Cyrus–Beck algorithm

From Wikipedia, the free encyclopedia
(Redirected from Cyrus–Beck)

The **Cyrus–Beck algorithm** is a generalized line clipping algorithm. It was designed to be more efficient than the Sutherland–Cohen algorithm which uses repetitive clipping.[1] Cyrus–Beck is a general algorithm and can be used with a convex polygon clipping window unlike Sutherland-Cohen that can be used only on a rectangular clipping area.

Here the parametric equation of a line in the view plane is:

$$\begin{aligned} p(t) &= tp_1 + (1-t)p_0 \\ &= p_0 + t(p_1 - p_0) \end{aligned}$$

where $0 \leq t \leq 1$.

Now to find intersection point with the clipping window we calculate value of dot product. Let p_E be a point on the clipping plane *E*.

Calculate $n \cdot (p(t) - p_E)$.

if > 0 vector pointed towards interior
if = 0 vector pointed parallel to plane containing *p*
if < 0 vector pointed away from interior

Here *n* stands for normal of the current clipping plane (pointed away from interior).

By this we select the point of intersection of line and clipping window where (dot product = 0) and hence clip the line.

Contents [hide]

Notes [edit]

1. ^ "Clipping" (presentation)

1] Sutherland-Cohen can be used only on a rectangular clipping area.

2] Cyrus–Beck is a general algorithm and can be used with a convex polygon clipping window.

```
p(t) = p0 + t(p1-p0)        /* it's parametric function */
```

3] if > 0 ; vector says p(t) is OUTSIDE && A < 90 degree.

```
if < 0 ; vector says p(t) is INSIDE && a > 90 degree.
```

```
 if = 0 ; vector says p(t) is on edge E .. here outer normal edge is perpendicular
to the E and p(t)-B
```

```
.. we will writing here a function code for it as given below :
```

```
/*

if( DtProd (N,P(t)-B) > 0)
       {
```

```
             p(t) OUTER & A < 90 degree ;    /* P(t) is OUTSIDE ..

          }
    else if( DtProd (N,P(t)-B) < 0)
          {
             p(t) INNER & A > 90 degree ;    /* P(t) is INSIDE ..

          }
   else( DtProd (N,P(t)-B) = 0)
          {
             p(t) lies on to the edge E ;    /* where outer normal edge N would be
perpendicular to both E and p(t)-B..

          }
```

```
 */
```

See also [edit]

Algorithms used for the same purpose:

- Cohen-Sutherland
- Liang-Barsky
- Nicholl–Lee–Nicholl
- Fast-clipping

References in other media:

- Tron: Uprising

References [edit]

- Mike Cyrus, Jay Beck. "Generalized two- and three-dimensional clipping". Computers & Graphics, 1978: 23-28.
- James D. Foley. *Computer graphics: principles and practice*. Addison-Wesley Professional, 1996. p. 117.

External links [edit]

- http://cs1.bradley.edu/public/jcm/cs535CyrusBeck.html
- http://softsurfer.com/Archive/algorithm_0111/algorithm_0111.htm[*dead link*]

This computer graphics–related article is a stub. You can help Wikipedia by expanding it.

Categories: Clipping (computer graphics) | Computer graphics stubs