \forall \mathbf{s} \in \mathcal{P}\}, \tag{8}$$

and $\mathcal{P}$ is a given parameter space. This parametrically represented shape S is a *bounded star shape* if any ray originating from $\mathbf{0}$ intersects the shape S exactly at one point.

Since any ray originating from $\mathbf{0}$ intersects a bounded star shape S exactly at one point, the difference of intersection points and $\mathbf{0}$ give us the equation of a vector family. In other words, we can use the parametric equation of a bounded star shape as a parametric equation of vector family for parameterization of ray-linear implicit representations.

Definition 4 A parametrically represented bounded star shape S is called a *guide shape* if it is used for parameterization of ray-linearly represented shapes.

Note that for parameterization of solid shapes, the guide shapes will be surfaces.

A natural example of a guide surface is a sphere. A parametric equation of a sphere is $x=\sin\theta\sin\psi$, $y=\cos\theta\sin\psi$, and $z=\cos\psi$. Using this parametric equation of a sphere in equation 7 we find the following parametric equations for any ray-linearly represented shape: $x=(\sin\theta\sin\psi)/F(\sin\theta\sin\psi,\cos\theta\sin\psi,\cos\psi)$, $y=(\cos\theta\sin\psi)/F(\sin\theta\sin\psi,\cos\theta\sin\psi,\cos\psi)$, and $z=\cos\psi/F(\sin\theta\sin\psi,\cos\theta\sin\psi,\cos\psi)$.

This parametric equation of a sphere is not a good choice since it generates congestions at both poles. We use icosahedra or cubes as guide surfaces since they give better distributions. When polyhedra with planar faces are used as guide shapes, the parametric equations becomes simpler. For instance, let the i^{th} square face of a cube be given by four vectors: $\mathbf{v}_{i,0,0}$, $\mathbf{v}_{i,0,1}$, $\mathbf{v}_{i,1,1}$, $\mathbf{v}_{i,1,0}$ for $i=1,2,\ldots,6$. The parametric equation for the i^{th} face is $f_i(u,v)=\mathbf{v}_{i,0,0}(1-u-w)+\mathbf{v}_{i,0,1}w+\mathbf{v}_{i,1,0}u$, where $u,w\in[0,1]$. Using these parametric equations of the faces of a cube in equation 7 we find the following parametric equation of the boundary of a given ray-linear shape
$S=\cup_{i=1}^{6} S_i$, where
$S_i = \{\mathbf{v}(u,w) = f_i(u,w)/F(f_i(u,w)) \mid \forall u,w \in [0,1]\}$.