Ray-Affine Functions:
A General Dual Form to Describe
Curves, Surfaces and Volumes

by

Ergun Akleman, Larry F. Hodges,
Russel Mersereau

GIT-GVU-92-27
October 1992

Graphics, Visualization & Usability
Center

Georgia Institute of Technology
Atlanta GA 30332-0280
Ray-Affine Functions:
A General Dual Form to Describe Curves, Surfaces and Volumes

by

Ergun Akleman, Larry F. Hodges, Russell Mersereau

GIT-GVU-92-27
October 1992

Graphics, Visualization & Usability Center
Georgia Institute of Technology
Atlanta GA 30332-0280
Ray-Affine Functions: 
A General Dual Form 
to Describe Curves, Surfaces and Volumes

Ergun Akleman
Larry F. Hodges
Graphics, Visualization and Usability Center

Russell M. Mersereau
School of Electrical Engineering

Georgia Institute of Technology
Atlanta, Ga 30332

October 26, 1992

Abstract

In computer graphics modeling, two different forms are used to represent curves and surfaces: implicit and parametric. Functions that can be expressed both in implicit and parametric forms are called dual forms. To date the only known dual forms are monoids and superquadrics. In this paper we introduce a new dual form: ray-affine functions. Ray-affines include both monoids and superquadrics and provide a wide range of other modeling functions including exponentials and sinusoids. Ray-affines are closed under operations that implement morphing, union and interpolation, and smooth approximations of union and interpolation. This feature of ray-affine functions lets the user construct a ray-affine function to model a shape as a smooth approximation of a control shape given by set union or set intersection of shapes defined by simpler ray-affine functions.

1 Introduction

Mathematically modeled curves and surfaces are represented by either parametric or implicit equations. Parametric and implicit equations possess different modeling qualities. Traditionally computer graphics has favored parametric representations over implicit representations. There are two reasons behind this popularity. First, parametric representations are easier to render. Second, piecewise parametric polynomial based tools, such as Bezier curves or B-splines, have been very successful for designing smooth curves. However, for designing smooth surfaces, piecewise parametric polynomial based tools are not versatile. For instance, tensor product Bezier surfaces or B-Splines cannot represent many common shapes and their smooth approximations, such as octahedrons, hexahedrons, spheres or tori. To generate of such shapes with piecewise parametric polynomials, mathematically involved tools have to be developed [LD89], [LD90].

Implicit representations, however, have many advantages in surface design. These advantages include simpler implementation of geometric operations such as union and intersection, inside/outside tests, and functional composition of functions.

Smooth approximations of unions and intersections of implicitly represented shapes have been proposed by Ricci[Ric73] for constructive solid geometry (CSG). In CSG, the shapes (volumes) are given by the implicit inequalities $I(x, y, z) \leq 0$, where $V(x, y, z) \in \mathbb{R}^3$, $I(x, y, z)$ is positive real. Constructive solid geometry does not limit the shapes that can be generated. However, the resulting functions are non-polynomials and not easy to render.

In the last decade, implicit equation research has focused primarily on implicit polynomials [SN86], [Bli90]. However, even implicit polynomials are, in general, not easy to render. Rendering an implicit polynomial with degree higher than two is still a root finding process and it is clearly more difficult in three-dimensional space. An exception are monoids, the only polynomial dual forms. Dual forms are functions that have both implicit and parametric representations. Since monoids have parametric representations, it is possible to directly render them [AB87a], [AB87b], [Hcf90]. However, monoids provide a limited variety of shapes. As a result, the modeling capabilities of implicit polynomials have been of limited usefulness.

Recently more attention has been given to piecewise implicit polynomials [Sed85], [Baj90], [Bai92]. However, when they are used piecewise, implicit polynomials do not neces-
sarily continue to provide implicit function properties. Furthermore, piecewise implicit polynomials are mathematically complex and lose their intuitive appeal for modeling.

An important non-polynomial implicit function family are the superquadrics which were proposed by Barr [Bar81]. Superquadrics are also dual forms, therefore, they provide direct rendering. Barr showed that, with deformation, superquadrics can generate many shapes. However, the major problem with superquadrics is the deformation itself. Deformation is suitable for animation but is not suitable for modeling. For instance, it is almost impossible to find a starting superquadric shape and deformation that go to a tetrahedron or icosahedron.

Figure 1: Two ray-affine shapes, their union and two smooth approximations of the union.

We propose a new non-polynomial family, ray-affine functions. Ray-affine functions are dual forms. Therefore, they provide the property of parametric equations: easy rendering by just plugging in parameter values. In addition, ray-affine functions provide the best features of implicit forms. First of all ray-affine functions can be described by a number of operations which are intuitive, such as the union and intersection operations used in constructive solid geometry. Ray-affine functions not only provide exact union and intersection but also smooth approximations of unions as shown in Figure 1 and intersections as shown in Figure 2. In addition, ray-affine functions are closed under operations that implement morphing over shapes as shown in Figure 3. Ray-affines also provide a good estimate of distance to the shape [Akl92].

Ray-affines are a subset of a much larger class of functions that we call ray-polynomials. Ray-polynomials include all polynomials. As a result almost any shape can be approximated by using ray-polynomials. Since ray polynomials are also closed under the union, intersection and morphing operations, these shapes can be intuitively constructed from simpler shapes. However, since a generic ray-polynomial is not a dual form and can not provide easy rendering, we limit our discussion in this paper to ray-affines.

Figure 3: Morphing of the shapes provided by ray-affines.

Limiting our modeling to ray-affines limits the shapes we can produce. A ray-affine can only generate a ray-affine shape. A shape is ray-affine if there exists a center point such that each ray originating from this point intersects the shape at no more than one point. In other words, using ray-affine functions, toroids or knots cannot be generated, however, smooth approximations to any star or convex polyhedron can be generated. Examples of ray-affine curves are shown in Figure 4. Since star and convex shapes are more frequently used than any other type of shapes and since, if it is necessary, it is always possible to use higher degree ray-polynomials, we do not consider this limitation to pseudo-affine shapes as too restrictive.

Figure 4: Examples of ray-affine curves.

The remainder of this paper is organized as follows. In the second section, we provide the notations to describe both ray-polynomials and ray-affines. In the third section, we provide the definitions of ray-polynomials and ray-affines. Based on the definition of ray-affines, in Section 4 we describe the parameterization of ray-affine functions and develop the concept of a guide shape. In Section 5 examples of ray-affine functions are given. In Section 6 we show how to construct ray-polynomials. In Section 7, set operation properties of ray-affine functions are discussed and examples are given. In Section 8, the properties of the ray-constant part is discussed and a conclusion is given in Section 9.

1 Other non-polynomial implicit function families, such as Blinn's exponential functions [Bl82], soft objects [Wuy90] or convolution surfaces [Blo91], are not directly related to ray-affines.

2 This explains why there is no difference operator. Since the set difference of two ray-affine shape is not a ray-affine shape, ray-affine functions cannot be closed under any difference operator.

3 A star shape is a closed ray-affine shape.

4 For instance, using ray-quadratics, toroids can be generated. For more information, the reader can refer to [Akl92]. Smooth approximation of set difference of two ray-affine shapes can also be given by ray-quadratics.
2 Notation

An n-dimensional vector space, \( \mathcal{V}^n \), will be specified by a set of linearly independent unit vectors, \( v_1, v_2, \ldots, v_n \). A vector \( v \in \mathcal{V} \) on this space will be given by a set of real numbers using brackets: \( v = [a_1, a_2, \ldots, a_n] \), where

\[
v = [a_1, a_2, \ldots, a_n] = a_1 v_1 + a_2 v_2 + \cdots + a_n v_n.
\]

An n-dimensional point space, \( \mathcal{D}^n \), will be given by an origin point \( z_0 \) and the vector space \( \mathcal{V}^n \). A point \( z \in \mathcal{D} \) will be denoted by a set of real numbers using parentheses: \( z = (a_1, a_2, \ldots, a_n) \), where

\[
z = (a_1, a_2, \ldots, a_n) = [a_1, a_2, \ldots, a_n] + z_0.
\]

For \( \mathcal{D}^2 \) and \( \mathcal{D}^3 \) a vector will be denoted \( v = (x, y, z) \), and \( v = (x, y) \) respectively. All functions in this paper are defined from an n-dimensional point space into reals.

An n-dimensional parameter space, \( \mathcal{P}^n \), is a closed subset of the n-dimensional point space. A parameter \( s \in \mathcal{P}^n \) on this space will be given by a set of real numbers using parentheses: \( s = (u_1, u_2, \ldots, u_n) \).

A shape parameter, \( p \), is always a positive real number. A ray will be given by the following parametric equation:

\[
z = f_r(t) = v_r t + z_r
\]

where \( v_r \) gives the direction of ray, \( z_r \) is the starting point and \( t \) is a positive real number. The symbols for points, vectors and rays are illustrated in Figure 5.

![Figure 5: Symbols to be used for points, vectors and rays.](image)

3 Definitions

The concept of ray-affine is based on the observation that some functions are simplified when they are projected onto a region. The following introduction explains this concept.

3.1 Introduction

First observe that distance functions in the form of

\[
F(z) = \sqrt[3]{|x - x_*|^p + |y - y_*|^p}
\]

are simplified into a linear function over all rays originating from the point \((x_*, y_*)\). Any such ray can be represented by the following parametric equation:

\[
z = (x, y) = v t + z_r = (a_1 t + x_*, a_2 t + y_*),
\]

where \( v = [a_1, a_2] \) and \( z_r = (x_*, y_*) \). On this ray, the distance functions become linear with respect to the ray parameter, \( t \):

\[
F(z) = \sqrt[3]{|a_1|^p + |a_2|^p} \ t.
\]

In other words, the distance functions simplify to linear functions if they are projected onto any ray originating from the point \((x_*, y_*)\). Notice that this simplification property is not a property of only distance functions. Several other function families also simplify to linear functions when projected onto rays. We call all of these functions ray-linear. For the general case, ray-linear functions, \( \forall v \in \mathcal{V}^n \) simplify as follows.

\[
F(v t + z_r) = F(v + z_r) t.
\]

If ray-linear functions are equated to zero to describe an implicit equation, we find only one solution: the center point \( z_r \). Ray-linear functions are not useful for shape modeling, but, if we use the implicit form of

\[
F(z) - C = 0,
\]

where \( C \) is a real constant, many different shapes can be obtained. For instance, if \( F(z) \) is a distance function, the implicit equation \( F(z) - C = 0 \), represents the shapes shown in Figure 5.

![Figure 6: The shapes generated by distance functions for p values of 1, 2, 4, 8, \infty.](image)

The function \( F(z) - C \) is an affine function on each ray and, therefore, it is called ray-affine. Note also that \( C \) in \( F(z) - C \) does not have to be constant. Similarly, the constant part can be given with a ray-constant function \( C(z) \), which simplifies into a constant when projected onto a ray. Ray-constants provide a large class of functions from exponentials to sinusoidals as shown in Section 8.

Natural extensions of ray-constants and ray-linear are \( m \)-degree ray-polynomials. In this paper, although we focus on ray-affines, we observe that construction of ray-linear and ray-constants needs all ray-polynomials as shown in Section 6. Therefore, we provide the general definition of \( m \)-degree ray-polynomials. Ray-linear and constants are instances of these \( m \)-degree ray-polynomials.

3.2 Ray-Polynomials

**Definition 1** \( H^{<m>} : \mathcal{D}^n \rightarrow \mathbb{R} \) is a \( m \)-degree ray-polynomial with a center point \( z_r \) iff \( \forall v \in \mathcal{V}^n \) the intersection equation of \( H^{<m>}(z) \) and the ray \( z = v t + z_r \) is in the following form:
\[ H^{<m>}(v + z_r) = A t^m, \]
where \( A = H^{<m>}(v + z_r) \). \( H^{<1>}(z) = F(z) \) denotes ray-linears and \( H^{<0>}(z) = -C(z) \) denotes ray-constants.

Since \( H^{<m>}(v + z_r) \) is constant over the ray, it can be viewed as a coefficient. A polynomial over the ray can be given as

\[ \sum_{m=0}^{M} H^{<m>}(v + z_r) t^m. \]

Such a polynomial, when it is equal to zero, will generate a shape with at most \( M \) intersections with a ray as shown in Figure 7. For the general case there is no analytical solution to find all of these \( M \) intersection points.

### 3.3 Ray-Affine Functions

The summation form of ray-polynomials, for \( M = 1 \), gives the ray-affine function:

\[ H^{<1>}(z) + H^{<0>}(z) = F(z) - C(z). \]

The ray-affine implicit equation is given by,

\[ F(z) - C(z) = 0, \]

as illustrated in Figure 8. There is an analytical solution for all ray-affine implicit equations. Let us find the intersection point \( z_{\text{root}} \) on the ray \( z = v + z_r \). The ray-affine implicit equation on this ray is given by,

\[ F(v + z_r) - C(v + z_r) = 0. \]

As a result of the ray-affine property this equation is simplified:

\[ F(v + z_r) t - C(v + z_r) = 0. \]

Then from the equation, the value of \( t \) at the intersection is found:

\[ t = \frac{C(v + z_r)}{F(v + z_r)}. \]

Using this \( t \) value in the ray equation, the intersection point is obtained:

\[ z_{\text{root}} = \frac{C(v + z_r)}{F(v + z_r)} v + z_r. \]

This last equation is like a parametric equation if we consider \( v \) as the parameter. In the next section, we explain how to describe vectors to parameterize ray-affine implicit equations.

### 4 Parameterization

Note that if two vectors have the same direction, they describe the same ray. Since, on each ray, the ray-affine equation has only one solution, there is no need to use more than one vector in the same direction. If a distinct parameter value is attached to each direction, the ray-affine implicit equation will be automatically parameterized. For this purpose, a vector family must be described.

#### 4.1 Guide Shape

To describe a vector family, we develop the concept of a guide shape. A guide shape is represented by a parametric equation:

\[ z = f(s), \]

where \( s \in \mathcal{P} \) is a parameter. In addition the shape of the guide shape should be ray-affine with a center point \( z_r \). Then, using the parametric equation of the guide shape, a parametric equation of the vector family can be described as,

\[ v = f(s) - z_r. \]

For instance, a unit circle with a center point \( z_r = (x_r, y_r) \) can be given by the following parametric equation:

\[ z = (x, y) = (\sin u + x_r, \cos u + y_r), \]

Then the related vector family is given by the following equation:

\[ v = [\sin u, \cos u], \]

where \( u \in \mathcal{P} \). This vector family provides distinct rays as illustrated in Figure 9. As a result, the parametric equation of the shape described by a ray-affine implicit equation will be

\[ z = \frac{C(f(s))}{F(f(s))} (f(s) - z_r) + z_r. \]

For example, let us parameterize the ray-affine equation of

\[ \sqrt{x^2 + y^2} - C = 0. \]
In this equation \( F(z) = \sqrt{x^2 + y^2}, \) \( C(z) = C \) and the center point of this ray-affine function is the origin, \( z_0. \) A ray family originating from the origin is given by the following equation:

\[
z = vt + z_0 = (\sin u, \cos u)t
\]

If we replace these values in the ray-affine implicit equation, we find:

\[
\sqrt{\sin u^2 + \cos u^2} t - 1 = 0.
\]

Then, the value of \( t \) at the intersection point is

\[
t = \frac{C}{\sqrt{\sin u^2 + \cos u^2}}.
\]

Note that if \( t \) is not positive real, there is no intersection on the ray. Let us assume that \( t \) is found to be positive real, then, using this value of \( t \) in the ray equation, we find the parametric equation:

\[
z = \left( \frac{\sin u}{\sqrt{\sin u^2 + \cos u^2}}, \frac{\cos u}{\sqrt{\sin u^2 + \cos u^2}} \right)
\]

Using only one parametric equation for a vector family creates a problem in three-dimensions. For instance, if the parametric equation of a ray family

\[
x = \sin u \sin vt, \quad y = \cos u \sin vt, \quad z = \cos u \cos vt,
\]

is used, it will result in an undesired non-uniform distribution. To solve this problem, we must use more than one parametric equation to define a vector family.

### 4.2 Generalization of a Guide Shape

In the previous section a guide shape was given by one parametric equation. However, a guide shape does not have to be represented by only one parametric equation. In fact, most of the shapes in computer graphics are represented by more than one parametric equation. For instance, a shape that consists of polygons can be considered a shape represented by a set of linear parametric equations, or a shape represented by patches can be considered a shape defined by a set of bilinear parametric equations. Let a shape be given by a set of parametric equations,

\[
\{f_1(s), f_2(s), \ldots, f_n(s)\}.
\]

This shape can be used as a guide shape if it has a star shape with center point, \( z_r \). If so, the following parametric equations provide a desired vector family:

\[
v = z - z_r = \{f_1(s) - z_r, f_2(s) - z_r, \ldots, f_n(s) - z_r\}.
\]

then:

\[
z = \{I_1(s), I_2(s), \ldots, I_n(s)\},
\]

where

\[
I_i(s) = \frac{C f_i(s)}{F(f_i(s))} (f_i(s) - z_r) + z_r \text{ for all } i = 1, 2, \ldots, n.
\]

Let us examine how a polyhedron can be given by a set of parametric functions. For instance, a cube consists of six polygons (squares). Each one of these polygons can be defined by a parametric function. Let us assume that the vertices of one polygon (say side i) are \( z_{i,0,0}, z_{i,0,1}, z_{i,1,1}, z_{i,1,0} \). Then the parametric equation for this polygon i is

\[
f_i(u, v) = z_{i,0,0}(1 - u - v) + z_{i,0,1}v + z_{i,1,0}u.
\]

(Note that \( f_i(1, 1) = z_{i,1,1} \), since, in this case, the fourth vertex is linearly dependent to the other three vertices.)

To describe the surface of a sphere there will be six of these parametric equations. In another words, a cube can be easily used as a guide shape. Not only cubes but also all the other platonic polyhedra [Wil79] provide simple guide shapes. We have experimented all of these except the dodecahedron for the ray-affine equation of a sphere. The octahedron, cube and icosahedron give satisfactory tessellations for the sphere whereas the tetrahedron does not. In Figure 10, we show the effect of the guide shape on rendering with a tetrahedron, octahedron, cube and icosahedron. Intuitively speaking, if the shape described by the ray-affine equation is not known, the best bet is to use an octahedron or a cube. If an approximation of the desired shape is known, using the approximate shape as a guide shape is the best solution.

**Figure 10:** Tessellations of sphere by using different guide shapes.

### 5 Examples of RAFFs

The shapes generated by implicit equations are just a solution to a given equation. The same shape can be generated by many different functions. For instance,

\[
\sqrt{x^2 + y^2} - 1 = 0, \quad x^2 + y^2 - 1 = 0
\]

and

\[
x^2 + y^2 - 2\sqrt{x^2 + y^2} + 1 = 0
\]

generate the same shape. However, one of these functions is ray-affine while the others are not. To show that a shape can be generated by a ray-affine function, the functions that generate the shape must be in a ray-affine form.

Affine functions, distance functions and the super-quadrics in the form of

\[
p^2 \sqrt{(r\sqrt{zp^2 + yq^2})^p + zp^2 - 1}
\]
are already in ray-affine function form. Monoids are given in a non-ray-affine form
\[ H_1^{<m>}(z) + H_2^{<m-1>}(z), \]
where both functions are polynomials. The related ray-affine form,
\[ \frac{H_1^{<m>}(z)}{H_2^{<m-1>}(z)} + 1 \]
generates the same shapes as monoids. All shapes generated by quadratic functions can be expressed by a ray-affine form [Aki92]. Moreover, non-monomial polynomials in the form of
\[ H_1^{<m>}(z) + H_2^{<k>}(z), \]
where both function are polynomials, can be transformed into ray-affine form as
\[ \sqrt{\frac{H_1^{<m>}(z)}{H_2^{<k>}(z)}} + 1. \]
For instance, \( y^3 + x = 0 \) is not monoid and its related ray-affine form is
\[ \sqrt{x^3/y} + 1. \]

6 Construction of RAFs

Comparison with known functions gives a perspective about the power of the ray-affine form. However, to demonstrate the real power of the ray-affine form, we must provide operations to generate ray-affine functions. These operations include union and intersection and morphing operations over functions.

The following theorem provides operations to generate \( m^{th} \) degree ray-polynomials. Using the operations given by the theorem, ray-linear and ray-constants can be generated to construct ray-affine functions.

Theorem 1 The following operations give an \( m^{th} \) degree ray-polynomial with a center point \( z_r \), if all functions used have the same center point \( z_r \).

1) \( H_1^{<m>}(z) + H_2^{<m>}(z) \),
2) \( H_1^{<m-k>}(z)H_2^{<k>}(z) \),
3) \( \frac{H_1^{<m+k>}(z)}{H_2^{<k>}(z)} \),
4) \( |H_1^{<m>}(z)| \),
5) \( H_1^{<m>}(z)C(Z)H_2^{<m>}(z)^{1-C(Z)} \),
6) \( C(Z)\sqrt{H_1^{<m>}(z)C(Z) + H_2^{<m>}(z)C(Z)} \).

Proof We will just give a proof for (6). All others are relatively simple to prove and left to the reader.

Since all the functions are \( m^{th} \) degree ray-polynomial with the same center point \( z_r \), all of them will be simplified into polynomials when projected onto a ray originating from the center point.

\[ C(vt + z_r) = C(v + z_r) = A \]
\[ H_1^{<m>}(vt + z_r) = H_1^{<m>}(v + z_r)t^m = B_1t^m \]
\[ H_2^{<m>}(vt + z_r) = H_2^{<m>}(v + z_r)t^m = B_2t^m \]
where \( A, B_1, B_2 \) are real numbers. Then:
\[ \sqrt{H_1^{<m>}(z)}C(Z) + H_2^{<m>}(z)c(Z) \Rightarrow \]
\[ \sqrt{(B_1t^m)^A + (B_2t^m)^A} = \sqrt{B_1^t + B_2^t}t^m \]
The function is \( m^{th} \) degree ray-polynomial with center point \( z_r \). This finishes the proof. 

By choosing ray-linear as one of \( x, y \) or \( z \) and choosing ray-constants as any real number, the first two operations can be used to construct all polynomials. The third operation provides rational polynomials. The last three operations not only generate non-polynomials but also provide operations that implement morphing, union and intersection over the shapes described by ray-affine functions.

7 Operations over Functions

7.1 Introduction

A ray-linear function which which is always positive, \( F^+(z) \geq 0 \forall z \in D^m \) will be called a yontsal function if its center point is the origin [AH93]. In this section, shapes are represented by a ray-affine inequality in the following form:
\[ S = \{ z | F^+(z) = 1 \leq 0 \} \]
Inequalities instead of implicit equations are needed to describe set operations. The shape given by implicit equation \( F^+(z) = 1 = 0 \) is the border of \( S \).

Constraining ray-linear to yontsal functions (positive real ray-linears) does not limit the shapes. Let a shape be given by the following ray-affine inequality:
\[ F(z) - 1 \leq 0. \]

We observe that it is possible to change this inequality into a new one which gives the same shape:
\[ F^+(z) - 1 = 0.5 F(z) + 0.5|F(z)| - 1 \leq 0 \]
Note that \( F^+ \) is still a ray-linear. An approximate version of this positive real ray-linear function is given by the following equation:
\[ F^+(z) - 1 = |(0.5 + \epsilon)F(z) + (0.5 - \epsilon)|F(z)| - 1 \leq 0 \]
If \( \epsilon \) is small enough, this inequality also gives the same shape. Note that here \( F^+(z) \) is also ray-linear.

5 Yontsal is a word created by suffixation the word Yont, with the suffix sal. In Turkish, Yont means sculpt and sal which has same meaning of suffix al.
Let $S_i$'s be given by an implicit inequality as
$$S_i = \{ z | F_i^+(z) - 1 \leq 0 \}$$
where $i = 1, 2, \ldots, N$. (Note that if $F_i(z)$ is linear, $S_i$ is a half-space.) In the following subsection, we show how to generate smooth approximations of unions and intersections of $S_i$'s and morphing using the operations given in theorem 1.

### 7.2 Intersection Operation

For positive real $p \geq 2$, the following ray-affine inequality provides a smooth approximation of $S = S_1 \cap S_2 \ldots \cap S_N$ [Ric73].
$$S^p = \{ z | \sqrt[p]{F_1^+(z)^p + \ldots + F_N^+(z)^p} - 1 \leq 0 \}$$
When $p$ goes to infinity, the inequality goes to
$$\lim_{p \to \infty} S^p = \{ z | \max(F_1^+(z), \ldots, F_N^+(z)) - 1 \leq 0 \} = \bigcap_{i=1}^{N} S_i$$
If all of the $S_i$'s are half-spaces, the intersection operation provides convex polygons and polyhedra and their smooth approximations. Several examples of smooth approximation of convex polyhedra are illustrated in Figure 11. If the $S_i$'s are not half-spaces, the intersection operation provides smooth approximations of general convex shapes. Several examples of smooth approximations of general convex shapes are shown in Figure 12.

### 7.3 Union Operation

For positive real $p$'s, the following ray-affine inequality provides a smooth approximation of $S = S_1 \cup S_2 \ldots \cup S_N$ [Ric73].
$$S^{-p} = \{ z | -p\sqrt[p]{F_1^+(z)^{-p} + \ldots + F_N^+(z)^{-p}} - 1 \leq 0 \}$$
When $p$ goes to infinity, the inequality goes to
$$\lim_{p \to -\infty} S^{-p} = \{ z | \min(F_1^+(z), \ldots, F_N^+(z)) - 1 \leq 0 \} = \bigcup_{i=1}^{N} S_i$$
Several examples of this operation, which gives star shapes, are illustrated in Figure 13.

### 7.4 Morphing Operations

For $0 \leq m \leq 1$, the following ray-affine inequality provides a morphing operation over the shapes $S_1$ and $S_2$.
$$S^m = \{ z | F_1^+(z)^m F_2^+(z)^{1-m} - 1 \leq 0 \}$$
When $m = 1$, the inequality gives $F_1(z) - 1 \leq 0$, when $m = 0$, it gives $F_2(z) - 1 \leq 0$. Changing value of $m$ from one to zero, a transformation form one shape to the other is made. This operation resembles the geometric mean. Several examples of morphing operations are shown in Figure 14.

### 8 An Important Property of Ray-Constants

The next theorem allows any function to be used in ray-affine form.

**Theorem 2** [Akl92]

Let $C_1(z) = 0, C_2(z) = 0, \ldots, C_n(z) = 0$ be ray-constant functions with the same center point $z_+$, and let $I$ be any function from $\Re^n$ to $\Re$. Then
$$I(C_1(z), C_2(z), \ldots, C_n(z))$$

is a ray-constant function with the same center point.

The simplest ray-constant function is the angle between two vectors. For instance:
$$\frac{x}{\sqrt{x^2 + y^2 + z^2}}$$
gives the angle between the vectors $[1, 0, 0]$ and $[x, y, z]$. It is possible to attach a unique value to every ray with ray-constants. In other words, each ray-constant function can be considered as a parameter. Using more than one ray-constant function it is possible to give a distinct set of parameters to each ray. Then using a parametric function over this set of ray-constant functions another ray-constant function will be obtained.

Examples of ray-constant functions are shown in Figure 15.

### 9 Conclusions

We have introduced a new dual-form, ray affine functions, for representing ray-affine shapes which includes convex and star shapes. Ray-affine functions can be constructed by using union and intersection operations of constructive solid geometry. They also provide morphing operations. Yontsal functions can be used in constructive solid geometry, in Blian exponential functions [Blü82] in the form
$$\sum_{i=0}^{n} e^{-F_i^+(z)}$$
or in soft object equations [Wyn90] They can also be globally deformed as superquadrics. (Global deformed ray-affine shapes can be represented by ray-affine functions on deformed rays.) Although, ray-affines can only provide ray-affine shapes, it is always possible to use higher degree ray-polynomials for more complicated shapes.

### 10 Acknowledments

We would like to thank Dr. W. T. Rhodes for his suggestion of the name, ray-affine, instead of the original name, pseudo-affine. We also thank Argun Kocaoglu for his creation of the word, yontsal.
References


Figure 11: Smooth approximations of the intersections of the half-spaces provided by ray-affines for $p$ values 2, 4, 8, 16 from top to bottom.

Figure 12: Smooth approximations of general convex shapes.

Figure 13: Smooth approximations of unions of the shapes provided by ray-affines for $p$ values 2, 4, 8, 16 from top to bottom.

Figure 14: Examples of morphing operation.
Figure 15: Examples of shape generation by using ray-constant functions.