Raytracing Quadrics

Miles Macklin

February 6, 2013

1 Introduction

This short article shows how to define and manipulate quadric surface equations in matrix form and intersect rays against them for rendering. The treatment here is inspired by [SWBG06] and work by Simon Green at NVIDIA.

2 Quadrics

The quadratic equation in 3 variables can be written as:

$$f(x,y,z) = Ax^2 + 2Bxy + 2Cxz + 2Dx + Ey^2 + 2Fyz + Gy + Hz^2 + 2Iz + J = 0 \tag{1}$$

It can also be written in matrix form using homogenous coordinates:

$$\mathbf{x}^T \mathbf{Q} \mathbf{x} = 0 \tag{2}$$

where $\mathbf{Q}$ is the matrix of coefficients:

$$\mathbf{Q} = \begin{bmatrix} A & B & C & D \\ B & E & F & G \\ C & F & H & I \\ D & G & I & J \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3}$$

Because this matrix is symmetric it can be diagonalised by a matrix $\mathbf{T}$. With appropriate scaling, our matrix $\mathbf{Q}$ can be expressed as a diagonal matrix $\mathbf{D}$ with entries ± 1 or 0:

$$\mathbf{Q} = \mathbf{T}^{-T} \mathbf{D} \mathbf{T}^{-1} \tag{4}$$

We refer to the basis that diagonalises $\mathbf{Q}$ as the parameter space.

$\mathbf{Q}$ can may be transformed to a different basis by any affine transformation $\mathbf{M}$ by multiplying by $\mathbf{M}^{-1}$ on both sides. This is equivalent to moving a point $\mathbf{x}'$ back to the basis for $\mathbf{Q}$,

$$\mathbf{Q}' = \mathbf{M}^{-T}\mathbf{Q}\mathbf{M}^{-1}. \tag{5}$$

$$\mathbf{x}' = \mathbf{M}\mathbf{x} \tag{6}$$

$$\mathbf{x}'^T\mathbf{Q}'\mathbf{x}' = \mathbf{x}'^T\mathbf{M}^{-T}\mathbf{Q}\mathbf{M}^{-1}\mathbf{x}' = \mathbf{x}^T\mathbf{Q}\mathbf{x} \tag{7}$$

As we will soon see, view (or eye) space is a convenient space for ray-tracing. To transform our quadric equation to view space we multiply by the inverse model view matrix $\mathbf{MV}$:

$$\mathbf{Q}' = \mathbf{MV}^{-T}\mathbf{Q}\mathbf{MV}^{-1} = \mathbf{MV}^{-T}\mathbf{T}^{-T}\mathbf{D}\mathbf{T}^{-1}\mathbf{MV}^{-1} = (\mathbf{MV}\cdot\mathbf{T})^{-T}\mathbf{D}(\mathbf{MV}\cdot\mathbf{T})^{-1} \tag{8}$$

To ray trace the quadric we parameterise the view ray in the usual way,

$$\mathbf{x}_v = \mathbf{o} + t\mathbf{d}, \tag{9}$$

and transform this back to parameter space:

$$\mathbf{x}_p = (\mathbf{MV}\cdot\mathbf{T})^{-1}\mathbf{o} + t(\mathbf{MV}\cdot\mathbf{T})^{-1}\mathbf{d} = \mathbf{o}_p + t\mathbf{d}_p \tag{10}$$

Inserting this into our quadratic equation,

$$\mathbf{x}_p^T\mathbf{Q}\mathbf{x}_p = (\mathbf{o}_p + t\mathbf{d}_p)^T\mathbf{D}(\mathbf{o}_p + t\mathbf{d}_p) = 0, \tag{11}$$

and expanding and gathering for t, we have:

$$t^2(\mathbf{d}_p^T\mathbf{D}\mathbf{d}_p) + 2t(\mathbf{o}_p^T\mathbf{D}\mathbf{d}_p) + \mathbf{o}_p^T\mathbf{D}\mathbf{o}_p = 0 \tag{12}$$

which we can solve with the quadratic formula.

The advantage of defining our rays in view space is that the ray origin, or eye position $\mathbf{o}$ is simply the homogenous zero vector:

$$\mathbf{o} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{13}$$

Which simply extracts the last column of $(\mathbf{MV} \cdot \mathbf{T})^{-1}$ when we form $\mathbf{o}_p$:

$$\mathbf{o}_p = \mathbf{c}_4 \tag{14}$$

References

[SWBG06] Christian Sigg, Tim Weyrich, Mario Botsch, and Markus Gross. Gpu-based ray-casting of quadratic surfaces. In *Proceedings of the 3rd Eurographics / IEEE VGTC conference on Point-Based Graphics*, SPBG'06, pages 59–65, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.