Applications of Computer Graphics

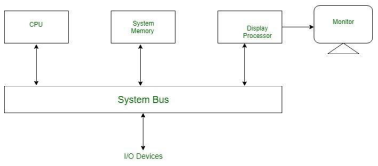
- **Computer-Aided Design -**CAD methods are now routinely used in the design of buildings, automobiles, aircraft, watercraft, spacecraft computers, textiles, and many other products. For some design applications, objects are first displayed in a wireframe outline form that shows the overall shape and internal features of objects. Wireframe displays allow designers to quickly see the effects of interactive adjustments to design shapes.
- **Presentation Graphics** - Presentation Graphics programs are used to produce illustrations for reports or to generate PowerPoint presentations. It is commonly used to summarize financial, statistical, mathematical, scientific, and economic data for research reports, managerial reports consumer information bulletins, and other types of reports.
- **Computer Art** - Computer graphics methods are widely used in both fine art and commercial art applications. Artist's paintbrush programs (Lumena), Pixel Paint and Super Paint, symbolic mathematics packages (Mathematica), CAD packages, desktop publishing software, animation packages that provide facilities for designing object shapes, and specifying object motions.
- **Entertainment** - Computer graphics methods are now commonly used in making motion pictures, music videos, and television shows. Sometimes graphics objects are displayed by themselves, and sometimes graphics objects are combined with the actors and live scenes.
- **Education and Training** - Computer-generated models of physical, financial, and economic systems are often used as educational aids. Models of physical systems, physiological systems, population trends, or equipment, such as color-coded diagrams can help trainees to understand the operation of the system.
- **Image Processing** - In computer graphics, a computer is used to create a picture. Image processing, on the other hand, applies techniques to modify or interpret existing pictures, such as photographs and TV scans. Two principal applications of image processing are improving picture quality and machine perception of visual information as used in robotics.

Random Scan Display

In **Random-Scan Display** electron beam is directed only to the areas of screen where a picture has to be drawn. It is also called **vector display**, as it draws picture one line at time. It can draw and refresh component lines of a picture in any specified sequence. Pen plotter is an example of random-scan displays. The number of lines regulates refresh rate on random-scan displays. An area of memory called **refresh display files** stores picture definition as a set of line drawing commands. The system returns back to the first-line command in the list after all the drawing commands have been processed. High-quality vector systems can handle around 100, 00 short lines at this refresh rate. Faster refreshing can burn phosphor. To avoid this every refresh cycle is delayed to prevent a refresh rate greater than 60 frames per second.

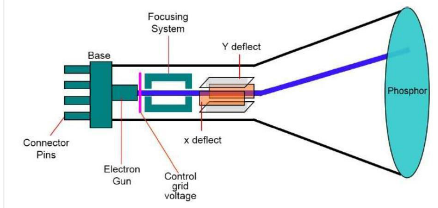
Random Scan System

Input in the form of an application program is stored in the system memory along with graphics package. Graphics package translates the graphic commands in the application program into a display file stored in system memory. This display file is then accessed by the display processor to refresh the screen. The display processor cycles through each command in the display file program. Sometimes the display processor in a random scan is referred to as *Display Processing Unit / Graphics Controller*.



Refresh Cathode Ray Tubes

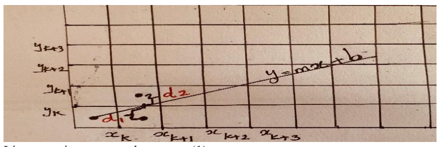
In a graphic system, the video monitor is a primary output device and the operation of most the video monitors is based on the Cathode Ray Tube. In a cathode-ray tube a **beam of electrons**, emitted by an **electron gun**, passes through **focusing and deflection systems**, that direct the beam towards a specific position on the **phosphor-coated screen.** The part of the phosphor screen emits light where the electron beam strikes it. The light emitted by the phosphor fades very rapidly, one way to keep the phosphor glowing is to redraw the picture repeatedly by quickly directing the electron beam back over the same points. This type of display is called a **refresh CRT**. Computer monitors often have a maximum **refresh rate.** This number, measured in hertz (Hz), determines how many times the screen is redrawn each second. Typical refresh rates for CRT monitors include 60, 75, and 85 Hz.



The primary components of an electron gun in a CRT are the heated metal cathode and a control grid. Heat is supplied to the cathode by directing a current through a coil of wire, called the filament, inside the cylindrical cathode structure. This causes electrons to be boiled off the hot cathode surface. In the vacuum inside the CRT envelope, the free negatively charged electrons are then accelerated towards the phosphor coating by a high positive voltage. The amount of light emitted by the phosphor coating depends on the number of electrons striking the screen.

RASTER SCAN	RANDOM SCAN
While the resolution of raster scan is lesser or lower than random scan	The resolution of random scan is higher than raster scan
While the cost of raster scan is lesser than random scan.	It is costlier than raster scan.
It stores picture definition as a set of intensity values of the pixels in the frame buffer.	It stores picture definition as a set of line commands in the memory.
While in raster scan, any alteration is not so easy.	In a random scan, any alteration is easy in comparison of a raster scan.
The refresh rate is 60 to 80 frames per second and is independent of picture complexity.	The refresh rate depends on the number of lines to be displayed i.e. 30 to 60 times per second.
Eg: TV Sets	Eg : Pen Plotter

Bresenham's Line Drawing Algorithm

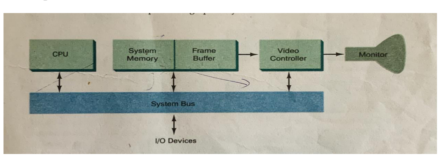


Line equation y=m.x +b ……….(1)
The y coordinate on the mathematical line at pixel column position xk +1 is calculated as
y=m(xk+1)+b ……………….(2)
then d1=y-yk
= m(xk+1)+b-yk ………………(3)
d2=(yk+1)-y
= yk+1 – m(xk+1)-b ………………(4)
The difference between d1 and d2 is
d1-d2 = m(xk+1)+b-yk - (yk+1 – m(xk+1)-b)

Raster Scan Display

Raster Scan Displays are the most common type of graphics monitor which employs CRT. It is based on television technology. In a raster scan system electron beam sweeps across the screen, from top to bottom covering one row at a time. A pattern of illuminated pattern of spots is created by turning beam intensity on and off as it moves across each row. **A memory area called refresh buffer or frame buffer stores picture definition.** This memory area holds intensity values for all screen points. Stored intensity values are restored from the frame buffer and painted on-screen taking one row at a time. Each screen point is referred to as a pixel or pel (shortened form of picture element). The **refresh rate** (or "vertical refresh rate", "vertical scan rate") is the number of times per second that a raster-based display device redraws images. On cathode ray tube (CRT) displays, higher refresh rates produce less flickering, thereby reducing eye strain. In raster scan systems refreshing is done at a rate of 60-80 frames per second. Refresh rates are also sometimes described in units of cycles per second / Hertz (Hz). At the end of each scan line, the electron beam begins to display the next scan line after returning to the left side of the screen. The return to the left of the screen after the refresh of each scan line is known as the **horizontal retrace** of the electron beam. At the end of each frame, the electron beam returns to the top left corner and begins the next frame, this movement is known as **vertical retrace.**

In addition to the central processing unit, a special purpose processor called the **video controller or video display controller** is used to control the operation of the display device . A fixed area of the system memory is reserved for the frame buffer, and the video controller is given direct access to the frame buffer memory. Frame buffer locations and corresponding screen positions are referenced in Cartesian coordinates.



= 2 m(xk+1) - 2yk +2b -1 ……………(5)
Substitute m=dy/dx in equ 5
(d1-d2)dx = 2.dy(xk+1) -2yk.dx+2b.dx -dx
(d1-d2)dx = 2.dy xk +2.dy-2yk.dx+2b.dx -dx
pk =**2.dy xk – 2dx.yk** +2.dy+2b.dx -dx
pk = **2.dy xk – 2dx.yk + C ………………..(6)** where Pk is decision parameter
if(pk <=0 ie d1-d2 <0 ,meaning actual line lays closer to yk ,then next pixel coordinate is
(xk+1,yk) .
if(pk >0 ie d1-d2 >0 ,meaning actual line lays closer to yk+1 ,then next pixel coordinate is
(xk+1,yk+1) .

Calculate successive decision parameters

At step k+1 the decision parameter is pk+1=2.dy.xk+1-2dxyk+1
+C ………….(8)
equ(8)-equ(6) pk+1- pk =2.dy.xk+1-2dxyk+1 +C - (2.dy xk – 2dx.yk + C)
=2.dy(xk+1-xk) -2.dx(yk+1-yk)
pk+1 = pk + 2.dy(xk+1-xk) -2.dx(yk+1-yk) …………..(9)
if pk <=0 then next pixel coordinate is (xk+1,yk) and **pk+1 = pk + 2.dy(xk+1-xk)**
pk+1 = pk + 2.dy ………(10)
If pk >0 , then next pixel coordinate is (xk+1,yk+1) and
pk+1 = pk + 2.dy(xk+1-xk) -2.dx(yk+1-yk)
pk+1 = pk + 2.dy -2.dx ………..(11)
The first decision parameter P0 is evaluated at starting pixel position (x0,y0) and with m
evaluated as dy/dx ,using equation (6)
pk =**2.dy xk – 2dx.yk** +2.dy+2b.dx -dx
y=mx+b , b=y-mx →b=y-(dy/dx).x substitute in above equation
p0 =**2.dy x0– 2dx.y0** +2.dy+2(y0-(dy/dx)x0).dx – dx
p0 = **2.dy x0– 2dx.y0** +2.dy+2y0.dx-2dy.x0 – dx
p0 =2.dy-dx -……………..(12)

Bresenham's Line-Drawing Algorithm for |m|<1

1. Input the two endpoints and store the left endpoint in (x0,y0).
2. Load (x0,y0) into the frame buffer; that is plot the first point.

3. Calculate the constants dx, dy, 2dy, and 2dy–2dx and get the first value for the
decision parameter as
p0=2dy−dx
4. At each Xk along the line, starting at k = 0, perform the following test –
If **pk < 0,** the next point to plot is **(xk+1,yk)** and **Pk+1 =Pk +2dy**
Otherwise, the next point to plot is **(xk+1,yk+1)** and **Pk+1=Pk+2dy-2dx**
5. Repeat step 4 dx times.

Digital Differential Analyzer (DDA) Line Drawing Algorithm

```
#include "device.h"
#define Round(a) ((int)(a+0.5))
void lineDDA(int xa, int ya, int xb, int yb)
{ int dx=xb
-xa, dy=yb-ya , steps, k;
float xIncrement , yIncrement , x=xa , y=ya;
if(abs(dx)>abs(dy))
steps=abs(dx);
else
steps=abs(dy);
xIncrement=dx/(float)steps;
yIncrement=dy/(float)steps;
setPixel(Round(x) , Round(y));
for(k=0;k<steps;k++)
{ x+=xIncrement;
y+=yIncrement;
setPixel(Round(x),Round(y));
}
}
```

The advantages of DDA Algorithm are-
• It is a simple algorithm.
• It is easy to implement.
• It avoids using the multiplication operation which is costly in terms of time complexity.

The disadvantages of the DDA Algorithm are-

• There is an extra overhead of using the round-off () function.
• Using the round-off () function increases the time complexity of the algorithm.
• Resulted lines are not smooth because of the round-off () function.
• The points generated by this algorithm are not accurate.

Bresenham's Circle Drawing Algorithm

1. Input radius r and circle center(xc,yc) , then set the coordinates for the first point on the
circumference of a circle centered on the origin as (x0,y0)=(0,r).
2. Calculate the initial decision parameter as d0=3-2r
3. At each xi , from i=0 perform the following :
If di<0 ,next point to plot along the circle centered on (0,0) is (xi+1,yi) and
di+1=di+4xi+6 .
Otherwise ,
Next point to plot is (xi+1,yi-1) and
di+1=di +4(xi-yi)+10 .
4. Determine the symmetry points in the other seven octants.
5. Move each calculated pixel position (x,y) onto the circular path centered at (xc,yc) and
plot the coordinate values: x=x+xc; y=y+yc ;
6. Repeat steps 3 through 5 until x>=y

Example
Given a circle radius r=10 , determine positions along the circle octant in the first quadrant
from x=0 to x=y. The initial value of the decision parameter is
d0=3-2x10=-17
Initial point is (x0,y0)=(0,10)

k	dk	(x_{k+1},y_{k+1})	$4x_i+6$	4(xi-yi)+10
0	-17	(1,10)	4x0+6=6	
1	-17+6=-11	(2,10)	4x1+6=10	
2	-11+10=-1	(3,10)	4x2+6=14	
3	-1+14=13	(4,9)		4(3-10)+10=-18
4	13-18=-5	(5,9)	4x4+6=22	
5	-5+22=17	(6,8)		4(5-9)+10=-6
6	117-6=11	(7,7) x=y then stop		

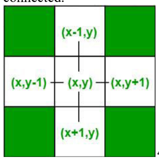
Boundary Fill Algorithm

Boundary Fill Algorithm starts at a pixel inside the polygon to be filled and paints the interior proceeding outwards towards the boundary. This algorithm works only if the color with which the region has to be filled and the color of the boundary of the region are different. If the boundary is of one single color, this approach proceeds outwards pixel by pixel until it hits the boundary of the region.

Boundary Fill Algorithm is recursive in nature. It takes an interior point(x, y), a fill color, and a boundary color as the input. The algorithm starts by checking the color of (x, y). If it's color is not equal to the fill color and the boundary color, then it is painted with the fill color and the function is called for all the neighbors of (x, y). If a point is found to be of fill color or of boundary color, the function does not call its neighbors and returns. This process continues until all points up to the boundary color for the region have been tested.

The boundary fill algorithm can be implemented by 4-connected pixels or 8-connected pixels.

4-**connected pixels** : After painting a pixel, the function is called for four neighboring points. These are the pixel positions that are right, left, above, and below the current pixel. Areas filled by this method are called 4-connected.

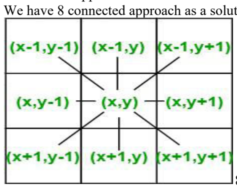


4 connected pixels

Boundary Fill Algorithm(4 connected approach)

```
void boundaryFill4(int x, int y, int fill_color,int boundary_color)
{
    if(getpixel(x, y) != boundary_color &&
       getpixel(x, y) != fill_color)
    {
        putpixel(x, y, fill_color);
        boundaryFill4(x + 1, y, fill_color, boundary_color);
        boundaryFill4(x, y + 1, fill_color, boundary_color);
        boundaryFill4(x - 1, y, fill_color, boundary_color);
        boundaryFill4(x, y - 1, fill_color, boundary_color);
    }
}
```

4 connected approach fails in such a situation.
We have 8 connected approach as a solution .



8 connected pixels

Boundary Fill Algorithm(8 connected approach)

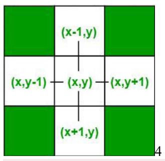
```
void boundaryFill8(int x, int y, int fill_color,int boundary_color)
{
    if(getpixel(x, y) != boundary_color &&
       getpixel(x, y) != fill_color)
    {
        putpixel(x, y, fill_color);
        boundaryFill8(x + 1, y, fill_color, boundary_color);
        boundaryFill8(x, y + 1, fill_color, boundary_color);
        boundaryFill8(x - 1, y, fill_color, boundary_color);
        boundaryFill8(x, y - 1, fill_color, boundary_color);
        boundaryFill8(x - 1, y - 1, fill_color, boundary_color);
        boundaryFill8(x - 1, y + 1, fill_color, boundary_color);
        boundaryFill8(x + 1, y - 1, fill_color, boundary_color);
        boundaryFill8(x + 1, y + 1, fill_color, boundary_color);
    }
}
```

Flood Fill Algorithm

In this method, a point or seed which is inside the region is selected. This point is called a seed point. Then four connected approaches or eight connected approaches is used to fill with a specified color .When boundary is of many colors and interior is to be filled with one color we use this algorithm.

In fill algorithm, we start from a specified interior point (x, y) and reassign all pixel values are currently set to a given interior color with the desired color. Using either a 4-connected or 8-connected approach, we then step through pixel positions until all interior points have been repainted.

<u>**Disadvantages -**</u>Very slow algorithm , May be fail for large polygons , Initial pixel required more knowledge about surrounding pixels.



4 connected pixels

```
Procedure floodfill (x, y,fill_ color, old_color: integer)
   If (getpixel (x, y)=old_color)
   {
   setpixel (x, y, fill_color);
   fill (x+1, y, fill_color, old_color);
    fill (x-1, y, fill_color, old_color);
   fill (x, y+1, fill_color, old_color);
   fill (x, y-1, fill_color, old_color);
    }
}
```

In Flood fill, all the connected pixels of a selected color get replaced by fill color.

In Boundary fill, the program stops when a given color boundary is found.

Disadvantages of Boundary-Fill over Flood-Fill:

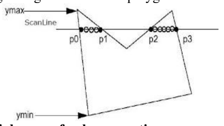
In boundary-fill algorithms, each pixel must be compared against both the new colour and the boundary colour. In flood-fill algorithms, each pixel need only be compared against the new colour. Therefore flood-fill algorithms are slightly faster.

Advantages of Boundary-Fill over Flood-Fill:

All pixels in the region must be made the same colour when the region is being created.

Scanline Polygon filling Algorithm

Scanline filling is basically the filling up of polygons using horizontal lines or scanlines.This algorithm works by intersecting scanlines with polygon edges and fills the polygon between pairs of intersections.



Special cases of polygon vertices:

If both lines intersecting at the vertex are on the same side of the scanline, consider it as two points.If lines intersecting at the vertex are at opposite sides of the scanline, consider it as only one point.

To effectively perform a polygon fill, we can store the polygon boundary in a sorted edge table that contains all the information necessary to process the scan lines efficiently, Each entry in the table for a particular scan line contains the maximum y value for that edge, the x-intercept value (at the lower vertex) for the edge, and the inverse slope of the edge.For each scan line , edges are in sorted order from left to right.

Next process the scan lines from the bottom of the polygon to its top, and produce an active edge list for each scan line crossing the polygon boundaries.The active edge list for a scan line contains all edges crossed by that scan line, with iterative coherence calculations used to obtain the edge intersections.

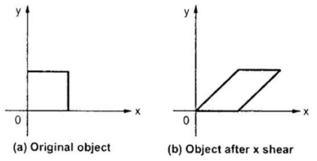
Step 1 − Find out the Ymin and Ymax from the given polygon.
Step 2 − ScanLine intersects with each edge of the polygon from Ymin to Ymax. Name each intersection point of the polygon.
Example : consider previous figure ,intersections are named as p0,p1,p2,p3
Step 3 − Sort the intersection point in the increasing order of X coordinate i.e. (p0,p1) ,(p1,p2) ,(p2,p3).
Step 4 − Fill all those pair of coordinates that are inside polygons and ignore the alternate pairs.

Shear

A transformation that slants the shape of an object is called the shear transformation. There are two shear transformations X-Shear and Y-Shear. One shifts X coordinates values and other shifts Y coordinate values. However; in both the cases only one coordinate changes its coordinates and other preserves its values. Shearing is also termed as Skewing

X-Shear

The X-Shear preserves the Y coordinate and changes are made to X coordinates, which causes the vertical lines to tilt right or left.

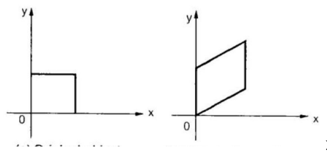


x' = x + Shx .y

y' = y

Y-Shear

The Y-Shear preserves the X coordinates and changes the Y coordinates

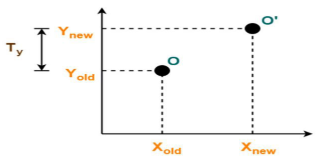


y' = y + Shy . x

x' = x

2D Transformation

Transformation means changing some graphics into something else by applying rules.We can have various types of transformations such as translation,rotation scaling , reflection, shearing, etc. When a transformation takes place on a 2D plane, it is called 2D transformation.
2D Translation is the process of moving an object from one position to another in a two-dimensional plane.



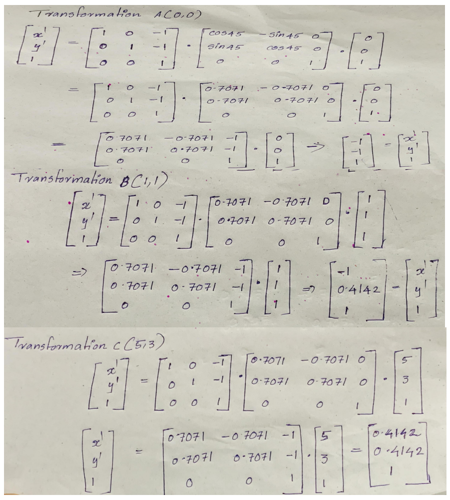
This translation is achieved by adding the translation coordinates to the old coordinates of the object as-
Xnew = Xold + Tx
Ynew = Yold + Ty
In Matrix form, the above translation equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \end{bmatrix} = \begin{bmatrix} X_{old} \\ Y_{old} \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix}$$

Translation Matrix

P'=P+T

Q) Perform 45 degree rotation of a triangle A(0,0) ,B(1,1) and C(5,3) about the origin and about the fixed point(-1,-1).



Transformation A(0,0)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos 45 & -\sin 45 & 0 \\ \sin 45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.7071 & -0.7071 & 0 \\ 0.7071 & 0.7071 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.7071 & -0.7071 & -1 \\ 0.7071 & 0.7071 & -1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

Transformation B(1,1)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.7071 & -0.7071 & 0 \\ 0.7071 & 0.7071 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} 0.7071 & -0.7071 & -1 \\ 0.7071 & 0.7071 & -1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} -1 \\ 0.4142 \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

Transformation C(5,3)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.7071 & -0.7071 & 0 \\ 0.7071 & 0.7071 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 3 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 0.7071 & -0.7071 & -1 \\ 0.7071 & 0.7071 & -1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 3 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.4142 \\ 0.4142 \\ 1 \end{bmatrix}$$

2D SCALING

In computer graphics, scaling is a process of modifying or altering the size of objects.Scaling may be used to increase or reduce the size of the object.
Scaling subjects the coordinate points of the original object to change.
Scaling factor determines whether the object size is to be increased or reduced.
If scaling factor > 1, then the object size is increased.
If scaling factor < 1, then the object size is reduced.

Specifying a value of 1 for both sx and sy leaves the size of objects unchanged .

When Sx and Sy are assigned the same value , uniform scaling is produced that maintains relative object proportions.When Sx and Sy are assigned the value ½,length and distance from origin are reduced by half.

Initial coordinates of the object O = (Xold, Yold)

Scaling factor for X-axis = Sx
Scaling factor for Y-axis = Sy
New coordinates of the object O after scaling = (Xnew, Ynew)
This scaling is achieved by using the following scaling equations-

Xnew = Xold x Sx Ynew = Yold x Sy

$$\begin{bmatrix} X_{new} \\ Y_{new} \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \end{bmatrix}$$

Scaling Matrix

Xnew = Xold x Sx

P'=S.P

Ynew = Yold x Sy

Q) Show that the composition of two successive rotations are additive R(ɵ1).R(ɵ2)=R(ɵ1+ ɵ2)

we can write rotation matrix $R(\theta_1)$ as

$$R(\theta_1) = \begin{bmatrix} \cos\theta_1 & \sin\theta_1 \\ -\sin\theta_1 & \cos\theta_1 \end{bmatrix} \text{ and } R(\theta_2) = \begin{bmatrix} \cos\theta_2 & \sin\theta_2 \\ -\sin\theta_2 & \cos\theta_2 \end{bmatrix}$$

$$R(\theta_1)\cdot R(\theta_2) = \begin{bmatrix} \cos\theta_1 & \sin\theta_1 \\ -\sin\theta_1 & \cos\theta_1 \end{bmatrix} \times \begin{bmatrix} \cos\theta_2 & \sin\theta_2 \\ -\sin\theta_2 & \cos\theta_2 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta_1\cdot\cos\theta_2 + \sin\theta_1\cdot(-\sin\theta_2) & \cos\theta_1\cdot\sin\theta_2 + \sin\theta_1\cdot\cos\theta_2 \\ -\sin\theta_1\cdot\cos\theta_2 + \cos\theta_1\cdot(-\sin\theta_2) & -\sin\theta_1\cdot\sin\theta_2 + \cos\theta_1\cdot\cos\theta_2 \end{bmatrix}$$

$$= \begin{bmatrix} \cos(\theta_1+\theta_2) & \sin(\theta_1+\theta_2) \\ -\sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) \end{bmatrix}$$

3D Rotation

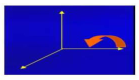
3D Rotation is a process of rotating an object with respect to an angle in a three-dimensional plane .
In 3 dimensions, there are 3 possible types of rotation-

X-axis Rotation
Y-axis Rotation
Z-axis Rotation

X-Axis Rotation

Xnew = Xold
Ynew = Yold .cosθ – Zold .sinθ
Znew = Yold .sinθ + Zold .cosθ

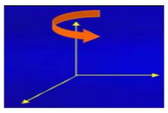
$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$



Y-Axis Rotation

Xnew = Zold . sinθ + Xold . cosθ
Ynew = Yold
Znew = Zold .cosθ – Xold . sinθ

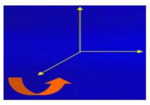
$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$



Z-Axis Rotation

Xnew = Xold .cosθ – Yold .sinθ
Ynew = Xold .sinθ + Yold .cosθ
Znew = Zold

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$



Window to viewport transformation.

Window to Viewport Transformation is the process of transforming 2D world-coordinate objects to device coordinates. Objects inside the world or clipping window are mapped to the viewport which is the area on the screen where world coordinates are mapped to be displayed.

A world-coordinate area selected for display is called a window.An area on a display device to which a window is mapped is called a viewport. The window defines what is to be viewed; The viewport defines where it is to be displayed.

World coordinate – It is the Cartesian coordinate w.r.t which we define the diagram, like Xwmin, Xwmax, Ywmin, Ywmax

Device Coordinate –It is the screen coordinate where the objects are to be displayed, like Xvmin, Xvmax, Yvmin, Yvmax

Window –It is the area on world coordinate selected for display.

ViewPort –It is the area on the device coordinate where graphics is to be displayed.

Clipping

When we have to display a large portion of the picture, then not only scaling & translation is necessary, the visible part of picture is also identified. This process is not easy. Certain parts of the image are inside, while others are partially inside. The lines or elements which are partially visible will be omitted.For deciding the visible and invisible portion, a particular process called clipping is used. Clipping determines each element into the visible and invisible portion. Visible portion is selected. An invisible portion is discarded.
The window against which object is clipped called a clip window. It can be curved or rectangle in shape.
Applications of clipping:
It will extract part we desire.
For identifying the visible and invisible area in the 3D or 2D object.
For drawing operations.
For deleting, copying, moving part of an object.
Types of Clipping: Point Clipping, Line Clipping , Area Clipping (Polygon) , Curve Clipping,Text Clipping ,Exterior Clipping

Given a homogeneous point (1, 2, 3). Apply rotation 90 degree towards X, Y and Z axis and find out the new coordinate points.

X-Axis Rotation
Xnew = Xold = 1
Ynew = Yold x cosθ – Zold x sinθ = 2 x cos90° – 3 x sin90° = 2 x 0 – 3 x 1 = -3
Znew = Yold x sinθ + Zold x cosθ = 2 x sin90° + 3 x cos90° = 2 x 1 + 3 x 0 = 2
New coordinates after rotation = (1,-3,2)
Y-Axis Rotation
Xnew = Zold x sinθ + Xold x cosθ = 3 x sin90° + 1 x cos90° = 3 x 1 + 1 x 0 = 3
Ynew = Yold = 2
Znew = Yold x cosθ – Xold x sinθ = 2 x cos90° – 1 x sin90° = 2 x 0 – 1 x 1 = -1
New coordinates after rotation = (3,2,-1)
Z-Axis Rotation
Xnew = Xold x cosθ – Yold x sinθ = 1 x cos90° – 2 x sin90° = 1 x 0 – 2 x 1 = -2
Ynew = Xold x sinθ + Yold x cosθ = 1 x sin90° + 2 x cos90° = 1 x 1 + 2 x 0 = 1
Znew = Zold = 3
New coordinates after rotation = (-2,1,3)

3D Scaling

Scaling is a process of modifying or altering the size of objects.Scaling may be used to increase or reduce the size of object.Scaling subjects the coordinate points of the original object to change.Scaling factor determines whether the object size is to be increased or reduced.
If scaling factor > 1, then the object size is increased.
If scaling factor < 1, then the object size is reduced.
Xnew = Xold x Sx
Ynew = Yold X Sy
Znew = Zold X Sz

Cohen Sutherland Line Clipping

Most popular line clipping algorithm ,speed up the processing of line segments by initial tests that reduce the number of intersections that must be calculated .
All lines come under any one of the following categories
Visible: If a line lies within the window, i.e., both endpoints of the line lie within the window. A line is visible and will be displayed as it is.
Not Visible: If a line lies outside the window, it will be invisible and rejected.
Clipping Case: If the line is neither visible case nor invisible case. It is considered to be the clipped case. Reject the portion of line that are outside the clipping window and accept portion that are inside the clipping window .

Advantage of Cohen Sutherland Line Clipping:

1. It calculates end-points very quickly and rejects and accepts lines quickly.
2. It can clip pictures much large than screen size.

Algorithm of Cohen Sutherland Line Clipping:

Step1: Calculate positions of both endpoints of the line

Step2: Perform OR operation on both of these end-points

```
Step3:If the OR operation gives 0000
    Then
        line is considered to be visible
    else
       Perform AND operation on both endpoints
    If And ≠ 0000
       then the line is invisible
     else
    And=0000
  Line is considered the clipped case.
```

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

Given a 3D object with coordinate points A(0, 3, 3), B(3, 3, 6), C(3, 0, 1), D(0, 0, 0). Apply the scaling parameter 2 towards X axis, 3 towards Y axis and 3 towards Z axis and obtain the new coordinates of the object.

Scaling factor along X axis = 2
Scaling factor along Y axis = 3
Scaling factor along Z axis = 3

For Coordinates A(0, 3, 3)
Xnew = Xold x Sx = 0 x 2 = 0
Ynew = Yold x Sy = 3 x 3 = 9
Znew = Zold x Sz = 3 x 3 = 9
New coordinates of corner A after scaling = (0, 9, 9).

For Coordinates B(3, 3, 6)
Xnew = Xold x Sx = 3 x 2 = 6
Ynew = Yold x Sy = 3 x 3 = 9
Znew = Zold x Sz = 6 x 3 = 18
New coordinates of corner B after scaling = (6, 9, 18).

For Coordinates C(3, 0, 1)
Xnew = Xold x Sx = 3 x 2 = 6
Ynew = Yold x Sy = 0 x 3 = 0
Znew = Zold x Sz = 1 x 3 = 3
New coordinates of corner C after scaling = (6, 0, 3).

For Coordinates D(0, 0, 0)
Xnew = Xold x Sx = 0 x 2 = 0
Ynew = Yold x Sy = 0 x 3 = 0
Znew = Zold x Sz = 0 x 3 = 0
New coordinates of corner D after scaling = (0, 0, 0).

Step4: If a line is clipped case, find an intersection with boundaries of the window
$m=(y_2-y_1)(x_2-x_1)$

(a) If bit 1 is "1" line intersects with left boundary of rectangle window
$y_3=y_1+m(x-X_1)$
where $X = X_{wmin}$
where X_{wmin} is the minimum value of X co-ordinate of window

(b) If bit 2 is "1" line intersect with right boundary
$y_3=y_1+m(X-X_1)$
where $X = X_{wmax}$
where X more is maximum value of X co-ordinate of the window

(c) If bit 3 is "1" line intersects with bottom boundary
$X_3=X_1+(y-y_1)/m$
where $y = y_{wmin}$
y_{wmin} is the minimum value of Y co-ordinate of the window

(d) If bit 4 is "1" line intersects with the top boundary
$X_3=x1+(y-y_1)/m$
where $y = y_{wmax}$
y_{wmax} is the maximum value of Y co-ordinate of the window

Sutherland-Hodgeman Polygon Clipping:

It is performed by processing the boundary of polygon against each window corner or edge. First of all entire polygon is clipped against one edge, then resulting polygon is considered, then the polygon is considered against the second edge, so on for all four edges.

Four possible situations while processing

1. If the first vertex is an outside the window, the second vertex is inside the window. Then second vertex is added to the output list. The point of intersection of window boundary and polygon side (edge) is also added to the output line.
2. If both vertexes are inside window boundary. Then only second vertex is added to the output list.

3. If the first vertex is inside the window and second is an outside window. The edge which intersects with window is added to output list.
4. If both vertices are the outside window, then nothing is added to output list.

Disadvantage of Cohen Hodgmen Algorithm:

This method requires a considerable amount of memory. The first of all polygons are stored in original form. Then clipping against left edge done and output is stored. Then clipping against right edge done, then top edge. Finally, the bottom edge is clipped. Results of all these operations are stored in memory. So wastage of memory for storing intermediate polygons.

Sutherland-Hodgeman Polygon Clipping Algorithm :

1) Read coordinates of all vertices of the polygon.
2) Read coordinates of the clipping window
3) Consider the left edge of the window
4) Compare the vertices of each edge of the polygon, individually with the clipping plane
5) Save the resulting intersections and vertices in the new list of vertices according to four possible relationships between the edge and the clipping boundary discussed earlier.
6) Repeat the steps 4 and 5 for remaining edges of the clipping window. Each time the resultant list of vertices is successively passed to process the next edge of the clipping window.
7) Stop.

PROJECTIONS

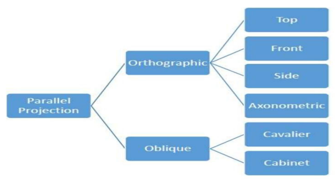
Parallel Projection: coordinate positions are transformed to the view plane along parallel lines .

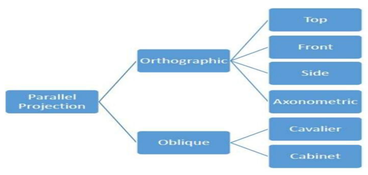
Perspective Projection: coordinate positions are transferred to the view plane along lines that converge to a point called the projection reference point (or center of projection).

Projected view of an object is determined by calculating the intersection of the projection lines with the view plane.

Parallel projection preserves relative proportions of objects , and accurate views of the various sides of an object are obtained with a parallel projection ,but does not give a realistic representation of the 3D object.

Perspective projection produces a realistic view but does not preserve relative proportions.





PERSPECTIVE PROJECTION

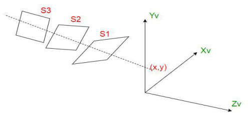
The distance and angles are not preserved and parallel lines do not remain parallel. Instead, they all converge at a single point called center of projection or projection reference point. There are 3 types of perspective projections which are shown in the following chart.
One point perspective projection is simple to draw.
Two point perspective projection gives a better impression of depth.
Three point perspective projection is most difficult to draw.
Parallel lines that are parallel to the view plane will be projected as parallel lines.
Any set of parallel lines of objects that are not parallel to the projection plane are projected into converging lines .
The point at which set of projected parallel lines appears to converge is called vanishing point .
A different set of projected parallel lines will have a separate vanishing point.
Vanishing point for any set of lines that are parallel to one of the principal axes of an object is referred to as the principal vanishing point.

What is visible Surface detection?

When a picture that contains the non-transparent objects and surfaces are viewed, the objects that are behind the objects that are closer cannot be viewed. To obtain a realistic screen image, these hidden surfaces need to be removed. This process of identification and removal of these surfaces is known as Hidden-surface removal or visible surface detection .
The hidden surface problems can be solved by two methods − Object-Space method and Image-space method.
The Object-space method is implemented in physical coordinate system and image-space method is implemented in screen coordinate system.Object space method compares objects and parts of objects to each other within the scene definition to determine which surfaces as a whole we should label visible .
In image space method visibility is decided point by point at each pixel position on the projection plane .
Most visible surface detection algorithm us image space methods .

Depth Buffer Algorithm.

It is also called a Z Buffer Algorithm. Depth buffer algorithm is simplest image space algorithm. For each pixel on the display screen, we keep a record of the depth of an object within the pixel that lies closest to the observer. In addition to depth, we also record the intensity that should be displayed to show the object.



Algorithm

1.Initialise the depth buffer and refresh buffer so that for all buffer positions(x,y),

depth(x,y)=0 refresh(x,y)=Ibackground

2. For each position on each polygon surface , compare depth values to previously stored values in the depth buffer to determine visibility.

. Calculate the depth z for each (x,y) position on the polygon.

. If z>depth(x,y) , then set

depth(x,y)=z , refresh(x,y) =Isurf(x,y)

I background –value of background intensity

Isurf(x,y) is the projected intensity value for the surface at pixel position (x,y).After all surfaces have been processed , the depth buffer contains depth values for the visible surfaces.

Surface Equation

AB+BY+CZ+D=0

Depth calculation along z axis

Z=(-AX-BY-D)/C

Once obtained depth at a pixel position, in the same scan line (horizontaly) next point depth can be obtained by substituting ‘x ‘ as ‘x+1’

Current point is(x,y) then next point in the same scan line is (x+1,y)

Z’=(-Ax-A-By-D)/C ,

Z’=(-Ax-By-D)/C –A/C=Z-A/C

After finishing one scan line we don’t want to start from the beginning of next scan line , start from polygon edge point of of next line

Vertically next polygon point depth where is (x-1/m,y-1) , after substituting

Depth calculation along z axis

Z=(-AX-BY-D)/C

=-((A(x-1/m)-B(y-1)-D)/C

=(-Ax+A/m-By+B-D)/C

=(-Ax-By-D)/C+(A/m+B)/C

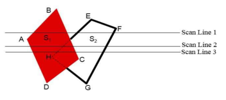
=z +(A/m +B)/C

Advantages:
• Simple to use
• Can be implemented easily in object or image space

Disadvantages:
• Takes up a lot of memory
It is a time-consuming process

SCAN LINE ALGORITHM

It is an image-space method to identify visible surfaces. This is an extension of the scan line algorithm for the polygon filling algorithm.we can process only one surface at a time if we use the depth buffer method.But scan line method can process more than one surfaces at a time.This method has depth information for the only single scan line.
In order to require one scan-line of depth values, we must group and process all polygons intersecting a given scan-line at the same time before processing the next scan-line. Two important tables, edge table and polygon table, are maintained for this.
The Edge Table − It contains coordinate endpoints of each line in the scene, the inverse slope of each line, and pointers into the polygon table to connect edges to surfaces.
The Polygon Table − It contains the plane coefficients, Color information of the particular polygon, and a flag that is initialized to false.



Scan Line Algorithm:

For all polygons (surfaces) intersecting a scan line , the faces are processed form left to right.
If there are overlapping faces, we should determine the depth so that nearest face to the view plane is identified and the intensities of that face is entered into the refresh buffer.
For each scan line an active list of edges {edges intersecting by scan line}of faces is maintained and they are sorted in the order of increasing x values.

In the figure, there are two polygon faces ABCD and EFGH.

For scanline 1,Active Edge List { AB,BC,EH,FG}

It is processed from left to right .
{At left most position the face flag is turned On and on the rightmost position, it is set to OFF.}

Between edges, AB and BC flag for surface S1 is ON, and intensity of surface s1 is entered into refresh buffer.

Between BC and EH both flags of S1 and S2 are OFF and no intensity is stored.

Between EH and EG only flag for S2 is ON and the intensity of Surface S2 is entered into the refresh buffer.For scan line 1 there is no depth calculation since there is no overlapping region.

For scanline 2 ,Active Edge List { AD,EH,BC,FG}

Between AD and BC flag for S1 is ON, and the intensity of S1 is entered into the refresh buffer.Between EH and BC flag for S1 is ON and the flag for S2 is ON, in this case, calculate depth from the viewpoint to these two surfaces S1 and S2, and the intensity of the surface which is nearest to the viewpoint is entered into the refresh buffer .

For scan line 3 we can follow the coherence property here, because scan line 2 and scan line 3 has same active edge list and other regularities also, in such case we don’t want to calculate the depth separately for scan line 3 again, instead follow depth information calculated previously in scan line 2 and store the same intensity details as in case of scan line 2.

Image processing and applications

1)Gamma-ray Imaging: In nuclear medicine injecting a patient with a radioactive isotope emits gamma rays as its decays. Images are collected from the emission collected by gamma-ray decoders. Images of this sort are used to locate sites of bone pathology such as infections or tumors.
2) X-ray Imaging - Medical diagnostics , Angiography , Computerized axial tomography (CT scan) , Industry applications
3) Ultraviolet Imaging - Lithography,Industrial inspection, microscopy, lasers, biological imaging, astronomical observations.
4) Imaging in the visible and Infrared band - light microscopy , astronomy , remote sensing , industry , law enforcement (biometrics)
5)Imaging in the Microwave Band - Radar –collect data over virtually any region at any time , regardless of weather or ambient lighting conditions.
6) Imaging in the Radio Band -Medicine –magnetic resonance imaging (MRI) , Astronomy

Components of image processing system.

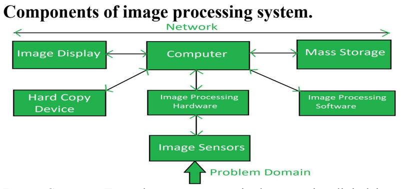


Image Sensors: Two elements are required to acquire digital images,the first is a physical device that is sensitive to the energy radiated by the object we wish to image. The second is a digitizer, which is a device for converting the output of the physical sensing device into digital form.

Specialized Image Processing Hardware: Consists of the digitizer plus hardware that performs other primitive operations, such as an ALU that performs parallel arithmetic and logic operations on entire images.

Computer:
The computer used in the image processing system is the general-purpose computer that is used by us in our daily life. It can range from a PC to a supercomputer. Sometimes custom computers are used to achieve a required level of performance.
Image Processing Software:
Image processing software is software that includes all the mechanisms and algorithms that are used in the image processing system.
Mass Storage:
Mass storage stores the pixels of the images during the processing. An image of size 1024 x 1024 pixels, in which the intensity of each pixel is an 8-bit quantity, requires 1 MB of storage space if the image is not compressed. Providing adequate storage in an image processing system can be a challenge.
Image Display: mainly color TV monitors, in some cases, there are stereo displays and these are implemented in the form of headgear containing two small displays embedded in goggles worn by the user.
Hard Copy Device:
Once the image is processed then it is stored in the hard copy device. It can be a pen drive or any external ROM device.Use laser printers, film cameras, and optical and CD ROM disks.
Network:
Because of the large amount of data inherent in image processing applications, the key consideration in image transmission is bandwidth.

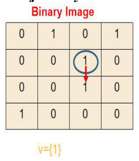
Types of adjacency/connectivity

1. 4-adjacency: Two pixels p and q with values from v are 4-adjacent if q is in the set N4 (p).

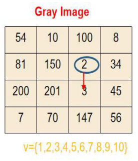
2. 8-adjacency: Two pixels p and q with values from v are 8-adjacent if q is in the set N8 (p).

3. m-adjacency (mixed): two pixels p and q with values from v are m-adjacent if: q is in N4 (p) or q is in ND (P) and the set N4 (p) ∩ N4 (q) has no pixel whose values are from V.

4 adjacency

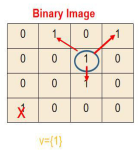


0	1	0	1
0	0	1	0
0	0	1	0
1	0	0	0

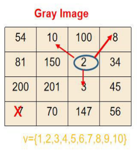


54	10	100	8
81	150	2	34
200	201	3	45
7	70	147	56

8 adjacency

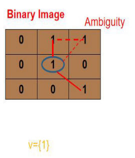


0	1	0	1
0	0	1	0
0	0	1	0
1	0	0	0



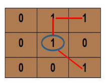
54	10	100	8
81	150	2	34
200	201	3	45
7	70	147	56

m adjacency



0	1	1
0	1	0
0	0	1

If 4 adjacency and 8 adjacency are possible at the same time priority is for 4 adjacency



0	1	1
0	1	0
0	0	1

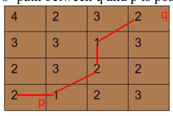
A digital path or curve from a pixel p with coordinates(x,y) to pixel q with coordinates
(s,t) is a sequence of distinct pixels with coordinates
(x0,y0),(x1,y1),…,(xn,yn) where (x0,y0)=(x,y) and (xn,yn)=(s,t) and pixels (xi,yi) and (xi-1,yi-1) are adjacent for 1<=i<=n . Here length of the path=n.
If (x0,y0)=(xn,yn) ,the path is a closed path .
We can define 4- , 8- , or m- paths depending on the type of adjacency.
Egs Define 4-adjacency, 8-adjacency and m-adjacency. Consider the image segment shown.

4	2	3	2 q
3	3	1	3
2	3	2	2
2 p	1	2	3

Let V={1,2} and compute the length of the shortest 4- ,8- and m-path between p and q. If a particular path does not exist between these two points, explain why?

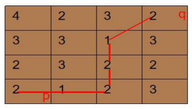
4- path between q and p is not possible, because q has no 4 adjacent pixels with values 1 or 2.

8- path between q and p is possible,



4	2	3	2 q
3	3	1	3
2	3	2	2
2 p	1	2	3

m- path between q and p is possible,



4	2	3	2 q
3	3	1	3
2	3	2	2
2 p	1	2	3

Let S represent a subset of pixels in an image. Two pixels p and q are said to be connected in S if there exists a path between them consisting entirely of pixels in S.

For any pixel p in S, the set of pixels that are connected to p in S is called a connected
component of S.

If it only has one connected component, then set S is called a connected set.

There are three types of connectivity on the basis of adjacency. They are:

a) 4-connectivity: Two or more pixels are said to be 4-connected if they are 4-adjacent with each others.

b) 8-connectivity: Two or more pixels are said to be 8-connected if they are 8-adjacent with each others.

c) m-connectivity: Two or more pixels are said to be m-connected if they are m-adjacent with each others.

CONVOLUTION

Another way of dealing images

Here we are going to discuss another method of dealing with images. This other method is known as convolution. Usually the black box(system) used for image processing is an LTI system or linear time invariant system. By linear we mean that such a system where output is always linear , neither log nor exponent or any other. And by time invariant we means that a system which remains same during time.

So now we are going to use this third method. It can be represented as.

Image
Convolution (*)
Image

It can be mathematically represented as two ways

g(x,y) = h(x,y) * f(x,y)

It can be explained as the "mask convolved with an image".Or **g(x,y) = f(x,y) * h(x,y)**It can be explained as "image convolved with mask".

There are two ways to represent this because the convolution operator(*) is commutative. The h(x,y) is the mask or filter.In order to perform convolution on an image, following steps should be taken.

- Flip the mask (horizontally and vertically) only once
- Slide the mask onto the image.
- Multiply the corresponding elements and then add them
- Repeat this procedure until all values of the image has been calculated.

Example of convolution

Let's perform some convolution. Step 1 is to flip the mask.

Mask

Let's take our mask to be this.

1	2	3
4	5	6
7	8	9

Flipping the mask horizontally

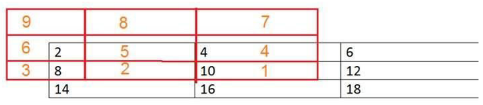
3	2	1
6	5	4
9	8	7

Flipping the mask vertically

9	8	7
6	5	4
3	2	1

Convolution

Convolving mask over image. It is done in this way. Place the center of the mask at each element of an image. Multiply the corresponding elements and then add them , and paste the result onto the element of the image on which you place the center of mask.



The box in red color is the mask, and the values in the orange are the values of the mask. The black color box and values belong to the image. Now for the first pixel of the image, the value will be calculated as

First pixel = (5*2) + (4*4) + (2*8) + (1*10)

= 10 + 16 + 16 + 10

= 52

Place 52 in the original image at the first index and repeat this procedure for each pixel of the image.

11. Knowledge Base

Knowledge is the last stage in DIP. In this stage, important information of the image is located, which limits the searching processes. The knowledge base is very complex when the image database has a high-resolution satellite.

Spatial and Gray Level Resolution

Spatial resolution states that the clarity of an image cannot be determined by pixel resolution (number of pixels in an image). Spatial resolution can be defined as the number of independent pixels values per inch.

Dots per inch

Dots per inch or DPI is usually used in monitors.

Lines per inch

Lines per inch or LPI is usually used in laser printers.

Pixel per inch

Pixel per inch or PPI is measure for different devices such as tablets , Mobile phones e.t.c.

Gray level resolution

Gray level resolution refers to the predictable or deterministic change in the shades or levels of gray in an image.

In short gray-level resolution is equal to the number of bits per pixel (BPP).

The number of different colors in an image depends on the depth of color or bits per pixel.

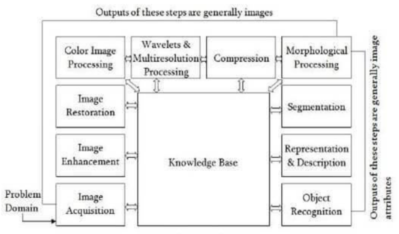
The mathematical relation that can be established between gray level resolution and bits per pixel can be given as.

$$L = 2^k$$

L – number of gray levels , K-bits per pixel

FUNDAMENTALS OF DIGITAL IMAGE PROCESSING

Following are Fundamental Steps of Digital Image Processing:



1. Image Acquisition

Image acquisition is the first step of the fundamental steps of DIP. In this stage, an image is given in the digital form. Generally, in this stage, pre-processing such as scaling is done.

2. Image Enhancement

Image enhancement is the simplest and most attractive area of DIP. In this stage details which are not known, or we can say that interesting features of an image is highlighted. Such as brightness, contrast, etc...

3. Image Restoration

Image restoration is the stage in which the appearance of an image is improved.

4. Color Image Processing

Region-based Segmentation Approach - Region Growing, Region Splitting and Merging

A region can be classified as a group of connected pixels exhibiting similar properties. The similarity between pixels can be in terms of intensity, color, etc. In this type of segmentation, some predefined rules must be obeyed by a pixel to be classified into similar pixel regions. Region-based segmentation methods are preferred over edge-based segmentation methods in case of a noisy image. Region-Based techniques are further classified into 2 types based on the approaches they follow.

Region growing method

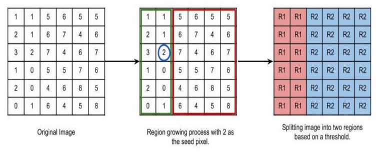
In the case of the Region growing method, we start with some pixel as the seed pixel and then check the adjacent pixels.

If the adjacent pixels abide by the predefined rules, then that pixel is added to the region of the seed pixel and the following process continues till there is no similarity left. This method follows the bottom-up approach.

In case of a region growing, the preferred rule can be set as a threshold.

For example: Consider a seed pixel of 2 in the given image and a threshold value of 3, if a pixel has a value less than 3 then it will be considered inside the seed pixel region. Otherwise, it will be considered in another region.

Hence 2 regions are formed in the following image based on a threshold value of 3.


Original Image
Region growing process with 2 as the seed pixel.
Splitting image into two regions based on a threshold.

Color image processing is a famous area because it has increased the use of digital images on the internet. This includes color modeling, processing in a digital domain, etc....

5. Wavelets and Multi-Resolution Processing

In this stage, an image is represented in various degrees of resolution. Image is divided into smaller regions for data compression and for the pyramidal representation.

6. Compression

Compression is a technique which is used for reducing the requirement of storing an image. It is a very important stage because it is very necessary to compress data for internet use.

7. Morphological Processing

This stage deals with tools which are used for extracting the components of the image, which is useful in the representation and description of shape.

8. Segmentation

In this stage, an image is a partitioned into its objects. Segmentation is the most difficult tasks in DIP. It is a process which takes a lot of time for the successful solution of imaging problems which requires objects to identify individually.

9. Representation and Description

Representation and description follow the output of the segmentation stage. The output is a raw pixel data which has all points of the region itself. To transform the raw data, representation is the only solution. Whereas description is used for extracting information's to differentiate one class of objects from another.

10. Object recognition

In this stage, the label is assigned to the object, which is based on descriptors.

Region splitting and merging method

In Region splitting, the whole image is first taken as a single region. If the region does not follow the predefined rules, then it is further divided into multiple regions (usually 4 quadrants) and then the predefined rules are carried out on those regions in order to decide whether to further subdivide or to classify that as a region. The following process continues till there is no further division of regions required i.e every region follows the predefined rules.

In Region merging technique, we consider every pixel as an individual region. We select a region as the seed region to check if adjacent regions are similarly based on predefined rules. If they are similar, we merge them into a single region and move ahead in order to build the segmented regions of the whole image.

Edge Detection - Edge Operators- Sobel and Prewitt

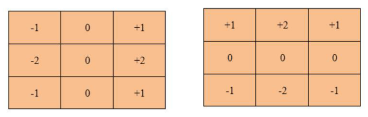
Edge detection is a technique of image processing used to identify points in a digital image with discontinuities, simply to say, sharp changes in the image brightness. These points where the image brightness varies sharply are called the edges (or boundaries) of the image.With the help of edges and lines, an object’s structure is known. That is why extracting the edges is a very important technique in graphics processing and feature extraction.

Sobel Edge Detection Operator

The Sobel edge detection operator extracts all the edges of an image, without worrying about the directions. The main advantage of the Sobel operator is that it provides differencing and smoothing effect.

Sobel edge detection operator is implemented as the sum of two directional edges. And the resulting image is a unidirectional outline in the original image

Sobel Edge detection operator consists of 3x3 convolution kernels. Gx is a simple kernel and Gy is rotated by 90°.These Kernels are applied separately to the input image.



-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

Advantages

Simple and time-efficient computation

Very easy at searching for smooth edges

Limitations:
Highly sensitive to noise

Not very accurate in edge detection

Detect with thick and rough edges does not give appropriate results

Diagonal direction points are not preserved always

Prewitt operator-This operator is almost similar to the sobel operator. It also detects vertical and horizontal edges of an image. It is one of the best ways to detect the orientation and magnitude of an image.

$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad M_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Advantages:
Good performance on detecting vertical and horizontal edges

Best operator to detect the orientation of an image

Negative transformation

The second linear transformation is negative transformation, which is an invert of identity transformation.

In negative transformation, each value of the input image is subtracted from the L-1 and mapped onto the output image.

In this the following transition has been done.

s = (L – 1) – r

since the input image of Einstein is an

8 bpp image, so the number of levels in this image are 256.

Putting 256 in the equation, we get s = 255 – r

Power–Law (Gamma) transformations

Power Law Transformation is of two types of transformation nth power transformation and nth root transformation

s=cr^γ

Variation in the value of γ varies the enhancement of the images. C is constant.Different display devices / monitors have their own gamma correction, that’s why they display their image at different intensity.This type of transformation is used for enhancing images for a different types of display devices. (Gamma Correction)The gamma of different display devices is different.For example, Gamma of CRT lies in between 1.8 to 2.5,