INTERACTIVE COMPUTATION OF RAY-QUADRIC SURFACES

Ergun Akleman

Visualization Laboratory,
216 Langford Center,
College of Architecture
Texas A&M University
College Station, TX 77843-3137
ergun@viz.tamu.edu

ABSTRACT

Ray-quadrics appear to be a powerful technology for shape modeling. This paper shows how some of the mathematical obstacles to the use of ray-quadrics can be addressed and presents a prototype shape modeling system based on ray-quadrics.

Our contributions in this paper are the following:

1. A new formulation for ray-quadrics, relating them to univariate quadratic polynomials. The new formulation simplifies the development of software implementation.

2. An intuitive method for polygonizing ray-quadric surfaces.

1. INTRODUCTION

The key problem in developing a sculptor-friendly shape modeling system comes from the restrictions imposed by the commonly used tensor-product B-spline parametric forms. Since B-splines can provide interactive shape construction and easy implementation, they are extremely attractive for the designers of solid modeler designer systems. However, since the control point mesh of tensor-product B-splines must be organized as a regular rectangular structure, they can only support surfaces with limited topology. For modeling surfaces of arbitrary topology with B-splines a number of techniques have been advanced. Forsey and Bartels proposed Hierarchical B-Splines [9], Loop and DeRose introduced S-patches for generalization of B-spline surfaces to support arbitrary topologies [13], Loop later introduced quad-nets to refine irregular meshes [14] and Grim and Hughes used manifolds for modeling surfaces of arbitrary topology [11]. Catmull and Clark Subdivision surfaces offers an non-analytical alternative, and overcomes the topological restrictions of the tensor product B-splines [8, 12, 15]. Other analytic alternatives for representing arbitrary topologies include implicit representations such as Soft-Objects [22], Constructive Solid Geometry [20, 19], F-representations [17], Hyperquadrics [10], Ray-quadrics [2] and others. The major obstacle for some implicit representations lies in computation time. The problem is that more complicated shapes require a larger system of implicit formulas. This usually results in more computations due to the increased size of the implicit formula. Ray-quadrics is not an exception. A general ray-quadric formula can be extremely hard to compute. However, for a certain subset of ray-quadrics the computation time remains constant, completely independent of shape complexity [1]. Moreover, this constant computation time is small enough to permit interactive shape construction. Thus, a subset of ray-quadrics guarantees interactive computation.

Another advantage of ray-quadrics is that their topological limitations are different from those of tensor product B-splines. Ray-quadrics can only represent surfaces that satisfy the ray-quadric condition. A surface satisfies the ray-quadric condition if there exists a point such that every ray originating from this point intersects the surface at *at most two* points. This condition can be considered an important compromise for many shape modeling applications. However, for some other applications such as sculpting faces, ray-quadrics can be extremely useful. Faces have just such handles (or holes) - the mouth, nostrils, eyes and so on - and can be topologically represented by toroids with several handles. If we exclude the ears, for most faces it is possible to find a point such that every ray originating from this point intersects the face at *at most two* points, which is exactly like a surface that satisfies the ray-quadric condition.

Ray-quadrics had two earlier problems that limited their use. First, the mathematical description of ray-quadrics was too complicated. Ray-quadrics are a subset of a class of functions called ray-polynomials. We have recently developed a more intuitive mathematical formulation which relates ray-polynomials with univariate quadratic equations. The new formulation based on ray-polynomials simplifies working with ray-quadrics and the development of software

based on ray-quadrics. Second, polygonization of ray-quadric surfaces was not intuitively simple. By introducing the concept of dummy intersection points, the polygonization process has become more intuitive. We have developed prototype software for interactive shape modeling based on these new approaches.

The paper is organized as follows. In the next section, the descriptions of ray-polynomials and ray-polynomial shapes are introduced. We also introduce ray-quadrics as subsets of ray-polynomials. In addition, we briefly explain the topological properties of ray-quadrics and how to construct ray-quadric shapes. The subsequent section is the discussion of the polygonization of ray-quadric surfaces. We conclude with the presentation of a prototype shape modeling system and discussion of its restrictions.

2. DESCRIPTION OF RAY-POLYNOMIALS

To describe ray-polynomials we first describe ray-constants. Let $\mathbf{v}$ be a vector in $\Re^3$, then a ray-constant A is a function from $\Re^3$ to $\Re$ that satisfies the condition $A(\mathbf{v}) = A(\alpha\mathbf{v})$ where α is any positive real number. In other words, ray-constants are functions only of the orientation of the vectors, $\mathbf{v}$.[1] A simple example of a ray-constant is the cosine of the angle it makes with a vector, $\mathbf{v}_0$, which is given by $\frac{\mathbf{v}_0 \bullet \mathbf{v}}{|\mathbf{v}_0||\mathbf{v}|}$. Let g be any function from $\Re$ to $\Re$, then it is easy to show that if $A(\mathbf{v})$ is ray constant then so is $g(A(\mathbf{v}))$. In this paper, capital letters A, B, C and D will be used to denote ray-constants.

The description of ray-polynomials is based on ray-constants. Let X denote the length of $\mathbf{v}$, then a function F from $\Re^3$ to $\Re$ will be called a ray-polynomial if it has the form

$$F(\mathbf{v}) = \sum_{n=0}^{N} A_n(\mathbf{v}) X^n(\mathbf{v}) = \sum_{n=0}^{N} A_n X^n. \qquad (1)$$

The function F will be called ray-quadric if $N = 2$, ray-affine or ray-linear if $N = 1$. Note that the usage of X instead of $|\mathbf{v}|$ helps to simplify the notation and emphasize the univariate structure of ray-polynomials.

We must also introduce the definitions of ray-polynomial solids $\mathcal{V}(F)$ and boundaries of ray-polynomial solids (Ray-polynomial surfaces) $\mathcal{S}(F)$ by relating them to their ray-polynomial implicit equations as

$$\mathcal{V}(F) = \{\ \mathbf{v}\ \mid\ F(\mathbf{v}) \leq 0\ \},$$

and

$$\mathcal{S}(F) = \{\ \mathbf{v}\ \mid\ F(\mathbf{v}) = 0\ \}.$$

[1]Ray-constant is really a function on the unit sphere. Since there is no requirement, the value of a ray-constant function can change drastically across the unit sphere. Moreover, the function is not continuous at origin 0. Unless it is clearly stated, we presume that the value of a ray-constant function at origin is a positive number.

If F is ray-quadric or ray-affine these shapes will be called ray-quadric or ray-affine solids or surfaces.

This paper focuses mainly on the problem of polygonization of ray-quadric surfaces. To explain the polygonization process using a familiar notation, instead of the general ray-quadric form, a restricted form is used[2]

$$F(\mathbf{v}) = X^2 - 2BX + C. \qquad (2)$$

For ray-affines we also use another simpler form

$$F(\mathbf{v}) = X - A. \qquad (3)$$

These forms give a geometrically intuitive explanation of the shapes. For instance, $\mathcal{V}(X - A)$ means the set of vectors whose lengths are no longer than $A(\mathbf{v})$ and $\mathcal{S}(X - A)$ means the set of vectors whose lengths are equal to $A(\mathbf{v})$. The same type of geometric intuition can also be used for ray-quadric shapes. For instance, $\mathcal{V}(X^2 - 2BX + C)$ means the set of vectors whose lengths are between $B - D$ and $B + D$ where $D = \sqrt{B^2 - C}$. There is an additional advantage to choosing these forms. $\mathcal{V}(X-A)$ and $\mathcal{V}(X^2-2BX+C)$ are guaranteed to be bounded solids when the values of $A(\mathbf{v}), B(\mathbf{v})$ and $C(\mathbf{v})$'s are bounded.

2.1. Topology of Ray-Quadric Surfaces

It is helpful to first examine ray-affine surfaces. The topology of ray-affines comes directly from their description; every ray originating from the origin will intersect ray-affine surface $\mathcal{S}(X - A)$ at at most *one* point. Based on this observation it is possible to show that $\mathcal{V}(X - A)$ is a star shape and $\mathcal{S}(X - A)$ is the boundary of a star shape. Stars are not as restrictive as they sound. Many shapes around us are star shapes. For instance, many faces can be approximated as star shapes. Construction of shapes resembling human faces is possible by using an earlier interactive shape modeling system based on ray-affines [1].

The claim in the previous paragraph is easily shown to be correct. A ray starting from the origin can be given by the direction of a vector. Since $A(\mathbf{v})$ is a ray-constant and has a unique value for any given direction, the equation $X = A$ has *at most one* solution.[3] It is important to note that if $A(\mathbf{v})$ is negative there will be no solution, since X is the length of the vector, which should be a non-negative real number.

Ray-quadrics are more expressive than ray-affines in the sense that all ray-affines are ray-quadrics, but the reverse is not true; every ray originating from the origin intersects ray-quadric surfaces at at most *two* points. Ray-quadrics include certain toroids with any

[2]By using these forms, we only lose the ray-linears which are given by AX.

[3]Note that we presume that the value of a ray-constant function at origin is a positive number. Therefore, zero vector will not be a solution.

number of handles. For instance, a ray-quadric face can have a mouth opening or nostrils but a ray-affine face cannot.

It is also easy to show why every ray originating from the origin intersects ray-quadric surfaces at at most *two* points. For a given direction, the equation 3 has two roots

$$X_+ = B + D \quad \text{and} \quad X_- = B - D$$

where $D = \sqrt{B^2 - C}$. For simplification, we will denote $X_\pm = B \pm D$, to be the set $\{X_+, X_-\}$. If both roots are positive reals, then two intersections exist. If both are negative or complex, there is no intersection. If $X_+ = X_-$ there is only one intersection. It is also true that there will be only one intersection when one root is positive and the other one is negative. However, this will not occur as long as $C(\mathbf{v})$ is positive.

2.2. Construction of Ray-Quadric Representations

The set operations over ray-affine solids can be implemented by functional composition of ray-constants which result in ray-quadrics [2, 1]. These operations are easier to explain by using the new formulations. Let $\mathcal{V}(X-A_1)$ and $\mathcal{V}(X-A_2)$ be two star solids, then

$$\mathcal{V}(X-A_1) \cap \mathcal{V}(X-A_2) = \mathcal{V}(X - min(A_1, A_2)), \quad (4)$$

$$\mathcal{V}(X-A_1) \cup \mathcal{V}(X-A_2) = \mathcal{V}(X - max(A_1, A_2)), \quad (5)$$

$$\mathcal{V}(X-A_1) - \mathcal{V}(X-A_2) = \mathcal{V}((X-A_1A_3)(X-A_2A_3)), \quad (6)$$

where $A_3 = sign(A_1 - A_2)$, with $sign(x) = -1$ if $x < 0$, $sign(x) = 1$ otherwise. These operations are illustrated in the Figure 1. As shown in the Figure 1 $sign(A_1 - A_2)$ term transforms symmetric set difference operations into a set difference operation by eliminating the roots when $A_1 < A_2$.

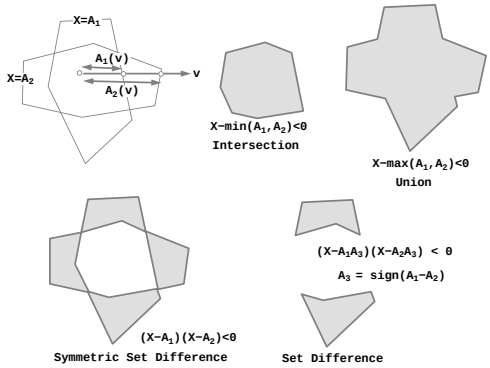

Figure 1: Set operations over star shapes that is given by ray-affines.

The sharp edges and corners resulting from exact set operations can be smoothed out by changing some blending parameters of the functional operators. These smoothing operations, which are also called approximate set operations, resemble real shape modeling operations. For instance, the intersection of a solid with a half-space is like cutting that solid with a knife. Smoothing the edges by using an approximate set-operation resembles sanding the sharp edges. These approximate operations can easily be obtained by exchanging the max and min operations with Ricci's approximate set operations [20] and adding a constant term into set-difference operation

$$\mathcal{V}(X-A_1) \cap_p \mathcal{V}(X-A_2) = \mathcal{V}(X - \sqrt[-p]{A_1^{-p} + A_2^{-p}}), \quad (7)$$

$$\mathcal{V}(X-A_1) \cup_p \mathcal{V}(X-A_2) = \mathcal{V}(X - \sqrt[p]{A_1^{p} + A_2^{p}}), \quad (8)$$

$$\mathcal{V}(X-A_1) -_r \mathcal{V}(X-A_2) = \mathcal{V}((X-A_1A_3)(X-A_2A_3) - r) \quad (9)$$

where blending parameters p and r are positive real numbers that control smoothing.

Another useful operation on $\mathcal{V}(X-A_1)$ and $\mathcal{V}(X-A_2)$ is called *boundary intersection and flesh* operation. Intersection of boundaries of solids (surfaces) that result in space curves can also be described by functional composition of ray-constants

$$\mathcal{V}((X-A_1)^2 + (X-A_2)^2). \quad (10)$$

This equation is notoriously unstable, since the zeroes of $(X-A)^2$ are all double roots and are hence subject to floating point errors that make them disappear. However, the equation 10 is extremely useful since these space curves represented by the equation serve as a skeleton of solids given by

$$\mathcal{V}((X-A_1)^2 + (X-A_2)^2 - r^2). \quad (11)$$

where r is a positive real number that creates a flesh around the skeleton curve.

3. POLYGONIZATION OF RAY-QUADRIC SURFACES

To provide fast rendering, we first polygonize our ray-quadric surfaces. Since the polygonization must be done during the construction process, it must be fast to provide interactive construction to the user. There exist many general methods for polygonization of implicit surfaces Bloo88,Over93,Bloo95,Hart97. However, because of special nature of ray-quadrics, fast polygonization of ray-quadric shapes requires a special method.

One of the earlier problem in polygonization was the computation of the ray-constant terms coming from approximate union and intersection operations. It was shown that these terms can be computed in constant time if union and intersection operations are organized [1]. This organization is briefly explained in the next section. A ray-affine shape modeling system for interactive construction of star surfaces by

union and intersection operations was developed earlier based on fast computation of these terms [1].

Computation of ray-constant terms for set-difference and flesh operators, which are given by equations 9 and 11, is simple. However, there is a completely different problem in this case: As a result of these two operations ray-quadric surfaces which are toroidal surfaces with several handles are constructed. To support interactive construction by these operators, our polygonization process should support ray-quadrics. Since the resulting equations of set-difference and flesh operators can be transformed into equation 3 the equation $X^2-2BX+C$ will be used to discuss the polygonization of ray-quadric surfaces. The polygonization process will be explained by using the simplest case where $B \geq D \geq 0$ with $D = \sqrt{B^2 - C}$. This condition guarantees that there are always two intersections with ray-quadric surfaces.

3.1. Polygonization for the Case $B \geq D \geq 0$

The assumption that $B \geq D \geq 0$ guarantees both roots of equation 3, $X_+ = B + D$ and $X_- = B - D$, are positive real and $X_+ \geq X_-$. In other words, there will always be intersections with a ray starting from the origin, and the order of the intersection points will never change over the surface.

3.1.1. Projection of a Point onto Ray-Quadric Surface

Let a direction be given by a vector $\mathbf{v}_i$ and $\mathbf{n}_i = \mathbf{v}_i/|\mathbf{v}_i|$ be the unit vector in the direction of $\mathbf{v}_i$. In addition, let $\mathbf{v}_{i_+}$ and $\mathbf{v}_{i_-}$ denote intersection points with the ray-quadric surface along the ray from the origin in the direction of $\mathbf{v}_i$. For simplification, we will denote $\mathbf{v}_{i_\pm}$ to be the set of intersection points $\{\mathbf{v}_{i_+}, \mathbf{v}_{i_-}\}$. Then the intersection points can be simply computed by multiplying the unit vector $\mathbf{n}_i$ in the direction of $\mathbf{v}_i$ with the roots $X_\pm$

$$\mathbf{v}_{i_\pm} = \mathbf{n}_i X_\pm = \mathbf{v}_i \frac{B \pm D}{|\mathbf{v}_i|}.$$

It is possible to view this operation in two different ways. One interpretation is the intersection of a ray in the direction of $\mathbf{v}_i$ with the ray-quadric surface. The other interpretation is the projection of a point given by $\mathbf{v}_i$ onto the ray-quadric surface as shown in Figure 2. The second interpretation is useful since it makes it possible to present a simple explanation of polygonization of ray-quadric surfaces. Let us first imagine projecting a triangle onto a ray-quadric surface.

3.1.2. Projection of a Triangle onto a Ray-Quadric Surface

Let a triangle T be given by an ordered set of three vectors $T = \{\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k\}$. We call this triangle a *guide triangle*. A triangle must satisfy three conditions to be used as guide triangle.

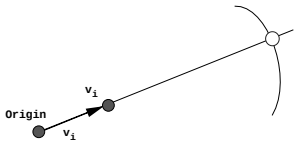

Figure 2: Finding an intersection point or projection of a point on a ray-quadric surface.

- The order of the vertices of the guide triangle must guarantee that its surface normal vector points away from the origin. In other words, the vertices of the triangle must satisfy an inequality such as $(\mathbf{v}_i - \mathbf{v}_j \times \mathbf{v}_j - \mathbf{v}_k) \bullet \mathbf{v}_i > 0$.
- The vertex directions must satisfy the inequality $\mathbf{n}_i \neq \mathbf{n}_j \neq \mathbf{n}_k$. This condition guarantees that the triangle will not reduce to a line or a point when it is projected.
- The solid angle of the triangle that is seen from the origin must be small. This condition is necessary in order to get a good approximation of a ray-quadric surface.

By using the procedure in the previous section, we can project a guide triangle onto the ray-quadric surface and obtain two triangles which we call projection triangles $T_+ = \{\mathbf{v}_{i_+}, \mathbf{v}_{j_+}, \mathbf{v}_{k_+}\}$ and $T_- = \{\mathbf{v}_{k_-}, \mathbf{v}_{j_-}, \mathbf{v}_{i_-}\}$. Since in this case *the inside* of the ray-quadric solid is between two projection triangles T_+ and T_- the order in the second triangle has to be reversed to get the correct normal vector. The projection of a guide triangle is illustrated in Figure 3. It is simple to generalize the guide triangle idea into a guide polyhedron idea.

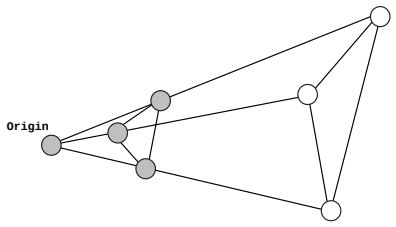

Figure 3: Guide and projection triangles.

3.1.3. Projection of a Guide Polyhedron onto a Ray-Quadric Surface

Let a star polyhedron be given and the origin be the center point of this star shape. Let us also assume that each face of the star polyhedron is transformed into a set of triangles. Then, the polygonization of ray-quadric surface can be easily done by projecting all of the triangles of the star polyhedra onto ray-quadric shape. We call this star polyhedron guide

polyhedron. In practice, instead of general star polyhedra, we use regular convex polyhedra such as a cube or an icosohedron. No doubt The faces of the cube or icosohedron are tesselated into smaller triangles as required to get better tesselation of the ray-quadric surface.

3.2. Polygonization for General Case

In the general case where the condition $B^2 \geq C \geq 0$ does not necessarily hold, it is possible for roots to be negative, complex or change across the surface. If both roots are negative or complex, there will be no intersection with the ray-quadric surface in the direction of the given vector $\mathbf{v}_i$. In this situation we can say that $\mathbf{v}_i$ cannot be projected onto the ray-quadric surface. If none of the three vertices of a guide triangle can be projected, the usage of this guide triangle will simply be eliminated. However, if one or two vertices but not all three can be projected, the polygonization becomes complicated and a special process has to be developed.

In the case when only one or two vertices can be projected, the problem becomes determination of the directions that give only one intersection, as shown in the 2D example in Figure 4. This determination requires changing the guide polyhedron into a new one as also shown in Figure 4. To change the guide polyhedron, each guide triangle with one or two unprojected vertices has to be transformed into a new set of guide triangles. The procedure for changing a guide triangle into a set of new guide triangles is shown in Figure 5. This procedure is based on the creation of dummy intersection points when there is no intersection. Using these dummy intersection points, the guide triangles can be transformed onto the ray-quadric surface. As explained below, the dummy intersection points are chosen so that their order is reversed from neighboring intersection points. Thus, the projection triangles will intersect with each other. The corners generated by this intersection are used to describe new guide triangles as shown in Figure 5. Note that the dummy intersection points are thrown away once the new guide vertices are found.

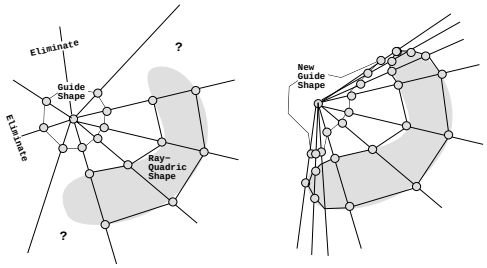

Figure 4: Determination of directions that give only one intersection.

Determination of the dummy intersection points

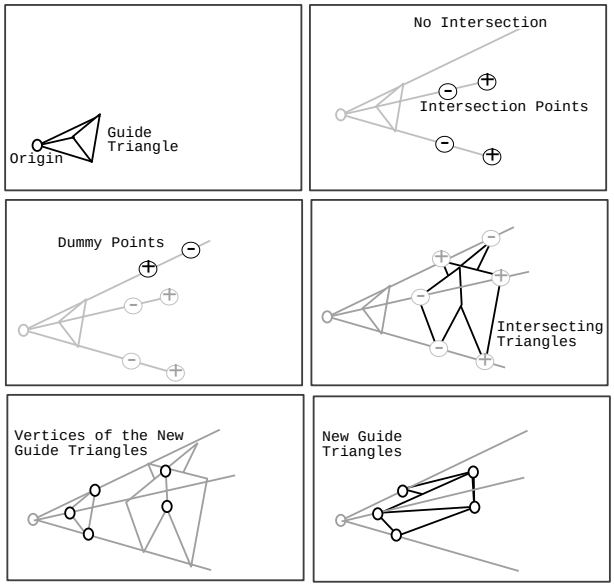

Figure 5: Determination of new guide triangles

is the crucial step in this process. Computation of dummy intersection points depends on whether roots of $X^2 - 2BX + C$ are complex or negative.

- If $B^2 - C$ is negative, the roots will be complex. This case will occur when applying both set-difference and flesh operators given by equations 9 and 11. In this case, a dummy $D = \sqrt{B^2 - C}$ value is computed as $D_{dm} = -\sqrt{C - B^2}$. If B is positive dummy roots X_{dm_-} and X_{dm_+} are given $X_{dm_\pm} = B \pm D_{dm}$. Note that since $X_{dm_-} \geq X_{dm_+}$, the order of dummy intersection points will be reversed.

- If B is negative and $B^2 - C$ is positive, roots will be negative real. This case will occur only when applying set-difference operator given by equation 9. In the set-difference operation if the term $sign(A_1 - A_2)$ is negative B will be negative. In our applications both A_1 and A_2 are non-negative, therefore, $B^2 - C$ will be always positive. In this case, dummy roots are computed as $X_{dm_\pm} = -B \mp D = -X_\pm$. The effect of this computation is again reversing the order of roots of a shape resulting from symmetric set difference operation.

The procedure described above is applied until the distance between dummy points is smaller than a chosen threshold. Once we reached that threshold, a point in between two dummy points will be used as an intersection point. The algorithm explained in [1] uses old values of ray-constants in order to compute new ones in a given direction $\mathbf{v}$. Therefore, the refinement of guide triangles to achieve high quality can be costly, since refinement requires computation of values of ray-constants for new directions. To avoid this

computation, the system computes only one iteration in interactive mode. The resulting shapes are still of good quality. All images in this paper are obtained using only one iteration.

Since the concept of dummy intersection points makes the process geometrically intuitive, we think that the new process is appealing for programming.

4. PROTOTYPE SYSTEM

We have recently developed a new prototype shape modeling system based on results discussed in previous sections. Using this new system, ray-quadric surfaces can be interactively constructed with entry level graphic workstations such as SGI O2.

In this system, constructions using union and intersection are inherited from the earlier ray-affine shape modeling system, which is mentioned in the previous section. Since shapes resembling faces can be represented by star shapes, it was possible to interactively construct shapes resembling human faces by using that ray-affine system. In the new system the user specifies two star shapes instead of one.

4.1. Star Surface Construction and Its User Interface

In the system, users specify control shapes by using exact set operations over half-spaces. These half spaces are given by the ray-quadric plane equation $X - \frac{a_i|\mathbf{v}|}{\mathbf{n}_i\mathbf{v}}$, where a_i is the distance to the origin and $\mathbf{n}_i$ is the unit normal to the planar surface. The user describes these half-spaces by first drawing convex polygons on each of two perpendicular planar surfaces. These perpendicular planar surfaces are either the planes x=0 and y=0 or x=0 and z=0. Each convex polygon actually 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 polyhedral control shape. By changing the blending parameter p of approximate intersection operation, the user can generate a smoothed out convex shape as shown in Figure 6.

If the user get exact unions of these convex polyhedra, the resulting shape becomes a star polyhedron with a center point origin. The user can smooth out sharp corners and edges resulting from exact set operations by changing the blending parameter p of the approximate union and intersection operations. We view this star polyhedron as a ray-linear surface control shape. [4]

[4]If this star polyhedron resulting from exact set operations is used as guide shape for the polygonization of approximate shapes, errors from polygonization will be bounded.

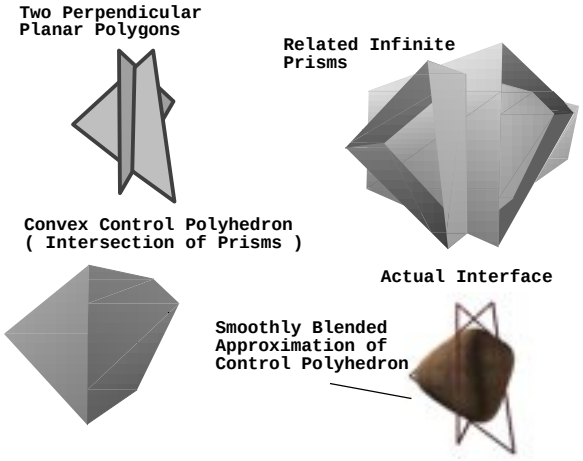

Figure 6: Usage of infinite prisms to describe convex control polyhedron.

4.2. Ray-Quadric Surface Construction

Ray-quadric surfaces are constructed by two star polyhedra. Smooth approximations of each star polyhedron give smoothly blended star shapes. Set-difference and flesh operators over two smoothly blended star shapes gives ray-quadric surfaces. These operations are given by equations 9 and 11.

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 the program this manipulation is accomplished by simply moving the vertices of the polygons. The example in Figure 7 shows the control shape, set-difference, boundary intersection and flesh operations. All solids that are shown in Figure 8 are ray-quadric solids which are constructed using union, intersection and set-difference operations and then smoothing out sharp edges resulting from the exact set operations. The shapes in Figure 9 are constructed with boundary intersection and flesh operations.

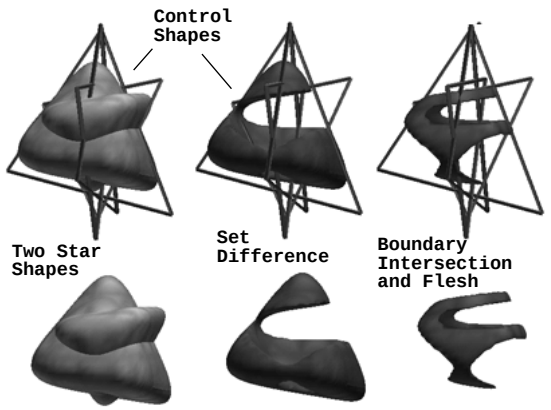

Figure 7: Ray-quadric solid construction.

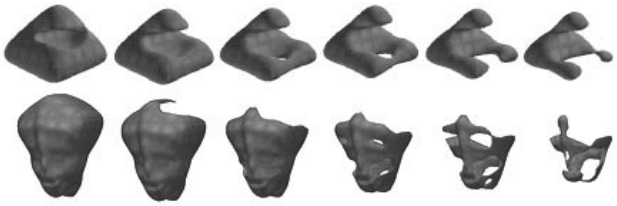

Figure 8: Effect of parameter r in set difference operations.

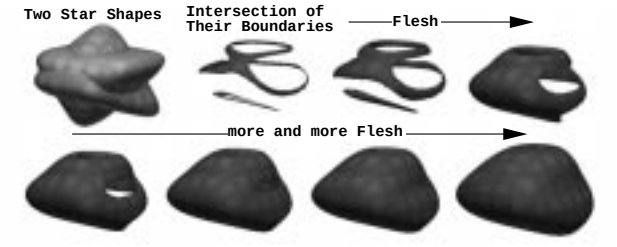

Figure 9: Effect of flesh parameter r in boundary intersection and flesh operations.

5. DISCUSSION AND FUTURE WORK

In this work, we develop a simple formulation for ray-polynomials and provide a simple method for polygonization of ray-quadric surfaces. We also develop a prototype ray-quadric shape modeling system to show the viability of ray-quadrics as a computer aided shape modeling tool. However, this prototype system is extremely limited and does not provide the full range of possibilities of ray-quadrics [2].

5.1. Limitations of the Prototype User-Interface

One of the limitations in the system comes from construction methodology. First, convex shapes are obtained using intersections. Then, star shapes are constructed by using unions. The order does not have to be this way. However, by choosing this order the computation price is considerably reduced.

In the general case, the computational price for determining the values of ray-constant functions A's can be linearly dependent on the total number of half-spaces $\mathcal{N}$. This can be a big problem when the number of half-spaces is large. By using our order of operations and some bookkeeping explained in detail in [1] this price is greatly reduced into a constant during the construction process.

Some of the restrictions of user interface could be removed in the future. For instance, one of the restrictions is that the system provide only a certain type of convex control shape: those that are given as a union of convex polyhedra that are constructed by the union of two perpendicular prisms. This restriction could be removed by describing convex shapes as either a convex hull of a set of points or an intersection of the half-spaces which can be manipulated freely in three-dimensions with virtual reality tools.

Another restriction that can be removed comes from the building blocks themselves. These building blocks do not have to be half-spaces and the faces of the control shapes do not have to be planar. For instance, a cylinder could be part of a control solid since it is the exact intersection of an infinite cylinder and two half-spaces. Removing this restriction requires a different construction methodology which permits approximate intersection of constructed convex shapes. Removing this restriction would again require addressing the linearly dependent computation price for ray-constants.

Current interface cannot support ray-linear surfaces which are given as $\mathcal{S}(AX)$ since they are captured neither by $\mathcal{S}(X - A)$ nor by $\mathcal{S}(X^2 - 2BX + C)$. Some applications may require ray-linear surfaces. For instance, regular toroid is constructed by flesh operator over an infinite cylinder (a ray-affine surface) and a plane passing thorough origin (a ray-linear surface).

Current interface has another problem. When the number of polygons is high, it becomes difficult to choose and move the points in 3D space. For specific applications, usage of projections can be useful. Moreover the system requires a great deal of knowledge about ray-quadrics. A sculptor, generally, does not want to know all the restrictions.

5.2. Future Work

Despite its limitations, this prototype ray-quadric shape modeling system shows the viability of ray-quadrics as a computer aided shape modeling tool. Its user interface can be improved into a more intuitive one by restricting ourselves to special kind of objects such as faces.

We have shown that feature based templates can provide an intuitive user interface for facial modeling. [3]. Set-operation based construction of ray-quadrics solids is suitable for developing feature based user interfaces for modeling faces. In such a system, the sculptors can simply draw the polygonal facial features for front and side views of the face and the system can automatically construct the facial solid from the front and side views as shown in the Figure 10. Development of such a system requires methods to find the control shapes from described features. As a future work, we are planning to develop a ray-quadric based facial modeler.

When ray-quadrics are used in facial modeling, they can provide additional advantage. Facial data may be greatly reduced by using Ray-quadric representations. The user can save the data as a template. The template will actually be the coefficients of the ray-quadrics equations. Seventy integer coefficients between 0-128 are enough to represent the face shown in Figure 10. These coefficients are extremely robust;

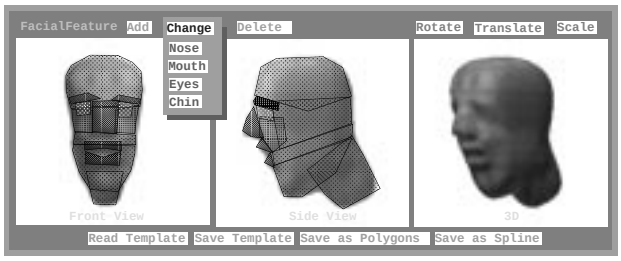

Figure 10: Feature Based User Interface for Facial Modeler.

in other words, the resulting shape is only slightly effected by small changes in the coefficients [1].

6. ACKNOWLEDGMENTS

We would like to thank to Donald House, John Ferguson, Lori Cagle and refeerees for their helpful suggestions. We also thank Dietmar Saupe for the initial spark for the new formulation.

7. REFERENCES

[1] E. Akleman, "Interactive Construction of Smoothly Blended Star Solids", *Proceedings of Graphical Interface'96*, May, 1996.

[2] E. Akleman, "Ray-Quadrics", *Proceedings of Implicit Surfaces'96*, Oct., 1996.

[3] E. Akleman, "Making Caricatures with Morphing", *ACM Siggraph Visual Proceedings*, Aug., 1997.

[4] J. I. Blinn, "A Generalization of Algebraic Surface Drawing", *ACM Transaction on Graphics,* vol 1, no. 3, pp. 235-256, 1982.

[5] J. Bloomenthal, "Polygonization of Implicit Surfaces", *Computer Aided Design*, No. 5, pp. 341-355, 1988.

[6] J. Bloomenthal and K. Ferguson, "Polygonization of Non-Manifold Implicit Surfaces", *Computer Graphics*, vol 29, no. 4, pp. 309-316, 1995.

[7] E. Catmull, "Making an Animation Studio", *Talk presented in Texas A&M* , April 1997.

[8] E. Cattmull and J. Clark. "Recursively Generated B-spline Surfaces on Arbitrary Topological Meshes" *Computer Aided Design*, vol 10, no. 6, pp. 350-355, 1978.

[9] D. Forsey and R. Bartels, "Hierarchical B-Spline Refinement", *Computer Graphics*, vol 22, no. 4, pp. 205-212, 1988.

[10] A. Hanson, "Hyperquadrics: Smoothly Deformable Shapes with Convex Polyhedral Bounds", *Computer Vision, Graphics and Image Processing*, vol 44, no. 1, pp. 191-210, 1988.

[11] . C. M. Grimm and J. F. Hughes, "Modeling Surfaces of Arbitrary Topology using Manifolds", *Computer Graphics*, vol 29, no. 4, pp. 359-368, 1995.

[12] . M. Halstead, M. Kass, and T. DeRose, "Efficient, Fair Interpolation Using Catmull-Clark Surfaces", *Computer Graphics*, vol 27, no. 4, pp. 35-44, 1993.

[13] C. Loop and T. DeRosa, "Generalized B-Spline Surfaces with Arbitrary Topology", *Computer Graphics*, vol 24, no. 4, pp. 347-356, 1990.

[14] C. Loop, "Smooth Spline Surfaces over Irregular Meshes", *Computer Graphics*, vol 28, no. 4, pp. 303-310, 1994.

[15] A. Nasri, "Polyhedral Subdivision Methods for Free Form Surfaces", *ACM Transactions on Graphics* , vol 8, no. 1, pp. 89-96, 1987.

[16] C. W. A. M. van Overveld and B. Wyvill "Shrinkwrap: An Adaptive Algorithm for Polygonization an Implicit Surface", *The University of Calgary, Department of Computer Science Research Report No. 93/514/19,* March 1993.

[17] A. Pasko, V. Adzhiev, A. Sourin and V. Savchenko, "Function Representation in Geometric Modeling: Concepts, Implementations and Applications", *Visual Computer,* no. 11, pp. 429-446, 1995.

[18] J. Peters, " Curvature Continuous Spline Surfaces Over Irregular Meshes", *Computer Aided Geometric Design,* vol 13, 1996.

[19] A. A. G. Requicha and H.B. Voelcker, "Solid Modeling: A Historical Summary and Contemporary Assessment", *IEEE Computer Graphics and Applications,* vol 2, no. 2, pp. 9-24, March 1982.

[20] A. Ricci, "A Constructive Geometry for Computer Graphics", *The Computer Journal,* vol 16, no. 2, pp. 157-160, May 1973.

[21] . B. T. Stander and J. Hart, "Guaranteing Topology of Implicit Surface Polygonization for Interactive Modeling", *Computer Graphics*, vol 31, no. 4, pp. 279-286, 1997.

[22] G. Wyvill and A. Trotman, "Ray Tracing Soft Objects", *CG International '90: Computer Graphics Around the World,* Editors: T. S. Chua and T. L. Kunii, pp. 467-476, 1990, Springer Verlag.