Clipping in Raster World

It is a procedure that identifies those operations of picture that are either inside or outside is called clipping.

The region against which an object is to be clipped is called a clip window.

Depending on application it can be polygons or even curve surfaces.

Applications

i. Extracting parts of defined scene for viewing

ii. Identifying visible surfaces in three dimension views

iii. Drawing, painting operations that allow parts of a picture to be selected for copying, moving, erasing or duplicating etc.

Displaying only those parts of picture that are within window area, discard everything outside window.

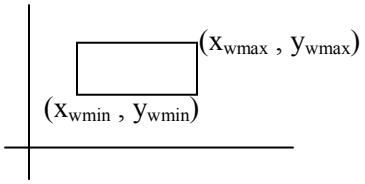
World coordinate clipping removes those primitives outside window from further consideration thus eliminating process necessary to transfer those primitives to device space.

View port clipping requires transformation to device coordinate be performed for all objects including those outside window area.

Point Clipping

Assuming the clip window is a rectangle, the lower left corner of the window is define by $x = x_{wmin}$ and $y = y_{wmin}$ and the upper left corner of the window is define by $x = x_{wmax}$ and $y = y_{wmax}$ a point p=(x,y) if visible for display if the following inequalities are satisfied:

$$x_{wmin} \leq x \leq x_{wmax}$$
$$y_{wmin} \leq y \leq y_{wmax}$$



Applied to scenes involving explosions or sea foam that are modeled with points distributed in some region of the scene.

Cohen Shutherland Line Clipping Algorithm

It is based on a coding scheme

Makes clever use of bit operations to perform this test efficiently

For each end point , a 4 bit binary code is used

The lower order (bit 1) is set to 1 if the end point is at the left side of the window otherwise set to 0

Bit 2, is set to 1 if the end point is at the right side of the window otherwise set to 0

Bit 3, is set to 1 if the end point is at the bottom of the window otherwise set to 0

Bit 4, is set to 1 if the end point is above the window otherwise set to 0

By numbering bit positions in region code as 1 – 4 from right to left coordinate regions can be co-related with bit positions as

Bit 4	Bit 3	Bit 2	Bit 1
x	x	x	x
up	bottom	right	left

Value of 1 in any bit position indicates that the point is in that relative position otherwise bit position is set to 0.

if point is within clipping rectangle, region code is 0000.

if point is below and to the left of rectangle then region code is 0101.

The region codes are shown

1001	**1000**	**1010**
0001	**0000**	**0010**
0101	**0100**	**0110**

Algorithm:

Step 1: Establish the region codes for all line end points:

Bit 1 is set to 1 if $x < x_{wmin}$ otherwise set it to 0 Bit 3 is set to 1 if $y < y_{wmin}$ otherwise set it to 0

Bit 2 is set to 1 if $x > x_{wmax}$ otherwise set it to 0 Bit 4 is set to 1 if $y > y_{wmax}$ otherwise set it to 0

Step 2: Determine which lines are completely inside the window and which are completely outside , using the following tests:

a. If both end points of the line have region codes 0000 the line is completely inside the window
b. If the logical AND operation of the region codes of the two end points is NOT 0000 then the line is completely outside (same bit position of the two end points have 1)

Step 3: if both the tests in step 2 fail then the line is not completely inside nor outside . so we need to find out the intersection with the boundaries of the window

slope (m) = $(y_2 - y_1)/(x_2 - x_1)$

a. If bit 1 is 1 then the line interests with the left boundary and $y_i = y_1 + m * (x - x_1)$ where $x = x_{wmin}$
b. If bit 2 is 1 then the line interests with the right boundary and $y_i = y_1 + m * (x - x_1)$ where $x = x_{wmax}$
c. If bit 3 is 1 then line interests with the bottom boundary and $x_i = x_1 + (y - y_1)/m$ where $y = y_{wmin}$
d. If bit 4 is 1 then line interests with the upper boundary and $x_i = x_1 + (y - y_1)/m$ where $y = y_{wmax}$

Here, x_i and y_i are the x and y intercepts for that line

Step 4: repeat Step1 to Step3 until the line is completely accepted or rejected