X.8 TRANSFORMING AXIS-ALIGNED BOUNDING BOXES

James Arvo
Apollo Systems Division
of Hewlett-Packard
Chelmsford, Massachusetts

A very common type of three-dimensional bounding volume is the axis-aligned box, a parallelepiped with each face perpendicular to one coordinate axis. The appeal of this shape is its simplicity. It is ubiquitous in ray tracing because it is among the simplest objects to test for ray intersection. It is also widely used to accelerate the rendering of display lists by facilitating quick visibility tests for collections of drawing primitives.

In both contexts it is frequently necessary to construct a bounding box of an object to which an affine transformation has been applied, typically by means of a 3×3 modeling matrix, M, followed by a translation, T. A simple and frequently acceptable means of constructing such a box is to transform the bounding box of the original object and enclose the resulting arbitrary parallelepiped by an axis-aligned box. This is equivalent to transforming the eight vertices of the original box and finding the extrema of the resulting coordinates. Since each point transformation requires nine multiplies and nine adds, this would entail 144 arithmetic operations and a minimum of 21 compares. This naive approach is wasteful because it ignores the information embodied in the cube's symmetry. We will show how to take advantage of this information.

We address two common methods of encoding a bounding box, B. The first is the use of three intervals, $[B_x^{min},B_x^{max}]$, $[B_y^{min},B_y^{max}]$ $[B_z^{min}, B_z^{max}]$. Aside from ordering, this is equivalent to storing two opposing vertices. The second method is to store the box center, $(B_x^{cent}, B_y^{cent}, B_z^{cent})$, and the box *half-diagonal*, $(B_x^{diag},B_y^{diag},B_z^{diag})$, which is the positive vector from the center of the box to the vertex with the three largest components. Both of these representations are amenable to very efficient transformation.

The algorithm shown in Fig. 1 transforms box A, encoded as intervals, into another axis-aligned box, B, of the same form. The algorithm is based on the following observation. To compute a component of the transformed box, say, the maximum along the ith axis, we need only consider which of the eight vertices produces the maximal product with the ith row of the matrix. There are two possibilities for each component of the potential vertex: the minimum or the maximum of the interval for that axis. By forming both products for each component and summing the largest terms, we arrive at the maximal value. The minimal value is found by summing the smaller terms. The translation component of the matrix does not influence these choices and is simply added in.

The algorithm shown in Fig. 2 transforms box A, now encoded as a center and half-diagonal vector, into another axis-aligned box, B, of the same form. In this form the new center, B^{cent}, is obtained by simply applying the affine transformation to A^{cent}. The ith component of the new half-diagonal, B_i^{diag}, is obtained by selecting the signed half-

```
procedure Transform_Interval_Box(M,T,A,B)
begin
    for i = 1...3 do

        Start with a degenerate interval at T_i to account for translation.
        B_i^min ← T_i;
        B_i^max ← T_i;

        Add in extreme values obtained by computing the products of the
        mins and maxes with the elements of the i'th row of M

        for j = 1. . .3 do
            a ← M_i,j * A_j^min;
            b ← M_i,j * A_j^max;
            B_i^min ← B_i^min + min(a, b);
            B_i^max ← B_i^max + max(a, b);
            endloop;
        endloop;
    end;
```

Figure 1. An algorithm for transforming an axis-aligned bounding box, A, stored as three intervals into another box, B, of the same form.

```
procedure Transform_CenterDiag_Box(M,T,A,B)
begin
    for i = 1. . .3 do

        Initialize the output variables by zeroing the new half-diagonal
        and setting the new center equal to the translation T.

        B_i^cent ← T_i;
        B_i^diag ← 0;

        Compute the i'th coordinate of the center by adding M_i,* · A^cent,
        and the i'th coordinate of the half-diagonal by adding |M_i,*| · A^diag.

        for j = 1. . .3 do
            B_i^cent ← B_i^cent + M_i,j * A_j^cent;
            B_i^diag ← B_i^diag + |M_i,j| * A_j^diag;
            endloop;
        endloop;
    end;
```

Figure 2. An algorithm for transforming an axis-aligned bounding box, *A*, stored as a center and a half-diagonal into another box, *B*, of the same form.

diagonal of *A*, which results in the maximal product with the *i*th row of *M*. Here "signed" means allowing each component to be either positive or negative independently. This generates all eight half-diagonals of box *A*, pointing from A^{cent} to each vertex. We achieve the maximum product with the row of *M* by making each of its three terms positive, negating the negative elements of *M*. Because A^{cent} is a positive vector, this is equivalent to taking the absolute value of each element of *M*, as shown in Fig. 2.

The cost of both of these algorithms is only 36 arithmetic operations and 9 compares. Note that in the first algorithm both $min(a,b)$ and $max(a,b)$ can be computed with one compare, and in the second algorithm each absolute value is counted as one compare.

See Appendix 2 for C Implementation (785)