-

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING

CS6513 – COMPUTER GRAPHICS LABORATORY

V SEMESTER - R 2013

LABORATORY MANUAL

Name : _____

Register No. : _____

Section : _____

VISION

College of Engineering is committed to provide highly disciplined, conscientious and enterprising professionals conforming to global standards through value based quality education and training.

MISSION

- To provide competent technical manpower capable of meeting requirements of the industry
- To contribute to the promotion of Academic Excellence in pursuit of Technical Education at different levels
- To train the students to sell his brawn and brain to the highest bidder but to never put a price tag on heart and soul

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION

To strive for acquiring, applying and imparting knowledge in Computer Science and Engineering through quality education and to provide enthusiastic professionals with commitment

MISSION

- To educate the students with the state-of-art technologies to meet the growing challenges of the electronics industry
- To carry out research through continuous interaction with research institutes and industry, on advances in communication systems
- To provide the students with strong ground rules to facilitate them for systematic learning, innovation and ethical practices

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

1. Fundamentals

To impart students with fundamental knowledge in Mathematics, Science and fundamentals of Engineering that will would them to be successful professionals

2. Core Competence

To provide students with sound knowledge in engineering and experimental skills to identify complex software problems in industry and to develop practical solution for them

3. Breadth

To provide relevant training and experience to bridge the gap between theory and practice this enables to find solutions for real time problem in industry and organization and to design products requiring interdisciplinary skills

4. Professionalism skills

To bestow students with adequate training and provide opportunities to work as team that will build up their communication skills, individual leadership and supportive qualities and to develop them to adapt and work in ever changing technologies

5. Lifelong Learning

To develop the ability of students to establish themselves as professionals in Computer Science and Engineering and to create awareness about the need for lifelong learning and pursuing advanced degrees

PROGRAMME OUTCOMES (POs)

a) To apply basic knowledge of Mathematics, Science and engineering fundamentals in Computer Science and Engineering field

b) To design and conduct experiments as well as to analyze and interpret and apply the same in the career

c) To design and develop innovative and creative software applications

d) To understand a complex real world problems and develop an efficient practical solutions

e) To create, select and apply appropriate technique, resources, modern engineering and IT tools

f) To understand their roles as professionals and give the best to the soicety

g) To develop a system that will meet expected need with realistic constraints such as economical, environmental, social, political, ethical, safe and sustainable

h) To communicate effectively and make others understand exactly what they are trying to convey in both verbal and written forms

i) To engage lifelong learning and exhibit their technical skills

j) To develop and manage projects in multidisciplinary environments

CS6513 – COMPUTER GRAPHICS LABORATORY

SYLLABUS

COURSE OBJECTIVES

- Understand graphics programming.
- Be exposed to creation of 3D graphical scenes using open graphics library suits.
- Be familiar with image manipulation, enhancement.
- Learn to create animations.
- To create a multimedia presentation/game/project

LIST OF EXPERIMENTS:

1. Implementation of algorithms for drawing 2D primitives – Line (DDA, Bresenham's) – all slopes circle (midpoint).
2. 2D geometric transformations – translation, rotation, scaling, reflection, shear, window to viewport.
3. Composite 2D transformations.
4. Line clipping.
5. 3D transformations - translation, rotation, scaling.
6. 3D projections – parallel, perspective.
7. Creating 3D scenes.
8. Image editing and manipulation - basic operations on image using any image editing software.
9. Creating gif animated images, image optimization.
10. 2D Animation – to create Interactive animation using any authoring tool

COURSE OUTCOMES

- Create 3D graphical scenes using open graphics library suits.
- Implement image manipulation and enhancement.
- Create 2D animations using tools.

CS6513 – COMPUTER GRAPHICS LABORATORY

CONTENTS

Sl. No.	Name of the Experiment	Page No.
CYCLE 1 – EXPERIMENTS		
A	**Implementation of Algorithms for drawing 2D Primitives**	
1	Implementation of DDA line drawing algorithm	7
2	Implementation of Bresenham's line drawing algorithm	9
3	Implementation of Bresenham's circle drawing algorithm	11
B	**2D Geometric Transformations**	
4	Implementation of two dimensional basic transformations – Translation, Rotation, Scaling	14
5	Implementation of two dimensional transformations – Reflection and Shear	19
6	Implementation of window – to – viewport mapping	23
7	Implementation of composite 2D transformations	26
8	Implementation of Cohen Sutherland line clipping algorithm	30
9	Implementation of Three Dimensional Transformations - Translation, Rotation, Scaling	33
10	Implementation of 3D image projections	36
11	Creation of 3D scenes	40
CYCLE 2 – EXPERIMENTS		
C	**Image Editing and Manipulation**	
12.	Implementation of basic operations on image using Photoshop	43
13.	Creation of gif animated images	45
14	Optimizing an image	47
15	Create a 2D interactive animation using Flash	55
ADDITIONAL EXPERIMENTS BEYOND THE SYLLABUS		
16	Generate a 2D image and add motion to it using C	58
17	Implementation of Bresenham's ellipse drawing algorithm	61
18	Implementation of line, circle and ellipse attributes	63
19	Implementation of Sutherland Hodgeman polygon clipping algorithm	66

Expt. No. 1

IMPLEMENTATION OF ALGORITHMS FOR DRAWING 2D PRIMITIVES

LINE DRAWING USING DDA ALGORITHM

Aim:

To write a C program to draw a line using DDA algorithm

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Read the starting and ending coordinates xa, ya, xb, and yb,
3. Find the x-coordinate difference and y-coordinate difference
   dx = xb - xa & dy = yb – ya,
4. Compare the difference and decide the step value
   if (dx > dy) step = dx
   else
   step = dy,
5. Find the increment values of coordinates
   xi = dx / step
   yi = dy / step,
6. Display the starting point using the function putpixel(xa,ya,4),
7. Find adjacent pixels using the formula xa = xa + xi & ya = ya + yi,
8. Repeat the steps till reaching the endpoints i.e., ya = yb & xa = xb,
9. Stop the program.

Sample Output:

Enter the starting coordinates: 100 100

Enter the ending coordinates: 200 200

Fig.No 1 Line using DDA

Result:

Thus the line was drawn successfully using DDA algorithm in C.

Outcome:

Thus the outcome of implementing 2D primitives has been attained.

Application:

- Image processing
- Computer art
- Presentation graphics

Viva-voce

1. What are the advantages of DDA algorithm?
2. Define – Computer Graphics
3. What are the properties of video display devices?
4. What are the various applications of computer graphics?
5. What is resolution?
6. What is a bitmap?
7. List out the important characteristics of video display device.
8. What is meant by pixel?
9. What is intensity?
10. Define – DDA Algorithm

Expt. No. 2

LINE DRAWING USING BRESENHAM'S ALGORITHM

Aim:

To write a program in C to draw a line using Bresenham's algorithm

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Read the starting and ending coordinates xa, ya, xb, yb,
3. Find the x-coordinate difference and y-difference

   dx = xb - xa & dy = yb – ya,
4. Calculate decision parameter 'p' value

   p = 2dy-dx,
5. Fix the starting and ending coordinates

   if(xa < xb)

   xstart = xa ystart = ya xend = xb

   yend = yb

   else

   xstart = xb

   ystart = yb

   xend = xa

   yend = ya,
6. Display the starting point using the function putpixel(xa,ya,4),
7. Find adjacent pixels and display it using the formula given below

   x =xstart & y=ystart

   while(x < xend)

   x = x + 1

   if(p < 0)

```
p = p + 2 * dy
else
 p = p + 2 * (dy - dx)
 y = y + 1
  putpixel(x,y,1),
```

8. Repeat the step 7 till reaching the end points,
9. Stop the program.

Sample Output:

Enter the xa & ya value: 200 200

Enter the xb & yb value: 350 45

Fig.No 2 Line using Bresenham's

Result:

Thus a line is drawn successfully using Bresenham's algorithm in C

Outcome:

Thus the outcome of implementing 2D primitives has been attained.

Application:

- Image processing
- Computer art
- Presentation graphics

Viva-voce

1. What is the property that reduces the pixel calculation in Bresenham's circle drawing algorithm?
2. What is the equation used to find decision parameter in Bresenham's line drawing algorithm?
3. What is meant by rasterization?
4. List out the advantages and disadvantages of DVST.
6. What are the two techniques for producing color displays with a CRT?
7. What is vertical retrace of the electron beam?
8. What is meant by frame buffer?
9. Distinguish between track ball and space ball.
10. Define – Digitizers

Expt. No. 3

CIRCLE DRAWING USING BRESENHAM'S CIRCLE ALGORITHM

Aim:

To draw a circle using Bresenham's circle drawing algorithm in C

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Get the radius and center of the circle r, xc, yc,
3. Obtain the first point on the circumference of a circle centered on the origin as $(X_0, Y_0) = (0, r)$,
4. Calculate the initial value of the decision parameter as $p = 5/4 - r$,
5. At each x_k position, starting at k=0, perform the following test if($p_k < 0$),
   the next point along the circle centered on (0,0) is (x_{k+1}, y_{k+1}) and $p_{k+1} = p_k + 2x_{k+1} + 1$
   Otherwise the next point along the circle is (x_{k+1}, y_{k-1}) and
   $p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}$,where $2x_{k+1} = 2x_k+2$ and $2y_{k+1} = 2y_k - 2$.
6. Determine symmetry points in other seven octants,
7. Move each calculated pixel position (x, y) onto the circular path centered on (xc, yc) and plot the coordinat
   values x = x + xc & y = y + yc,
8. Repeat the steps 5 to 7 until x >= y,
9. Stop the program.

Sample Output:

Enter the xa value: 200

Enter the ya value: 200

Enter the radius: 50

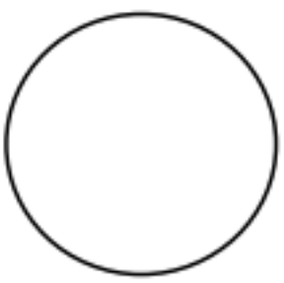


Fig.No 3 Circle using Bresenham's

Result:

Thus the circle was drawn successfully using Bresenham's circle drawing algorithm in C.

Outcome:

Thus the outcome of implementing 2D primitives has been attained.

Application:

- Image processing
- Computer art
- Presentation graphics

Viva-voce

1. What are the two basic techniques for producing color display with a CRT?
2. Give three difference between shadows mask and beam penetration method.
3. Differentiate LCD from LED.
4. Differentiate plasma panel display from thin film electroluminescent display.
5. Define – Bresenham's Circle Algorithm

Expt. No. 4

2D GEOMETRIC TRANSFORMATIONS

BASIC 2D TRANSFORMATIONS – TRANSLATION, ROTATION, SCALING

Aim:

To write a program to perform the basic 2D transformations like translation, rotation and scaling using transformation equation in C

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Obtain the coordinates of the line xa, ya, xb, yb,
3. Get the translation factors tx, ty, rotation angle a & scaling factors sx, sy,
4. Find the translated coordinates by applying the angle as

   $x' = x + tx$ & $y' = y + ty$.
5. Get the rotation coordinates by applying the angle as

   $x' = abs(xa - xb) \cos a - abs(ya - yb) \sin a$

   $y' = abs(xa - xb) \sin a + abs(ya - yb) \cos a$,
6. Scaling is applied as

   $x' = x * sx$

   $y' = y * sy$,
7. Draw the transformed line with the new coordinates (x', y'),
8. Similarly obtain the coordinates of rectangle / triangle as consecutive set of line end points and apply all the basic transformations,
9. Stop the program.

Sample Output:

Line

1. Translation 2. Scaling 3. Rotation 4. Exit

Enter the choice: 1

Enter the x1 value: 100

Enter the y1 value: 100

Enter the x2 value: 200

Enter the y2 value: 200

Enter the translation factor in x-axis: 50

Enter the translation factor in y-axis: 0

Fig.No 4.0 Translation

1. Translation 2. Scaling 3. Rotation 4. Exit

Enter the choice: 2

Enter the x1 value: 100

Enter the y1 value: 100

Enter the x2 value: 200

Enter the y2 value: 200

Enter the scaling factor in x-axis: 1

Enter the scaling factor in y-axis: 2

Fig.No 4.1 Scaling

1. Translation 2. Scaling 3. Rotation 4. Exit

Enter the choice: 3

Enter the x1 value: 100

Enter the y1 value: 100

Enter the x2 value: 200

Enter the y2 value: 200

Enter the rotation angle: 45

Fig.No 4.2 Rotation

Result:

Thus the program for performing basic 2D transformations using transformation equation is successfully executed in C.

Outcome:

Thus the outcome of implementing 2D geometric transformation has been met.

Application:

- Used in traditional printing
- Drawing technologies

Viva-voce

1. What are the changes accomplished by adding attributes?
2. Which transformation produces a mirror image of an object?
3. Which is not a basic transformation operation?
4. What is transformation?
5. What is a view plane?
6. What are the steps involved in 3D transformation pipeline?
7. What is fixed point scaling?
8. Distinguish between uniform scaling and differential scaling.
9. What are the different kinds of co-ordinate representation?

Expt. No. 5

2D TANSFORMATIONS (REFLECTION AND SHEARING)

Aim:

To write a program to perform the other 2D transformations like reflection and shearing in C

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. For reflection obtain the coordinates of the triangle xa, ya, xb, yb, xc, yc,
3. Calculate the reflection as, x` = x + 2 * (320 - x) and y` = y + 2 * (240 - y),
4. For shearing obtain the coordinates of the square xa, ya, xb, yb, xc, yc, xd, yd,
5. Shearing points can be calculated as x` = x + shx & y` = y + shy, where shx, shy are the shearing factors,
6. Draw the transformed objects with the new coordinates (x`, y`),
7. Stop the program.

Sample Output:

1. Reflection 2. Shearing 3. Exit

Enter the choice: 1

Enter the xa&ya value: 200 100

Enter the xb&yb value: 200 200

Enter the xc&yc value: 100 200

1. About x-axis 2. About y-axis 3. About both 4. Exit

Enter the choice: 1

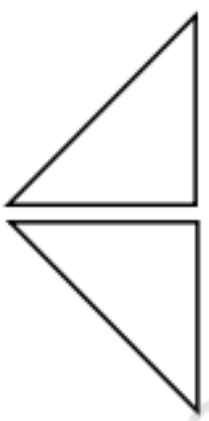


Fig.No 5.0 Reflection about x-axis

1. About x-axis 2. About y-axis 3. About both 4. Exit

Enter the choice: 2

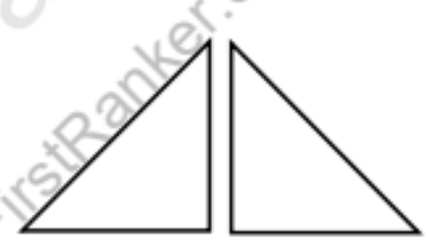


Fig.No 5.1 Reflection about y-axis

1. About x-axis 2. About y-axis 3. About both 4. Exit

Enter the choice: 3

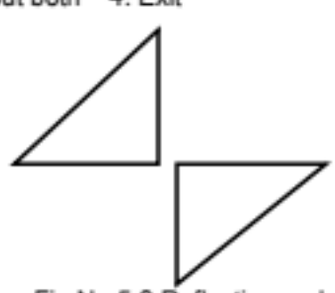


Fig.No 5.2 Reflection on both

1. About x-axis 2. About y-axis 3. About both 4. Exit

Enter the choice: 4

1. Reflection 2. Shearing 3. Exit

Enter the choice: 2

Enter the xa&ya value: 200 200

Enter the xb&yb value: 300 200

Enter the xc&yc value: 300 300

Enter the xd&yd value: 200 300

1. About x-axis 2. About y-axis 3. About both 4. Exit

Enter the choice: 1

Enter the shearing factor for x: 50

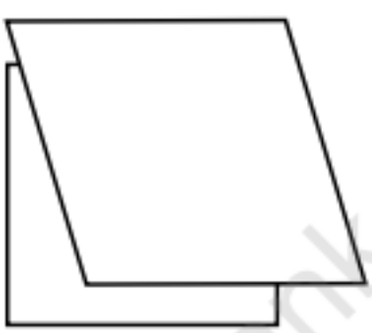


Fig.No 5.3 Shearing about x-axis

1. About x-axis 2. About y-axis 3. About both 4. Exit

Enter the choice: 2

Enter the shearing factor for y: 50

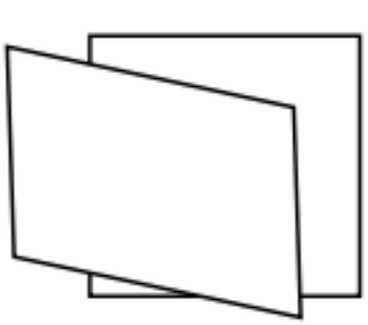


Fig.No 5.4 Shearing about y-axis

1. About x-axis 2. About y-axis 3. About both 4. Exit

Enter the choice: 3

Enter the shearing factor for x: 50

Enter the shearing factor for y: 50

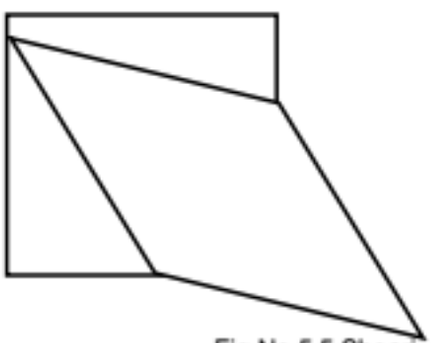


Fig.No 5.5 Shearing with both axis

Enter the choice: 4

Result:

Thus the program for performing 2D transformations reflection and shearing is successfully executed in C.

Outcome:

Thus the outcome of implementing 2D geometric transformation has been met.

Application:

- Typography
- Cartography
- Technical drawing.

Viva-voce

1. What is meant by reflection?
2. Define – Shearing
3. What are the applications for reflection?
4. Compare reflection from mirroring.
5. List out the advantages of using reflection.

Expt. No. 6

WINDOW – TO – VIEWPORT TRANSFORMATION

Aim:

To write a C program to perform window-to-viewport transformation

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Input the minimum and maximum coordinates of a window,
3. Input the minimum and maximum coordinates of a viewport,
4. Input the coordinates of image to be displayed,
5. Perform the scaling to transform the image to window and to viewport,
6. Stop the program.

Sample Output:

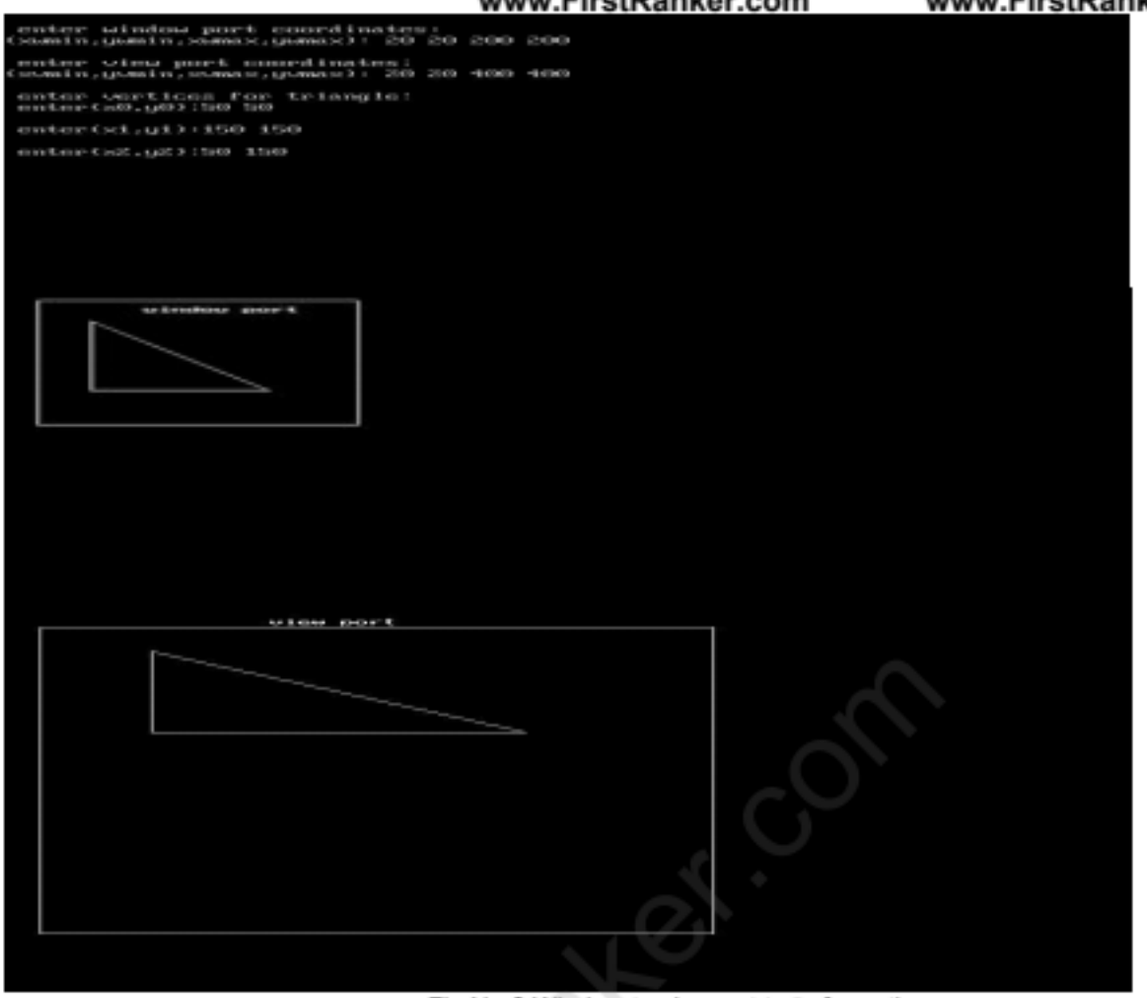


Fig.No 6 Window to viewport transformation

Result:

Thus the program for performing window-to-viewport transformation is successfully executed in C.

Outcome:

Thus the outcome of implementing 2D geometric transformation has been met.

Application:

- To draw maps, sketch of areas and buildings
- Visualization

VIVA - VOCE

1. Distinguish between window port and view port.
2. What is the need of homogeneous coordinates?
3. List out three font editing tools.
4. Distinguish between window port and view port.
5. Define – Clipping
6. What is the need for homogeneous coordinates?
7. List out two output primitives function.
8. What is a decision parameter?
9. List out the two software standards.
10. Define – Bitmap
11. Define – Pixelmap

Expt. No. 7

COMPOSITE 2D TRANSFORMATIONS

Aim:

To write a program to perform the composite 2D transformations like successive translation, rotation and scaling using transformation equation in C

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Obtain the coordinates of the line xa, ya, xb, yb,
3. Get the translation factors tx1, ty1, tx2, ty2 rotation angles a1, a2 & scaling factors sx1, sy1, sx2, sy2,
4. Find the translated coordinates by applying the formula as below.

   $x' = x + (tx1 + tx2)$ & $y' = y + (ty1 + ty2)$.

   It possesses associative property and successive translations are proved as additive.
5. Get the rotation coordinates by applying the formula as

   $x' = abs(xa - xb) \cos(a1 + a2) - abs(ya - yb) \sin(a1 + a2)$

   $y' = abs(xa - xb) \sin(a1 + a2) + abs(ya - yb) \cos(a1 + a2)$

   It possesses associative property and successive rotations are also proved as additive.
6. Scaling is applied as

   $x' = x * (sx1 * sx2)$

   $y' = y * (sy1 * sy2)$

   It possesses associative property and successive scaling is multiplicative.
7. Draw the transformed line with the new coordinates (x', y'),
8. Stop the program.

Sample Output:

1.Successive Translation 2. Successive Scaling 3. Successive Rotation 4.Exit

Enter the choice: 1

Enter the x1 value: 100

Enter the y1 value: 100

Enter the x2 value: 200

Enter the y2 value: 200

Enter the translation factor in x-axis: 25 25

Enter the translation factor in y-axis: 0 0

Fig.No 7.0 Successive Translation

1.Successive Translation 2. Successive Scaling 3. Successive Rotation 4.Exit

Enter the choice: 2

Enter the x1 value: 100

Enter the y1 value: 100

Enter the x2 value: 200

Enter the y2 value: 200

Enter the scaling factor in x-axis: 0.5 0.5

Enter the scaling factor in y-axis: 1 1

Fig.No 7.1 Successive Scaling

1.Successive Translation 2. Successive Scaling 3. Successive Rotation 4.Exit

Enter the choice: 3

Enter the x1 value: 100

Enter the y1 value: 100

Enter the x2 value: 200

Enter the y2 value: 200

Enter the rotation angle: 20 25

Fig.No 7.2 Successive Rotat

Result:

Thus the program for performing composite 2D transformations using transformation equation is successfully executed in C.

Outcome:

Thus the outcome of implementing composite 2D transformation has been met.

Application:

- oracle documentation
- printers
- pixel based display
- monitors

Viva-voce

1. What is transformation?
2. What is translation?
3. What is rotation?
4. What is scaling?
5. What is shearing?
6. What is reflection?
7. What are the different types of line type attributes?
8. What is pixel mask?
9. What is the area-fill attribute?
10. What is meant by homogeneous coordinates?
11. Define – Geometric Transformation
12. What is meant by translation vector?
13. Define – Scaling Factors
14. Define – Composite Transformation
15. List some examples for rigid-body transformation matrix.
16. Write down the syntax to translate transformation matrices.
17. What are the raster functions in graphics packages?
18. What are the results of performing two successive block transfers into the same area of a frame buffer using the binary arithmetic operations?
19. Write the routine to implement scaling as a raster transformation of a pixel block.
20. What is meant by frame buffer?

Expt. No. 8

LINE CLIPPING USING COHEN – SUTHERLAND ALGORITHM

Aim:

To write a program in C to clip a line using Cohen-Sutherland line clipping algorithm

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Obtain the coordinates of the clipping window xmin, ymin, xmax, ymax,
3. Get the coordinates of the line to be clipped xa, ya, xb, yb,
4. Region codes are assigned to the line end points according to relative position with respect to the clipping rectangle.
5. The two region codes of the line end points which passes through the clipping rectangle.
6. If the result is one (1 1 1 1), then the line is entirely outside the clipping window,
7. If the result is zero (0 0 0 0), then the line is completely inside the window hence we can save it for display,
8. For the intermediate results we have to find intersection point using line equation. The slope of the line is given by the equation m = (yb - ya) / (xb - xa),
9. The point at which the line intersects the clipping window can be obtained using the equation $x' <= x_1 + m(y - y_1)$ & $y' <= y_1 + m(x - x_1)$, where x is set either xmin or xmax and y is either ymin or ymax,
10. Using the intersection point the part of the outside the clipping window are clipped off,
11. Stop the program.

Sample Output:

Enter the coordinates for rectangle: 200 200 300 300

Enter the coordinates for line:150 200 350 450

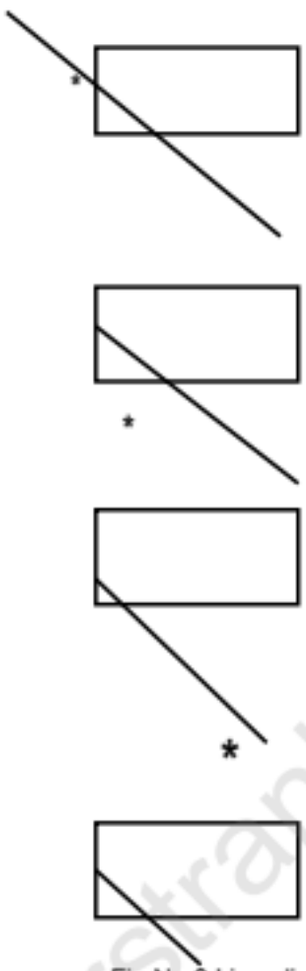


Fig.No 8 Line clip using Cohen-Sutherland algorithm

Result:

Thus the program to clip a line using Cohen-Sutherland algorithm has written and successfully executed in C.

Outcome:

Thus the outcome of implementing Cohen Sutherland line clipping algorithm has been met.

Application:

- Separation of synchronizing signals from composite picture signals
- Clip excessive noise spikes in FM transmitters
- Extracting part of a defined scene for viewing.
- Drawing and painting operations

Viva-voce

1. Define – Clipping
2. Distinguish between window port and view port.
3. What is covering (exterior clipping)?
4. List out the various Text clipping.
5. What are the various representation schemes used in three dimensional objects?
6. Define – Clip window
7. What are primitive types for clipping?
8. What is meant by point clipping?
9. What is meant by line clipping?
10. Differentiate Liang-Barsky Line clipping from Cohen Sutherland line clipping.
11. What is meant by vector method?
12. Define – Polygon Clipping
13. Write down the function for displaying a filled polygon.
14. What do you mean by view plane?
15. What are applications for clipping?
16. What is meant by exterior clipping?
17. Define – Curve Clipping
18. Write down the syntax for set view index?
19. What is meant by workstation transformation?
20. What is meant by view up vector?

Expt. No.9

3D TRANSFORMATION – TRANSLATION, ROATATION, SCALING

Aim:

To write a C program to perform translation, rotation and scaling on 3D objects

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Store the coordinate values in a homogeneous matrix,
3. Draw a 3D object with a specified coordinate value stored in a homogeneous matrix,
4. Perform translation with the use of translation matrix given below

$$\begin{matrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{matrix}$$

where tx, ty, tz are the translation factors,

5. Perform rotation with the rotation matrix given by

$$\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & cost & -sint & 0 \\ 0 & sint & cost & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

6. Perform scaling with the scaling matrix given by

$$\begin{matrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

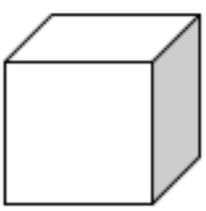
where sx, sy, sz are scaling factors,

7. Stop the program.

Sample Output:

Enter the translation factor: 50 50

Before Translation



After Translation

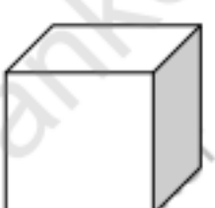


Fig.No 9 3D-Translation

Result:

Thus the program to perform translation, rotation and scaling on 3D objects were written and executed successfully in C.

Outcome:

Thus the outcome of implementing 3D transformations has been met.

Application:

- To create motions in images
- Video games
- Image processing
- Presentation graphics
- Visualization

Viva-voce

1. What are the steps involved in 3D transformation?
2. What do you mean by view plane?
3. What you mean by parallel projection?
4. What is the various representation schemes used in three dimensional objects?
5. What are the steps involved in 3D transformation pipeline?
6. What is view reference point?
7. What is vector dot product?
8. Write the parametric form for the line passing through the points (0,0,0) and (1,2,3)
9. Give an equation for the following planes (0,0,0) and normal to vector [0,1,0].
10. What is meant by perspective projection?
11. Define – Depth Cueing
12. What are the 3D display methods?
13. What are the techniques to achieve realism in computer graphics?
14. Define – Geometric Table
15. Define – Attribute Table
16. What is meant by curved lines?
17. What is meant by quadric surfaces?
18. Write short notes on polygon surface.
19. Write short notes on polygon tables.
20. Define – Curves

Expt. No.10

IMPLEMENTATION OF 3D IMAGE PROJECTIONS

Aim:

To implement a C program for projection on 3D image

Software requirements:

C, C++ compilers, Java, OpenGL

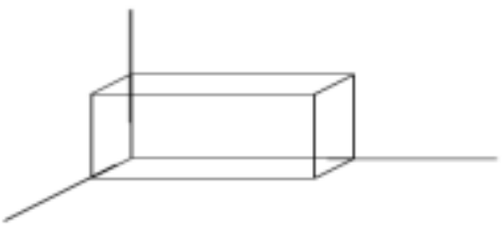
Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

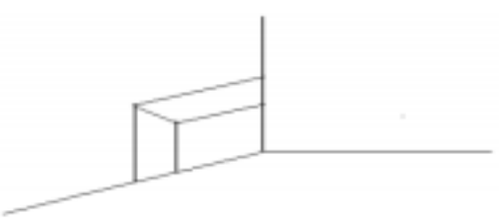
Algorithm:

1. Start the program.
2. Draw any image in the three dimensional plane,
3. Get the choice of axis as input from the user,
4. Perform the projection about the desired axis,
5. Display the projected image,
6. Stop the program.

Sample Output:



Projection about X-axis :
Enter the value of P : 45



Projection about Z-axis :
Enter the value of R : 45

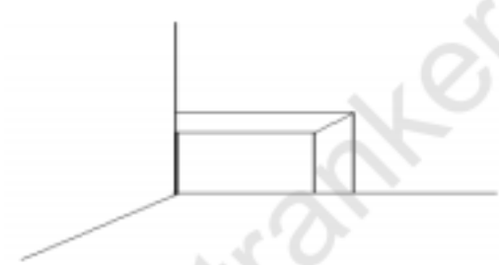


Fig.No 10 3D-Projections

Result:

Thus the projections on the three dimensional images was performed successfully and the output was verified.

Outcome:

Thus the outcome of implementing 3D image projection has been met.

Application:

- Simulation tools for Soft body dynamics including mesh collision detection
- Real time control during physics simulation and rendering
- Internal render engine with scanline ray tracing etc.

Viva-voce

1. Define – Projection
2. What is meant by perspective projection?
3. What is vanishing point?
4. State the basic types of projections.
5. What is boundary representation?
6. What is space-partitioning representation?
7. What is meant by parallel projection?
8. What is blobby object?
9. What is meant by polygon mesh?
10. What is Spline?
11. Define – Spline Curves
12. List any two properties of Bezier curve.
13. List any two properties of B-spline curve.
14. List any advantages of B-spline curve
15. Make a comparison of beizer and b-spline algorithms for curve generation
16. What is meant by Uniform periodic B-Spline curves.
17. What is blobby object?
18. What is meant by polygon mesh?
19. What is space –partitioning representation?
20. Why cubic Bezier curves are chosen?

Expt. No. 11

CREATION OF 3D SCENES

Aim:

To write a C program to compose a scene by using three dimensional objects

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD ore

Algorithm:

1. Start the program.
2. Initialize graphics functions,
3. Use the function bar3d to depict three dimensional bars on the screen,
4. Call sphere in the program by specifying necessary variables,
5. Compose a scene in which the bars and the sphere move on the screen by using the relevant control structure,
6. Stop the program.

Sample Output:

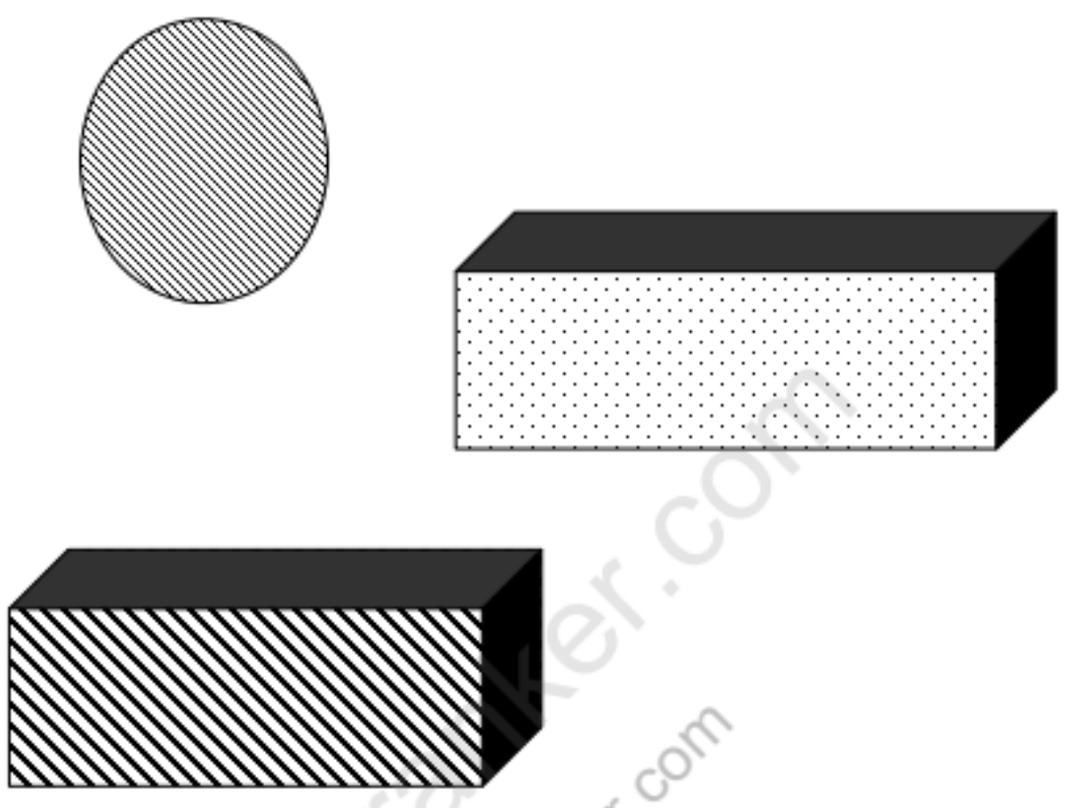


Fig.No 11 3D-Scenes

Result:

Thus a C program for composing a scene using three dimensional objects is executed and verified.

Outcome:

Thus the outcome of implementing 3D scenes has been met.

Application:

- 3D movies

Viva-voce

Animation

1. Define – Computer Graphics
2. What is scan line algorithm?
3. Define – Fractals
4. What is visible surface determination?
5. List the two basic types of hidden line algorithm.
6. Differentiate A-buffer algorithms from Z-buffer?
7. What is the importance of illumination and shading model in crating realistic image?
8. What is diffused reflection?
9. What is halfway vector?
10. Write a short note on warn model.
11. What is refraction effect?
12. Write short note on shadow.
13. Write short note on halftoning.
14. Define – Phong Shading
15. Define – Halftoning
16. State advantages of gouraud shading.
17. State disadvantages of gouraud shading.
18. What is index of refraction?
19. What is meant by angle of refraction?
20. Define – Halfway Vector

Expt. No.12

IMAGE EDITNG AND MANIPULATION

IMPLEMENTATION OF BASIC OPERATIONS ON IMAGE USING PHOTOSHOP

Aim:

To create an image by applying basic operations using Adobe Photoshop

Software requirements:

Adobe Photoshop 1.0

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Open adobe Photoshop 1.0,
2. Select the file menu --> click open --> browse a picture,
3. Select filter menu from the title bar,
4. Select filter option --> blur --> radial blur,
5. Set the appropriate radial to blur the picture and click ok.

Sample Output:

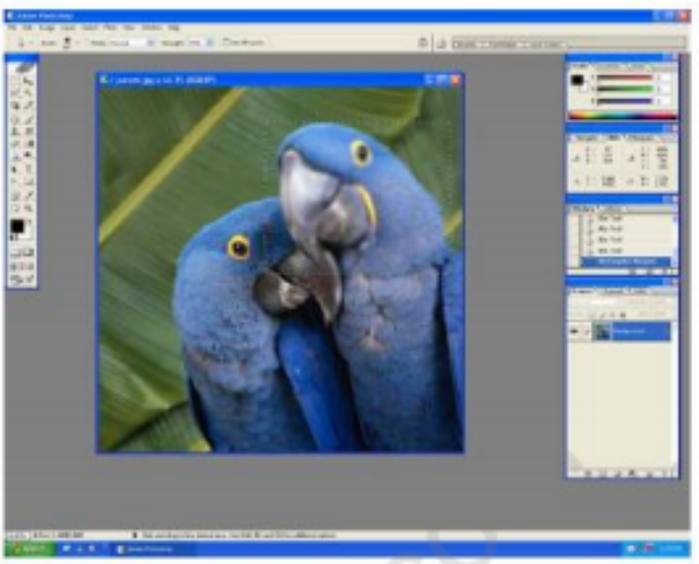


Fig.No 12 Image manipulation

Result:

Thus the program for generating filtered image was created, executed and the output was verified successfully.

Outcome:

Thus the outcome of applying image editing and manipulation techniques has been met.

Application:

- Photo Editing
- Face manipulation

Viva-voce

1. What is adobe photoshop?
2. List the various latest versions of adobe photoshop.
3. Can you explain Adobe Photoshop is raster based software or vector based software? Please explain both?
4. Please explain some important tool in Adobe Photoshop and their features?
5. What is a clipping mask and how we can create a Clipping Mask in Adobe Photoshop?

Expt. No. 13

CREATION OF GIF ANIMATED IMAGES

Aim:

To create a gif animated images using Photoshop

Software requirements:

Adobe Photoshop 1.0

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Create a document. Put each frame of the animation on a different layer. Alternately, open an existing video. From the File menu, choose Import -> Video rames to Layers,
2. Select the layers. Select the layers to be used in the animation from the Layers window. To select a group of layers, select the layer at the top of the group. Then hold the shift key and click on the bottom layer,
3. Open the Animation window. From the Window menu, choose Animation,
4. Click on the "Flyout" menu in the upper right hand corner of the Animation window and select "Convert to Frame Animation",
5. Create frames for each individual layer. Click the "Flyout" menu on the Animation window and choose "Make Frames From Layers",
6. Modify each frame as desired. Select the frame on the Animation window and change it as desired in the main photoshop window. To add or remove a graphic from another layer to any frame, select the frame and in the layers palette. Click the "eye" to toggle the visibility for that layer either on or off,
7. Click on the arrowhead under each frame to display the timing menu. Select the display time for each frame,
8. From the File menu, choose "Save for Web and Devices" and choose GIF from the drop-down menu. To save as a movie, select Export -> Render Video from the File menu to export the document as a movie

Sample Output:

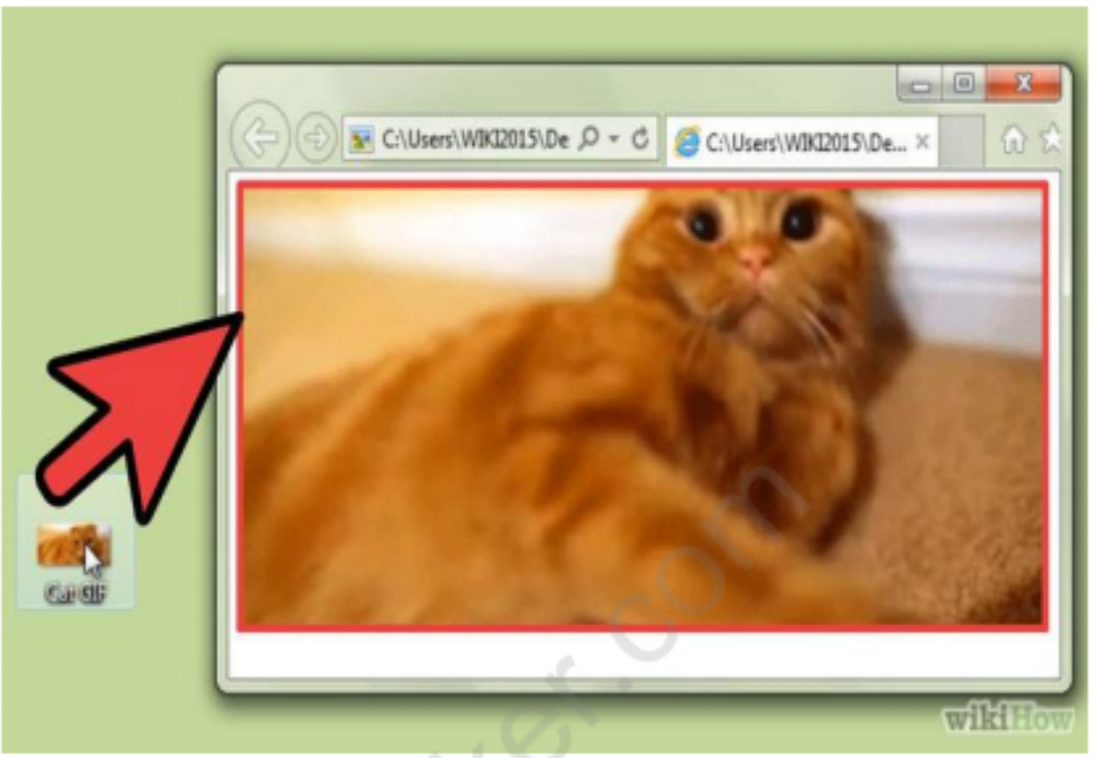


Fig.No 13 GIF animated image

Result:

Thus the gif animated images was created and executed successfully.

Outcome:

Thus the outcome of applying image editing and manipulation techniques has been met.

Application:

- Animation of cartoon images

Viva-voce

1. What is scope and uses of Adobe Photoshop?
2. Explain smart object in PhotoShop?
3. What are swatches palettes?
4. What is histogram in Photoshop?
5. Explain about the Photoshop Work Area?

Expt. No. 14

OPTIMIZING AN IMAGE

Aim:

To optimize an image using Photoshop

Software requirements:

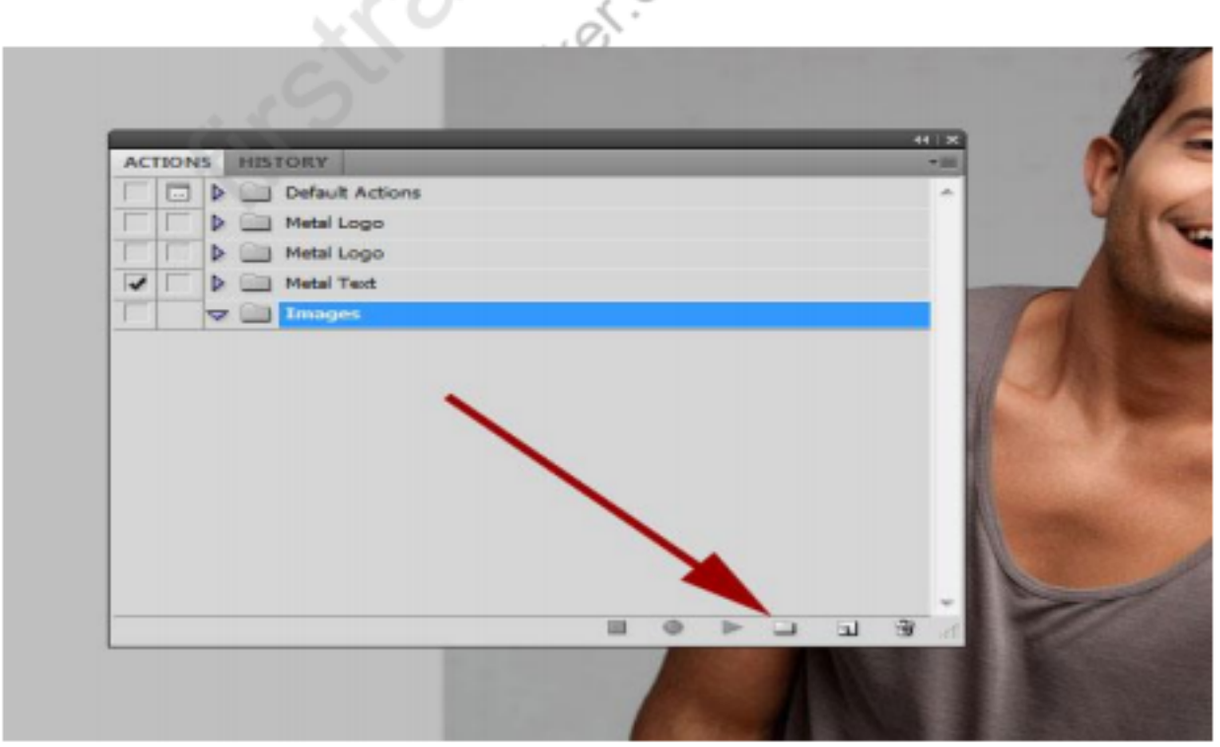
Adobe Photoshop 1.0

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Open Photoshop and Open Your Image Go to File -> Open (**Ctrl/Command+O**) and bring your image to your workspace,
2. Start Photoshop options -> Bring up the Actions Panel up by going to Windows -> Actions (**Alt/Option+F9**) From the Actions Panel, click on Create new set icon: name your new set. For example named as my Images,



3. To start the recording process, click on Create new action icon; give your action a name. For example named as Image Optimization. Click **Record** to start,

4. Save your image for web and devices. Save your image for web and devices by going to File -> Save for Web & Devices (Alt/Option+Shift+Ctrl/Command+S).
5. Follow settings while saving the images:

   Preset: JPEG High (selects JPEG with Quality at 60)

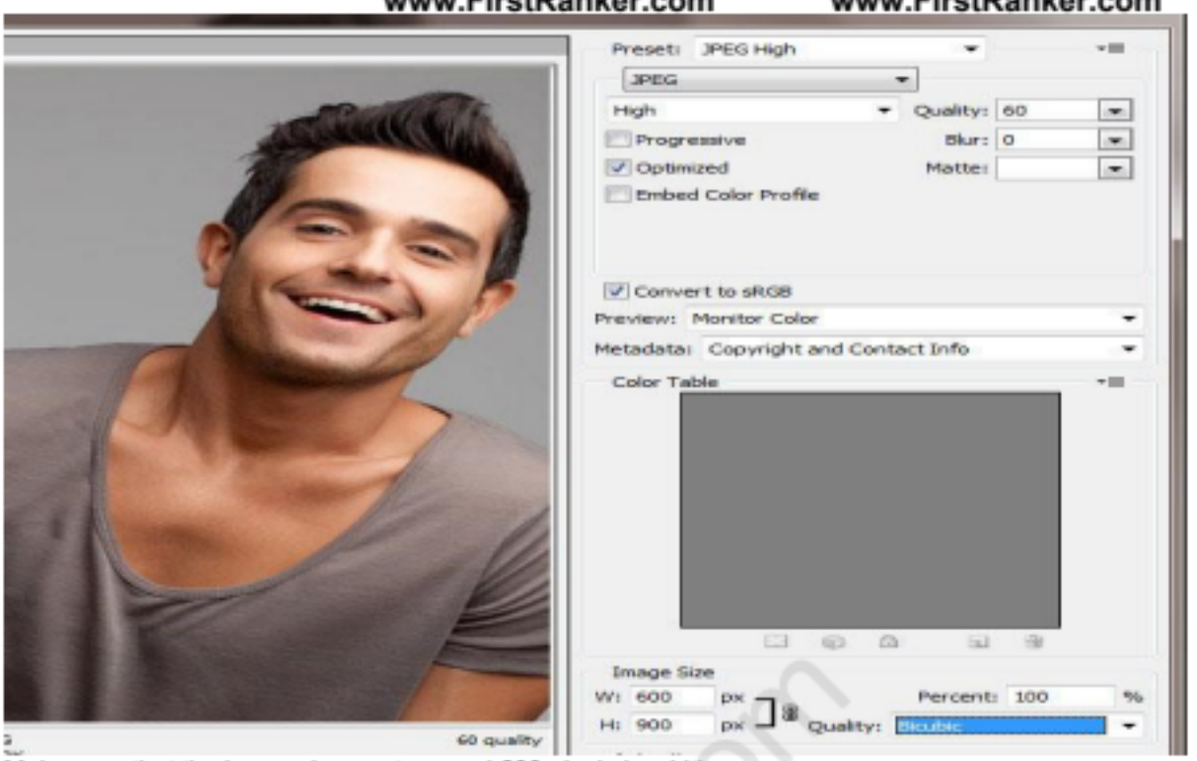
   Optimized: Checked

   Convert to sRGB: Checked

   Size: change according to your site specifications

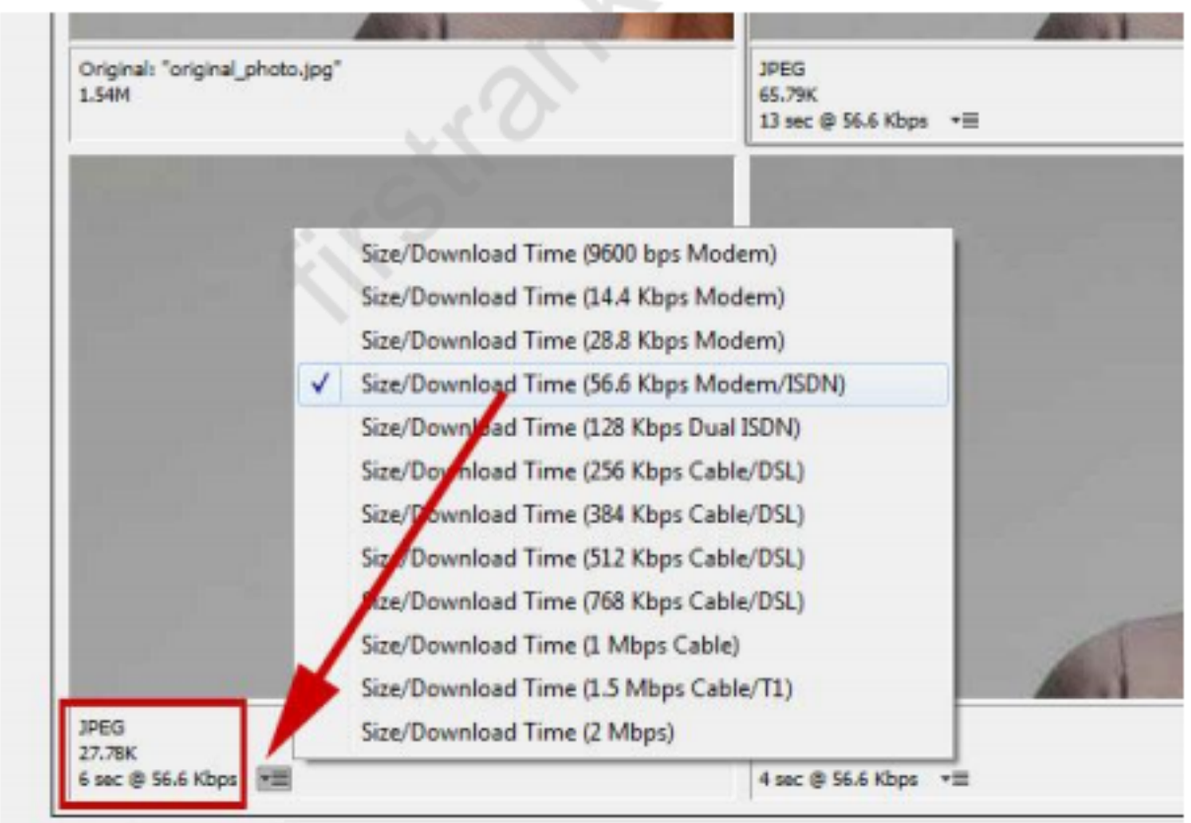
   Percent: 100%

   Quality: Bicubic



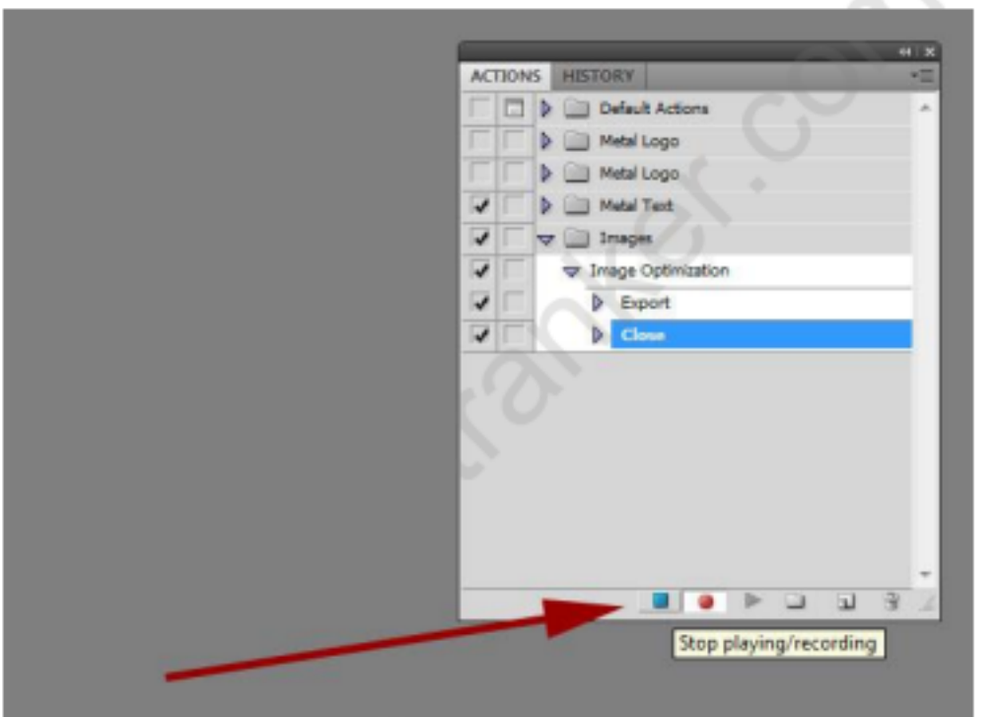
Make sure that the image does not exceed 600 pixels in width

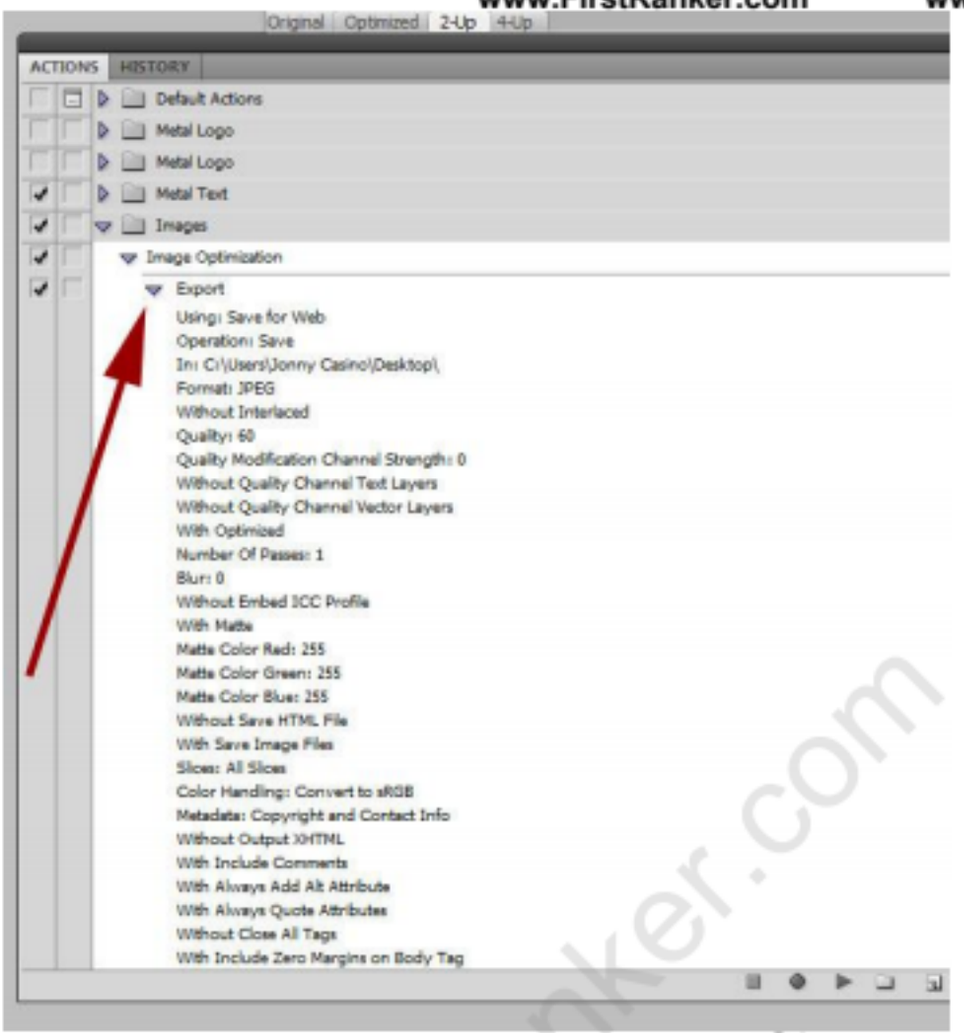
6. Click on the Select download speed icon to change the Size/Download Time to get an estimate on how long it will take to download your image at the selected Size/Download Time



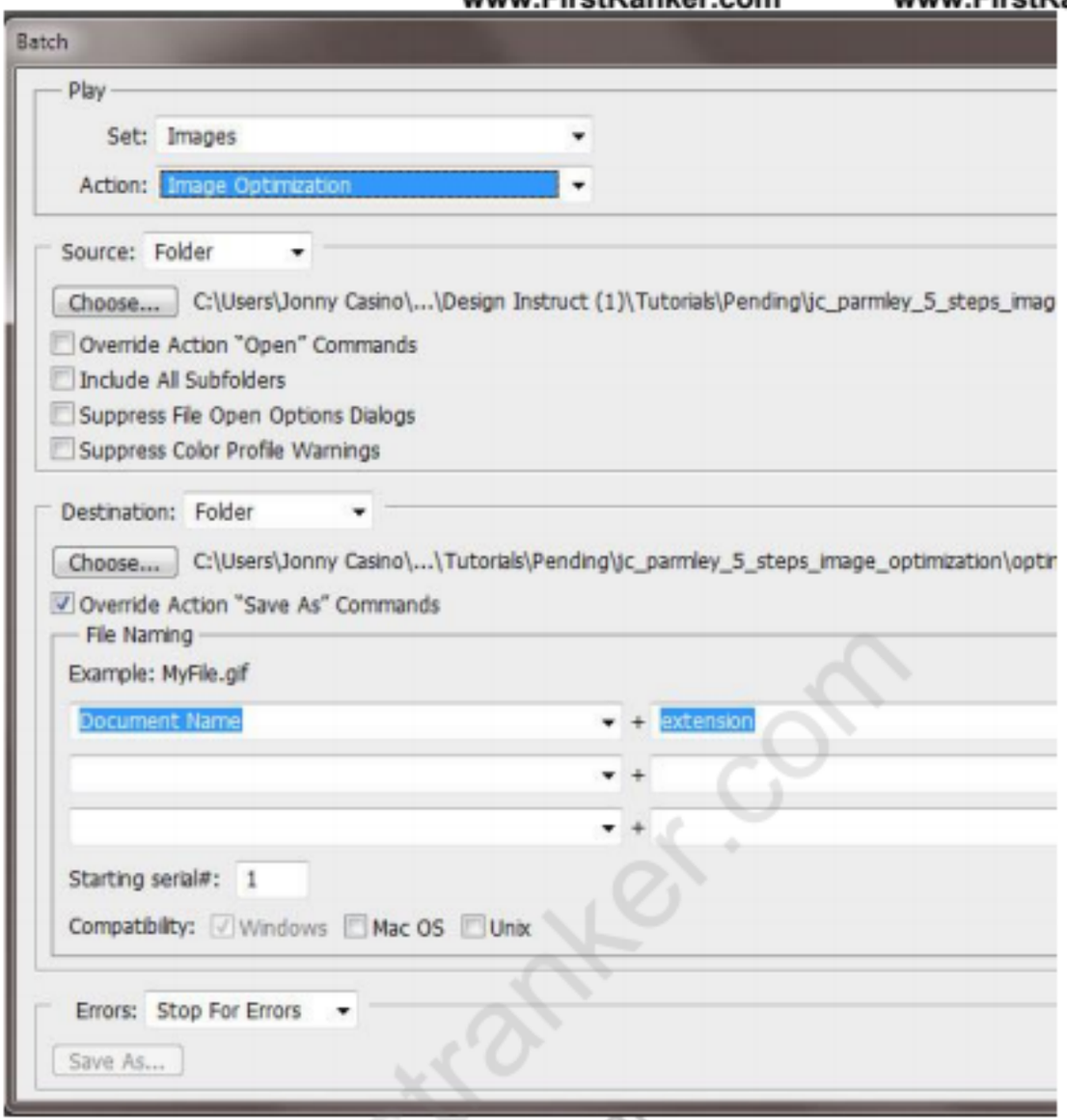
7. To preview the image in real size, click on the Preview button to see your image in the web browser

8. Click Save button,
9. Save the images in a different folder to preserve original images with the larger dimensions for future use,
10. Resize the images down to 600 pixels in width,
11. Close your image file from your workspace,
12. In Actions Panel, click on the Stop playing/recording button to stop recording the actions. This should record everything including closing the image





13. Go to File -> Automate -> Batch. Batch Process your images. Use the settings given below



14. Select the Actions Set and Action recorded,
15. Set Source and Destination Folders, and check Override Action "Save As" Commands,
16. Use Image Optimizing Tools named Smush to further reduce the size of the image,
17. Click on the UPLOADER tab to upload the optimized image.

Sample Output:

Fig.No 14 Optimizing Images

Result:

Thus an image is optimized and uploaded successfully.

Outcome:

Thus the outcome of applying image editing and manipulation techniques has been met.

Application:

- Photos and videos

VIVA - VOCE

1. Define – Compression Efficiency
2. What is Image Processing?
3. Explain Image Calibration?
4. What is Chromaticity?
5. Define – Colour Model
6. What are the uses of chromaticity diagram?
7. Give the transformation matrix for conversion of RGB to YIQ.
8. What is HSV model?
9. What is CMY colour model?
10. What are the parameters in the HLS colour model?
11. What are subtractive colours?
12. What are additive colours?
13. What is colour gamut?
14. Mention the term complementary colours.
15. Define – Primary Colours
16. Draw the colour model HLS double cone.
17. What is meant by hue?
18. What are the complementary colours?
19. What are the primary colours?
20. Define – Colour Gamut Model

Expt.No:15

CREATION OF 2D INTERACTIVE ANIMATION USING FLASH

Aim:

To compose a scene by applying shape tweening using Macromedia Flash

Software requirements:

Macromedia flash 8

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Open a macromedia flash from the start menu
2. From the toolbar select an object and place it on the play area
3. Select the frame, select the old object and delete it and draw the new object with new different color and shape
4. In the frame, right click and select insert key frame, and Select the first object, offset window --> select tween as shape and same for the second object
5. Press ctrl + enter to test the movie

Sample Output:

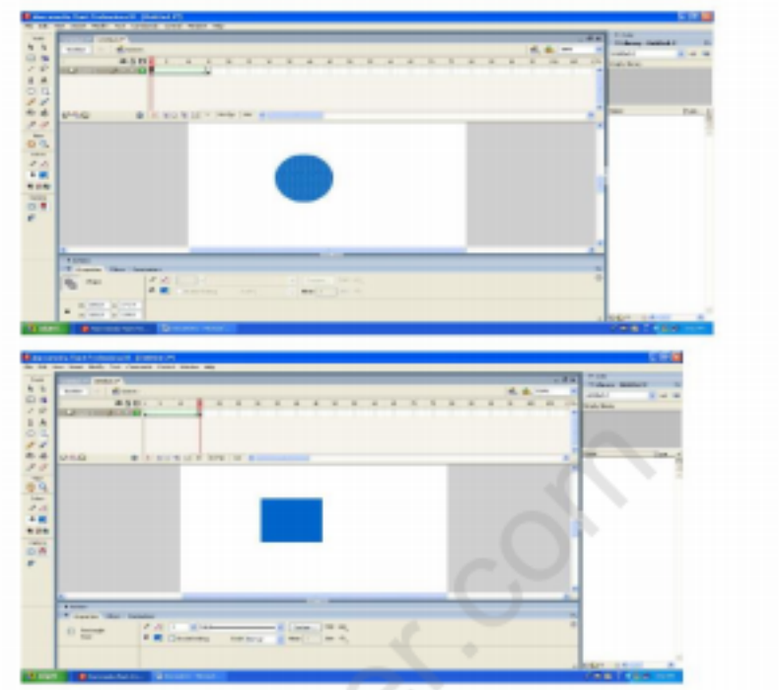


Fig.No 15 2D-Interactive animations

Result:

Thus the program for implementing the shape tweening was created, executed and the output is verified successfully.

Outcome:

Thus the outcome of creating 2d interactive animation has been met.

Application:

- Morphing of images

Viva-voce

1. How to embed Flash in HTML?
2. What is meant by Vector Graphic Animation? What is the use of pre-loader?
3. What is the use of depth? How many methods for depth are available?
4. What is Event Flow?
5. How to start a graphic animation at a specific frame? -
6. What is the difference between _root and parent?
7. Differentiate between AS2 and AS3.
8. How to create scrolling gallery in portfolio ?
9. Describe ChangeWatcher.watch.
10. Give an example that illustrates polymorphism in Flash Script.
11. How to make synchronous data calls in ActionScript?
12. How to add event listeners in MXML components. Name the AS3 components?
13. How to change background color and Stage size? -
14. How to add an effect to the movie clip?
15. How to create a logo using Pen tool?
16. Write a function for a button.
17. How to play a movie clip?
18. How to add conditional logic for the Submit button?
19. Name few open source alternatives of Photoshop.
20. List the features & capabilities of Photoshop

ADDITIONAL EXPERIMENTS BEYOND THE SYLLABUS

Expt.No:16

GENERATE A 2D IMAGE AND ADD MOTION TO IT USING C

Aim:

To write a C program to generate 2D image and add motion to it

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Include the necessary header files,
3. Initialize the graphic detection and graphic mode to bgi folder that supports graphics,
4. Use the line, circle and rectangle function to create a car,
5. Use the for loop up to the value 600 to move the car by setting the delay as 20,
6. Display the results,
7. Close the initialized graphic mode,
8. Stop the program.

Sample Output:

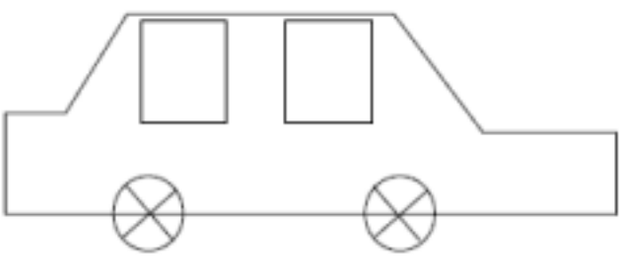


Fig.No 16 2D image with motion

Result:

Thus a C program to generate 2D image was created and motions were added to it.

Outcome:

Thus the outcome of generating 2D interactive image has been met.

Viva-voce

1. What is Image Animation?
2. What is Gray scale normalization?
3. What is Frame Averaging?
4. What are the steps in animation sequence?
5. How frame-by-frame animation works?
6. What is morphing?
7. What are the methods of motion specifications?
8. What is critical fusion frequency?
9. What is tweening?
10. What do you mean by fractals?
11. Define – Fractals
12. Give the classification of fractals.
13. What is topological dimension?
14. What is fractal dimension?
15. Write short note on fractal geometry.
16. Write about random fractals in detail.
17. Define – Hilbert's Curve
18. Write note on tiling a plane.
19. What are the various types of tiling a plane?
20. What is meant by drawing tiling?

Expt.No:17

IMPLEMENTATION OF BRESENHAM'S ELLIPSE DRAWING ALGORITHM

Aim:

To write a program in C to draw an ellipse using midpoint ellipse drawing algorithm

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Get the radius rx, ry and center of the ellipse xc, yc,
3. Obtain the first point on the ellipse centered on the origin as $(x_0, y_0) = (0, ry)$,
4. Calculate the initial value of the decision parameter in region1 as $p1_0 = ry^2 - rx^2ry + 1/4rx^2$.
5. At each xk position in region1,starting at k = 0,perform the following test
   if $(p1_k < 0)$, the next point along the ellipse centered on (0, 0) is (x_{k+1}, y_k) and $p1_{k+1} = p1_k + 2ry^2x_{k+1} + ry^2$
   Otherwise the next point along the ellipse is (x_{k+1}, y_{k-1}) and $p1_{k+1} = p1_k + 2ry^2x_{k+1} + ry^2 - 2rx^2y_{k+1}$, where $2ry^2x_{k+1} = 2ry^2x_k + 2ry^2$ and $2rx^2y_{k+1} = 2rx^2y_k - 2rx^2$.
6. Calculate the value of the decision parameter in region2 using the point (x0, y0) as $p2_0 = ry^2 (x_0 + 1/2)^2 + rx^2 (y_0 - 1)^2 - rx^2ry^2$.
7. At each yk position in region2,starting at k=0,perform the following test if (p2k > 0), the next point along the ellipse centered on (0, 0) is (xk, yk+1) and $p2_{k+1} = p2_k - 2rx^2y_{k+1} + rx^2$ Otherwise the next point along the ellipse is (x_{k+1}, y_{k+1}) and $P2_{k+1} = p2_k + 2ry^2x_{k+1} + rx^2 - 2rx^2y_{k+1}$, using the same incremental calculations for x & y as in region2,
8. Determine the symmetry points in other three quadrants.
9. Move each calculated pixel position (x, y) onto the elliptical path centered on (xc, yc) and plot the coordinate values, x = x + xc & y = y + yc.
10. Repeat the steps for region1 until $2ry^2x > 2rx^2y$,
11. Stop the program.

Sample Output:

Enter the xa value: 200

Enter the ya value: 200

Enter the x radius: 100

Enter the y radius: 50

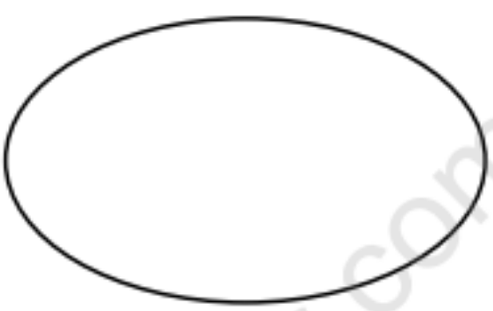


Fig.No 17 Ellipse using Bresenham's

Result:

Thus an ellipse is drawn successfully using midpoint ellipse drawing algorithm in C.

Outcome:

Thus the outcome of implementing ellipse drawing algorithms has been met.

Viva-voce

1. Define – Resolution
2. Use the Cohen Sutherland algorithm to clip line P1 (70,20) and p2(100,10) against a window lower left hand corner (50,10) and upper right hand corner (80,40)
3. Prove that two 2D rotation above the origin commutative i.e R1R2=R2R1.
4. What is meant by curve clipping?
5. Find out the aspect ratio of the raster system using 8 x 10 inches screen and 100 pixel/inch
6. How Many k bytes does a frame buffer nees in a 600 x 400 pixel?

Expt.No:18

IMPLEMENTATION OF LINE, CIRCLE AND ELLIPSE ATTRIBUTES

Aim:

To write a C Program to display the various output primitives with its attributes

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

Algorithm:

1. Start the program.
2. Initialize the variables,
3. Call the initgraph() function,
4. Set color for the output primitives,
5. Using outtextxy() display the chosen particular primitives,
6. Using switch cases mention the various primitives and their attributes,
7. The various primitives are arc, line, circle, rectangle and ellipse,
8. Close the graph and run the program,
9. Stop the program.

Sample Output:

```
Enter Choice: y
Enter 1 to get line
2. Circle
3. Box
4. Arc
5. Ellipse
6. Rectangle
7. Exit
```

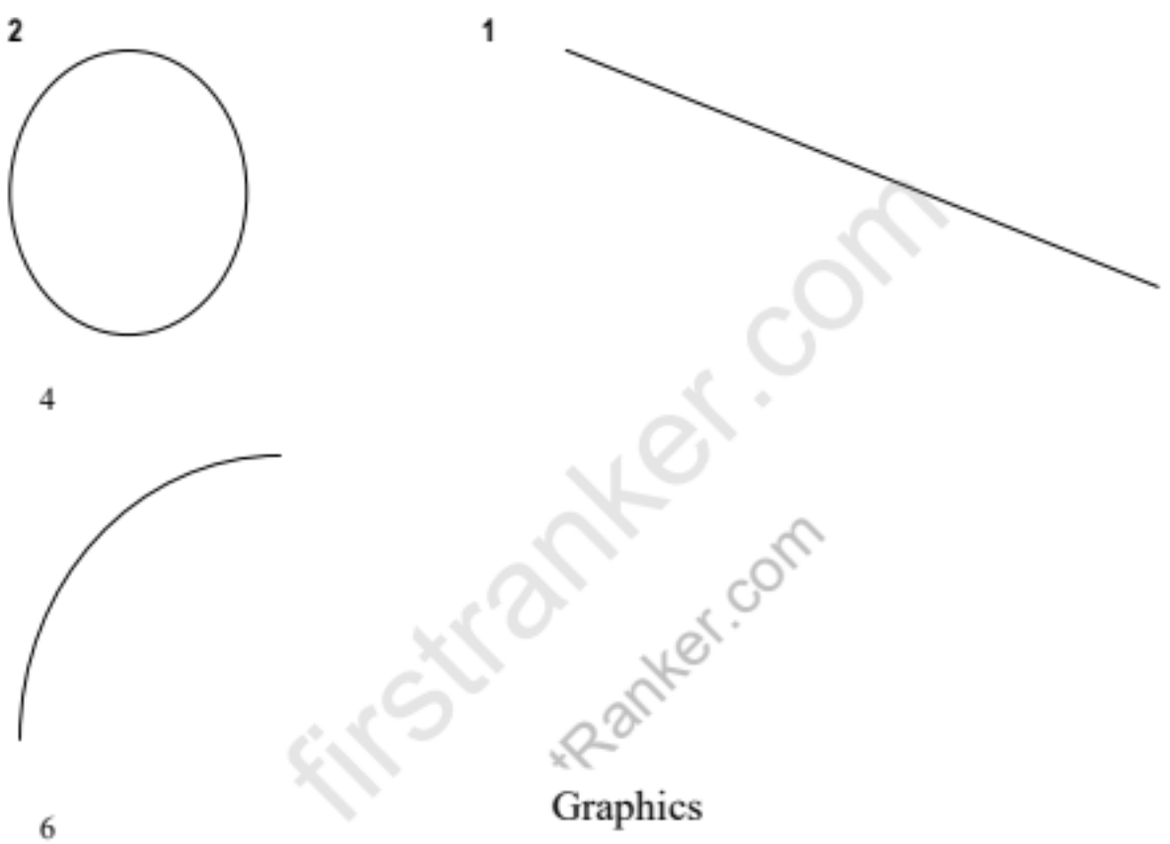


Fig.No 18 Graphics manipulations

Result:

Thus the program for implementing line, circle and ellipse attributes was executed successfully.

Outcome:

Thus the outcome of implementing line, circle, ellipse attributes has been met.

Viva-voce

1. How to generate the circle in second quadrant from first clockwise?
2. How to generate the circle in second quadrant from first anti clockwise?
3. How to generate the circle in third quadrant from first clockwise?
4. How to generate the circle in third quadrant from first anti clockwise?
5. Explain the Bresenham's circle algorithm.
6. Explain the different moves in this algorithm
7. When to choose vertical move?
8. When to choose horizontal move?
9. When to choose diagonal move?
10. Generate the circle in 1st quadrant having radius 8 and center is origin.
11. What is scan conversion?
12. How will you generate a polygon?
13. Which algorithms can be used to generate a polygon?
14. What is the equation of origin centered axis
15. How will you generate an ellipse?
16. What is the difference between circle generation and ellipse generation algorithm
17. Explain the different moves in ellipse algorithm.
18. When to choose vertical move in ellipse algorithm?
19. When to choose horizontal move in ellipse algorithm?
20. When to choose diagonal move in ellipse algorithm?

Expt.No:19

IMPLEMENTATION OF SUTHERLAND HODGEMAN POLYGON CLIPPING ALGORITHM

Aim:

To write a C program to implement Sutherland Hodgeman polygon clipping algorithm

Software requirements:

C, C++ compilers, Java, OpenGL

Hardware requirements:

Dual core processor, DDR2 1GB RAM, 250 GB HDD

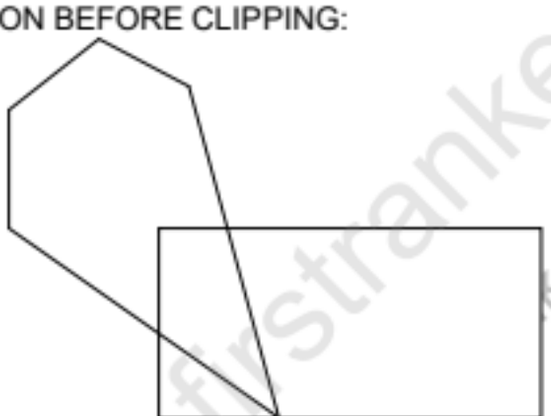
Algorithm:

1. Start the program .
2. Declare the variables for defining clip window & polygon,
3. While clipping a polygon following four possible cases to be considered
   - If the first vertex is outside the window boundary and the second vertex inside
   - If the first vertex is inside the window boundary and the second vertex outside
   - If both are outside
   - If both are inside,
4. Clip the polygon against the window boundary in the order left, right, bottom & above using intersection Calculation,
5. Check the vertices that formed the polygon lies inside or on the boundary. If that is inside or on the boundary save that point otherwise discard it,
6. Stop the program.

Sample Output:

```
Enter the no of sides of polygon:5
Enter the coordinates of polygon
50
50
200
100
350
350
80
200
40
80
Enter the rectangular coordinates of clipping window
150
150
300
300
```

POLYGON BEFORE CLIPPING:



POLYGON AFTER CLIPPING:

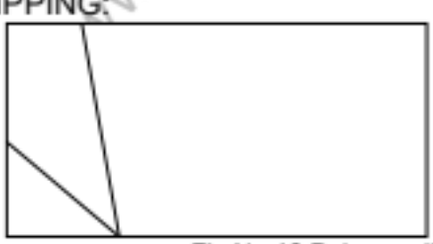


Fig.No 19 Polygon clippings

Result:

Thus the C program to implement Sutherland Hodgeman polygon clipping was executed and verified.

Outcome:

Thus the outcome of implementing Sutherland Hodgeman polygon clipping has been met.

Viva-voce

1. What is Cohen Sutherland line clipping?
2. What do you understand by clipping?
3. What is Z-buffer algorithm for removing hidden faces?
4. What are orthographic projections? When do we need them?
5. What is an aliasing? Explain different methods of minimizing its effect?
6. What is polygon clipping?
7. What is windowing and clipping?
8. List the advantages of interactive Graphics.
9. What do you mean by composite transformation? How it is useful?
10. What is concatenation?
11. What are the advantages of quaternion?
12. What is projection normalization?
13. Explain with the help of opengl functions perspective and parallel viewing opengl?
14. What is gluLookAt() function?

PROJECTS

1. Rasterize two simple graphical primitives
2. Write a ray tracer
3. 2D to 3D conversions
4. Create OpenGL game
5. Image processing and manipulation
6. Animating an image and video using flash