1. Introduction

A clipping operation is a fundamental operation within all computer graphics systems and it has been well explored as many algorithms have been developed. However there are still other ways how a graphical primitive can be clipped against a window. The type of the window has a strong influence on the final algorithm complexity. The window can be orthogonal and axis aligned, 4-sided orthogonal non-axis aligned convex or non-convex. Clipped primitives are mostly lines, line segments and polygons of the above mentioned types. It leads to algorithms with $O(N)$ [Skala 1994] up to $O(N^2)$ complexities in general. In some cases, when a clipping window is constant and many primitives are to be clipped, a clipping algorithm with $O(1)$ complexity [Skala 1996] can be used. Algorithms are mostly based on Cohen-Sutherland, Cyrus-Beck, Lindas and Sutherland-Hodgman algorithms [Theoharis, et al. 2008]. Some of them are intended for hardware implementation but all operate in the Euclidean space.

Points in computer graphics are represented in homogeneous coordinates as $x'y'zw$, i.e. actually in the projective extension of the Euclidean space. A line in the implicit form is represented by a vector $a,b,c$, i.e. $ax+by+cz=0$. A point $A$ is inside the clipping window or is outside it, depending on the sign of the value $ax+by+cz$. It can be seen that this algorithm is quite general as it does not need the clipping window edges aligned with the axes and it is applicable for clipping by a convex window as well. In the case of an orthogonal axis aligned window and with the corners given in the Euclidean space, i.e. $w_1=1$, the code can be even more optimized. If clipping is a part of the graphical pipeline, then clipping itself is made in the normalized space, i.e. in the interval $\langle -1,1 \rangle \times \langle -1,1 \rangle$.

It means that the transformation pipeline $R(\text{clip})S$ is actually modified to the “normalized” clip given by the transformation:

$$R \text{ (normalized clip)} \quad N = R' \text{ (normalized clip)} \quad S'$$

where: $R'$ and $S'$ are cumulative transformation matrices. Now, the vectors of edges are $\langle -1,0,0 \rangle$ or $\langle 0,1,0 \rangle$. It means that multiplications in the if $p_jk \geq 0$ statement are not needed. The cost of multiplication is actually hidden to the cumulative matrices, but it is made only once independently from a number of primitives processed.

Analyzing the algorithm, it can be seen that the proposed concept and algorithm above can be used for any 4-sided clipping window, not necessarily convex, for axis aligned windows and for N-sided convex polygon clipping as well. The tab.1 can be easily generated for a convex polygon by an algorithm as follows:

2. Proposed Smart Clip Algorithm

The approach presented is based on a basic principle – test first and then compute. Unlike Cohen-Sutherland’s algorithm it evaluates a position of the given line with respect to the corners of the clipping window.

2.1. Line clipping

Let us consider a line $p$ given by two points $x_i$ and $p_2$ as $p_1$, $p_2$ clipped by an orthogonal axis aligned window. The clipping window is given by corners $x_0$, $x_1$, $x_2$, $x_3$. The position of the line relative to the corners of the clipping window is determined by a code vector $c$. The binary code of the window vertices for the line $p$ can be determined as

$$c = \min\{i|p_jk \geq 0\}$$

with respect to the corners of the clipping window.

Algorithm 1

It should be noted, that the vector $c$ is constructed differently from the Cohen-Sutherland’s algorithm as the window corner’s position against the parameter $p_j$ is now coded.

This is actually a classification of corners by a half space given by a line.

where: $\langle x, y, w \rangle$ and $\langle a, b, c \rangle$ are homogeneous coordinates, but it is made only once independently from a number of primitives processed. It can be seen that this algorithm is quite general as it does not need the clipping window edges aligned with the axes and it is applicable for clipping by a convex window as well. In the case of an orthogonal axis aligned window and with the corners given in the Euclidean space, i.e. $w_1=1$, the code can be even more optimized. If clipping is a part of the graphical pipeline, then clipping itself is made in the normalized space, i.e. in the interval $\langle -1,1 \rangle \times \langle -1,1 \rangle$.

Also the intersection computation of $(\text{line} \cap \text{window})$ is to be done for all $E^2$ resulting segments. Therefore, the algorithm is of $O(N)\cdot O(K)$ complexity, where $K$ is a number of segments.

2.3. Non-convex Window

The presented concept and algorithm are also valid for non-convex line and line segment clipping with some modifications as the values $r_j$ have to be ordered as there might be more than one segment, i.e. several intervals $<t_{\text{left}}, t_{\text{right}}>\cap \langle 0,1 \rangle$ is to be done for all $E^2$ resulting segments. Therefore, the algorithm is of $O(N)\cdot O(K)\cdot O(K)$ complexity, where $K$ is a number of segments.

3. Results

The presented concept and algorithms were experimentally verified. The proposed algorithm is slightly faster than the Cohen-Sutherland and about 2 times faster than Cyrus-Beck algorithms if all points are given in homogeneous coordinates or the window is in a general position the proposed algorithm is significantly faster.

4. Conclusions

A new approach is presented to line and line segment clipping by a convex polygon in $E^2$. The algorithm is simple, easy to implement and convenient for GPU application, too. The presented concept is more general, offers higher robustness and reduces division operations. It is comparable and competitive, at least with other relevant algorithms.

A significant advantage of the proposed concept is the use of a separation function, which is not only more robust, but enables avoidance of complicated cases difficult to resolve [Skala 1989].

Acknowledgement

The author would like to thank to colleagues at the University of West Bohemia, Pilsen, VSB-Technical University, Ostrava in Czech Republic, Shandong University and Zhejiang University in China for their comments and suggestions. Research was supported by MSMT CR projects ME10060, LH12181.

References


