

Marathwada Mitra Mandal's
Institute of Technology
Lohgaon, Pune
Accredited with 'A' Grade by NAAC



Department of Computer Engineering

Computer Graphics Lab

210247

Prepared by,

Prof. Uma B. Karanje

Prof. Chaitanya S.Bhosale

SE COMP

Semester I Academic Year 2022-23

Programme Outcomes: As prescribed by NBA

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- 3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and cultural, societal, and environmental considerations.
- 4. Conduct investigations of complex problems:** The problems that cannot be solved by straightforward application of knowledge, theories and techniques applicable to the engineering discipline.
- 5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modeling to complex engineering activities, with an understanding of the limitations.
- 6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
- 7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
- 8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- 9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- 10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with the society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Course Objectives:

1. To acquaint the learner with the basic concepts of Computer Graphics
2. To learn the various algorithms for generating and rendering graphical figures
3. To get familiar with mathematics behind the graphical transformations
4. To understand and apply various methods and techniques regarding projections, animation, shading, illumination and lighting

Course Outcomes:

On completion of the course, student will be able to—

1. Apply mathematics and logic to develop Computer programs for elementary graphics operations like line and circle.
2. Develop scientific and strategic approach to solve complex problems in the domain of Computer Graphics such as polygon filling, windowing and clipping.
3. Compare different transformations and differentiate between 2-D and 3-D transformation.
4. Apply the logic to develop animation and gaming programs.
5. Understand the concepts of illumination models, shading algorithms and hidden surfaces.
6. Explain advanced tools and technologies for enhancement of graphics applications

Savitribai Phule Pune University														
Second Year of Computer Engineering (2019 Course)														
(With effect from Academic Year 2020-21)														
Semester-III														
Course Code	Course Name	Teaching Scheme (Hours/Week)			Examination Scheme and Marks						Credit Scheme			
		Lecture	Practical	Tutorial	Mid-Sem	End-Sem	Term work	Practical	Oral	Total	Lecture	Practical	Tutorial	Total
210241	Discrete Mathematics	03	-	-	30	70	-	-	-	100	03	-	-	03
210242	Fundamentals of Data Structures	03	-	-	30	70	-	-	-	100	03	-	-	03
210243	Object Oriented Programming (OOP)	03	-	-	30	70	-	-	-	100	03	-	-	03
210244	Computer Graphics	03	-	-	30	70	-	-	-	100	03	-	-	03
210245	Digital Electronics and Logic Design	03	-	-	30	70	-	-	-	100	03	-	-	03
210246	Data Structures Laboratory	-	04	-	-	-	25	50	-	75	-	02	-	02
210247	OOP and Computer Graphics Laboratory	-	04	-	-	-	25	25	-	50	-	02	-	02
210248	Digital Electronics Laboratory	-	02	-	-	-	25	-	-	25	-	01	-	01
210249	Business Communication Skills	-	02	-	-	-	25	-	-	25	-	01	-	01
210250	Humanity and Social Science	-	-	01	-	-	25	-	-	25	-	-	01	01
210251	Audit Course 3													
Total Credit											15	06	01	22
Total		15	12	01	150	350	125	75	-	700	-	-	-	-

Savitribai Phule Pune University Second Year of Computer Engineering (2019 Course) 210247: OOP and Computer Graphics Laboratory		
Teaching Scheme Practical: 04 Hours/Week	Credit Scheme 02	Examination Scheme and Marks Term Work: 25 Marks Practical: 25Marks
Companion Course : 210243: Object Oriented Programming(OOP), 210244: Computer Graphics		
Course Objectives: To understand basics of Computer Graphics, apply various methods and techniques for implementing line-circle drawing, projections, animation, shading, illumination and lighting using concepts of Object Oriented Programming.		
Course Outcomes: On completion of the course, learner will be able to– CO1: Understand and apply the concepts like inheritance, polymorphism, exception handling and generic structures for implementing reusable programming codes. CO2: Analyze the concept of file and apply it while storing and retrieving the data from secondary storages. CO3: Analyze and apply computer graphics algorithms for line-circle drawing, scan conversion and filling with the help of object oriented programming concepts. CO4: Understand the concept of windowing and clipping and apply various algorithms to fill and clip polygons. CO5: Apply logic to implement, curves, fractals, animation and gaming programs.		
Guidelines for Instructor's Manual The instructor's manual is to be developed as a reference and hands-on resource. It should include prologue (about University/program/ institute/ department/foreword/ preface), curriculum of the course, conduction and Assessment guidelines, topics under consideration, concept, objectives, outcomes, set of typical applications/assignments/ guidelines, and references.		
Guidelines for Student's Laboratory Journal The laboratory assignments are to be submitted by student in the form of journal. Journal consists of Certificate, table of contents, and handwritten write-up of each assignment (Title, Date of Completion, Objectives, Problem Statement, Software and Hardware requirements, Assessment grade/marks and assessor's sign, Theory- Concept in brief, algorithm, flowchart, test cases, Test Data Set(if applicable), mathematical model (if applicable), conclusion/analysis. Program codes with sample output of all performed assignments are to be submitted as softcopy. As a conscious effort and little contribution towards Green IT and environment awareness, attaching printed papers as part of write-ups and program listing to journal must be avoided. Use of DVD containing students programs maintained by Laboratory In-charge is highly encouraged. For reference one or two journals may be maintained with program prints in the Laboratory.		
Guidelines for Laboratory /Term Work Assessment Continuous assessment of laboratory work should be based on overall performance of Laboratory assignments by a student. Each Laboratory assignment assessment will assign grade/marks based on parameters, such as timely completion, performance, innovation, efficient codes, punctuality and		
Guidelines for Practical Examination Problem statements must be decided jointly by the internal examiner and external examiner. During practical assessment, maximum weightage should be given to satisfactory implementation of the problem statement. Relevant questions may be asked at the time of evaluation to test the student's understanding of the fundamentals, effective and efficient implementation. This will encourage, transparent evaluation and fair approach, and hence will not create any uncertainty or doubt in the minds of the students. So adhering to these principles will consummate our team efforts to the promising start of student's academics.		

Guidelines for Laboratory Conduction

The instructor is expected to frame the assignments by understanding the prerequisites, technological aspects, utility and recent trends related to the topic. The assignment framing policy need to address the average students and inclusive of an element to attract and promote the intelligent students. Use of open source software is encouraged. Based on the concepts learned. Instructor may also set one assignment or mini-project that is suitable to respective branch beyond the scope of syllabus.

Operating System recommended :- 64-bit Open source Linux or its derivative

Programming tools recommended: - Open Source C++ Programming tool like G++/GCC, OPENGL.


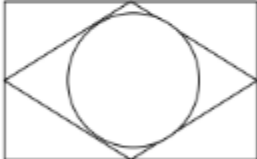
Virtual Laboratory:

- <http://cse18-iiith.vlabs.ac.in/Introduction.html?domain=Computer%20Science>
- <http://vlabs.iitb.ac.in/vlabs-dev/labs/cglab/index.php>

Part II : Computer Graphics

Suggested List of Laboratory Experiments/Assignments

(All assignments are compulsory)

Sr. No.	Group A
1.	Write C++ program to draw a concave polygon and fill it with desired color using scan fill algorithm. Apply the concept of inheritance.
2.	Write C++ program to implement Cohen Sutherland line clipping algorithm.
3.	<p>a) Write C++ program to draw the following pattern. Use DDA line and Bresenham's circle drawing algorithm. Apply the concept of encapsulation.</p>  <p>OR</p> <p>b) Write C++ program to draw the following pattern. Use DDA line and Bresenham's circle drawing algorithm. Apply the concept of encapsulation.</p> 
Group B	
4.	<p>a) Write C++ program to draw 2-D object and perform following basic transformations, Scaling b) Translation c) Rotation. Apply the concept of operator overloading.</p> <p>OR</p> <p>b) Write C++ program to implement translation, rotation and scaling transformations on equilateral triangle and rhombus. Apply the concept of operator overloading.</p>
5.	<p>a) Write C++ program to generate snowflake using concept of fractals.</p> <p>OR</p> <p>b) Write C++ program to generate Hilbert curve using concept of fractals.</p> <p>OR</p> <p>c) Write C++ program to generate fractal patterns by using Koch curves.</p>
Group C	
6.	<p>a) Design and simulate any data structure like stack or queue visualization using graphics. Simulation should include all operations performed on designed data structure. Implement the same using OpenGL.</p> <p>OR</p> <p>b) Write C++ program to draw 3-D cube and perform following transformations on it using OpenGL i) Scaling ii) Translation iii) Rotation about an axis (X/Y/Z).</p> <p>OR</p> <p>c) Write OpenGL program to draw Sun Rise and Sunset.</p>

7. a) Write a C++ program to control a ball using arrow keys. Apply the concept of polymorphism.
OR
 b) Write a C++ program to implement bouncing ball using sine wave form. Apply the concept of polymorphism.
OR
 c) Write C++ program to draw man walking in the rain with an umbrella. Apply the concept of polymorphism.
OR
 Write a C++ program to implement the game of 8 puzzle. Apply the concept of polymorphism.
OR
 d) Write a C++ program to implement the game Tic Tac Toe. Apply the concept of polymorphism.

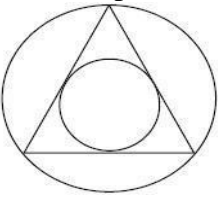
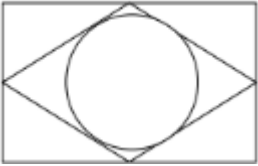
Mini-Projects/ Case Study

8. Design and implement game / animation clip / Graphics Editor using open source graphics library. Make use of maximum features of Object Oriented Programming.

[@The CO-PO Mapping Matrix](#)

PO/CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	-	1	2	1	-	-	-	-	-	-	-	-
CO2	-	1	2	1	-	-	-	-	-	-	-	-
CO3	2	1	1	-	-	-	-	-	-	-	-	-
CO4	1	2	2	1	-	-	-	-	-	-	-	-
CO5	-	2	2	1	-	-	-	-	-	-	-	-

INDEX

Sr. No.	Asg No	Title of Assignment	CO	PO
Prerequisites	0	Study of basic graphics primitives.		
1.	1.	Write C++ program to draw a concave polygon and fill it with desired color using scan fill algorithm. Apply the concept of inheritance.		
2.	2.	Write C++ program to implement Cohen Southerland line clipping algorithm.		
3.	3A.	a) Write C++ program to draw the following pattern. Use DDA line and Bresenham's circle drawing algorithm. Apply the concept of encapsulation. 		
	3B.	b) Write C++ program to draw the following pattern. Use DDA line and Bresenham's circle drawing algorithm. Apply the concept of encapsulation 		
4	4A	a) Write C++ program to draw 2-D object and perform following basic transformations, Scaling b) Translation c) Rotation. Apply the concept of operator overloading. OR		
	4B	b) Write C++ program to implement translation, rotation and scaling transformations on equilateral triangle and rhombus. Apply the concept of operator overloading.		
5	5A	a) Write C++ program to generate snowflake using concept of fractals. OR		

	5B	b) Write C++ program to generate Hilbert curve using concept of fractals. OR		
	5C	c) Write C++ program to generate fractal patterns by using Koch curves.		
6	6A	a) Design and simulate any data structure like stack or queue visualization using graphics. Simulation should include all operations performed on designed data structure. Implement the same using OpenGL. OR		
	6B	b) Write C++ program to draw 3-D cube and perform following transformations on it using OpenGL i) Scaling ii) Translation iii) Rotation about an axis (X/Y/Z). OR		
	6C	c) Write OpenGL program to draw Sun Rise and Sunset.		
7	7A	a) Write a C++ program to control a ball using arrow keys. Apply the concept of polymorphism. OR		
	7B	b) Write a C++ program to implement bouncing ball using sine wave form. Apply the concept of polymorphism. OR		
	7C	c) Write C++ program to draw man walking in the rain with an umbrella. Apply the concept of polymorphism. OR		
	7D	d) Write a C++ program to implement the game of 8 puzzle. Apply the concept of polymorphism. OR		
	7E	e) Write a C++ program to implement the game Tic Tac Toe. Apply the concept of polymorphism.		
8.		Design and implement game / animation clip / Graphics Editor using open source graphics library. Make use of maximum features of Object Oriented Programming.		

Use Object oriented programming concepts to implement the problem statements. Implement programs by using C++ /Java/OpenGL programming language.

Software Required:

1. 64 bit open source operating system
2. Geany, Qt creator
3. Eclipse version 3.8

Write-ups must include:

- **Group:**
- **Assignment No.**
- **Title**
- **Problem Statement**
- **Prerequisites**
- **Course Objectives**
- **Course Outcomes**
- **Theory(in brief)**
- **Algorithm**
- **Test Cases**
- **Conclusion**
- **FAQs:**
- **Output: Printout of program with output.**

PREREQUISITES ASSIGNMENT NO: 1

AIM: To learn Basic Graphics primitives.

PROBLEM STATEMENT: Study of basic graphics primitives.

PREREQUISITES:

1. Understanding of basics of computer graphics and data structure.

COURSE OBJECTIVE:

1. To Understand basics of Computer Graphics

THEORY:

- **Text Editor:**

A **text editor** is a type of program used for editing plain text files.

Text editors are often provided with operating systems or software development packages, and can be used to change configuration files and programming language source code.

- **Graphics Editor :**

A **raster graphics editor** is a computer program that allows users to paint and edit pictures interactively on the computer screen and save them in one of many popular "bitmap" or "raster" formats such as JPEG, PNG, GIF and TIFF. Usually an image viewer is preferred over a raster graphics editor for viewing images. Some editors specialize in the editing of photo-realistic images such as the popular Adobe Photoshop, while others are more geared to artist-created illustrations.

Graphics mode Initialization :

First of all we have to call the `initgraph` function that will initialize the graphics mode on the computer. `Initgraph` have the following prototype.

`void initgraph(int far *graphdriver, int far *graphmode, char far *pathdriver);`

`Initgraph` initializes the graphics system by loading a graphics driver from disk (or validating a registered driver) then putting the system into graphics mode. `Initgraph` also resets all graphics

settings(color, palette current position, viewport)to their defaults, then resets graphresult to 0.

***graphdriver** Integer that specifies the graphics driver to be used. You can give graphdriver a value using a constant of the graphics_drivers enumeration type.

***graphmode** Integer that specifies the initial graphics mode(unless *graphdriver = DETECT). If *graphdriver = DETECT, initgraph sets *graphmode to the highest resolution available for the detected driver. You can give *graphmode a value using a constant of the graphics_modes enumeration type.

***pathtodriver** Specifies the directory path where initgraph looks for graphics drivers (*.BGI) first.

1. If they're not there, initgraph looks in the current directory.
2. If pathtodriver is null, the driver files must be in the current directory.

*graphdriver and *graphmode must be set to valid graphics_drivers and graphics_mode values or you'll get unpredictable results. (The exception is graphdriver = DETECT.)

Graphics Primitives:

Function	Description
initgraph	It initializes the graphics system by loading the passed graphics driver then changing the system into graphics mode. <code>int gd,gm; gd=DETECT; initgraph(&gd,&gm,NULL);</code>
getmaxx getmaxy	It returns the maximum X & Y coordinate in current graphics mode and driver. Syntax: <code>int getmaxx() int getmaxy()</code>
setcolor	It changes the current drawing colour. Default colour is white. Each color is assigned a number, like BLACK is 0 and RED is 4. Here we are using colour constants defined inside graphics.h header file. <code>setcolor(int color)</code>
getpixel	It is used to get intensity of current pixel <code>int getpixel(int x, int y)</code>
setfillstyle	It sets the current fill pattern and fill color.

Function	Description
	<div><div><div><div><div><div></div><div></div></div></div><div><div><div></div><div></div></div><div><div></div><div></div></div></div></div><div><div><div></div><div></div></div><div><div></div><div></div></div></div></div><div><div><div></div><div></div></div><div><div></div><div></div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div><div></div><div></div></div><div><div></div><div></div></div></div>

Function	Description
outtextxy	It displays a string at a particular point (x,y) on screen <i>outtextxy(int x, int y, "msg")</i>
floodfill	It is used to fill a closed area with current fill pattern and fill color. It takes any point inside closed area and color of the boundary as input. <i>floodfill(int x, int y, int color)</i>
cleardevice	It clears the screen, and sets current position to (0, 0). <i>cleardevice();</i>
kbhit	It is used to determine whether a key is pressed or not. It returns a non-zero value if a key is pressed otherwise zero. <i>Kbhit()</i>
delay	It is used to suspend execution of a program for a M milliseconds. <i>delay(milliseconds)</i>
closegraph	It unloads the graphics drivers and sets the screen back to text mode. <i>closegraph();</i>

Basic template for writing Graphics program

Class sample

```

{
Public:
//Function
Void accept();
Void display();
};
int main()
{
sample s;
int gd,gm;
gd=DETECT;
-----Text mode -----
s.accept();
-----Initialize Graphics mode-----

```

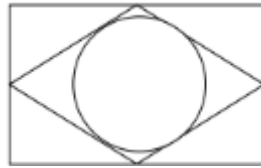
```
initgraph(&gd,&gm,NULL);  
//all graphics related primitives  
//Line.....  
//Cicle.....  
s.display();  
getch(); // display output  
closegraph();// exit from graphics mode  
return 0;  
}
```

CONCLUSION: Basic graphics primitives are learnt and implemented.

GROUP: A ASSIGNMENT NO: 1

AIM: To learn different algorithms for line & circle drawing on raster screen.

PROBLEM STATEMENT A. Write C++ program to draw the following pattern. Use DDALine and Bresenham's circle drawing algorithm. Apply the concept of encapsulation.



PREREQUISITES:

1. Knowledge of Object Oriented Programming
2. Basic mathematics, geometry, linear algebra

COURSE OBJECTIVE:

1. To acquaint the learner with the basic concepts of Computer Graphics
2. To learn the various algorithms for generating and rendering graphical figures

COURSE OUTCOME:

1. Apply mathematics and logic to develop Computer programs for elementary graphic operations

THEORY:

Line is denoted by two points $P1(x1, y1)$ and $P2(x2, y2)$. Mathematical equation for line is $y=mx+b$ where m is slope and b is intercept.

DDA's Line Generation algorithm

In computer graphics, a digital differential analyzer (DDA) is hardware or software used for linear interpolation of variables over an interval between start and end point. DDAs are used for rasterization of lines, triangles and polygons.

Algorithm:

- 1 Input the two line endpoints and store the left endpoint
- 2 Calculate $\Delta x = x2 - x1$ and $\Delta y = y2 - y1$


```

3  If  $\Delta x \geq \Delta y$  then length =  $\Delta x$  else length =  $\Delta y$ 
4   $\Delta x = (x_2 - x_1) / \text{length}$  ;  $\Delta y = (y_2 - y_1) / \text{length}$ 
5   $x = x_1 + 0.5 * \text{Sign}(\Delta x)$ 
6   $y = y_1 + 0.5 * \text{Sign}(\Delta y)$ 
7  plot pixel(x,y)
8  while(i < length)
     $x = x + \Delta x$ 
     $y = y + \Delta y$ 
    plot(x,y)
     $i = i + 1$ 
end while
9  Stop

```

Advantages of DDA Line Drawing algorithm

1. It is the simplest algorithm and it does not require special skills for implementation.
2. It is a faster method for calculating pixel positions than the direct use of equation $y = mx + b$. It eliminates the multiplication in the equation by making use of raster characteristics, so that appropriate increments are applied in the x or y direction to find the pixel positions along the line path.

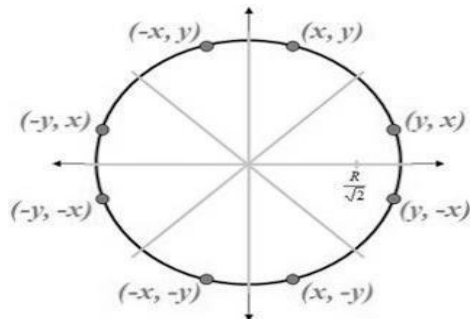
Disadvantages of DDA Algorithm

1. Floating point arithmetic in DDA algorithm is still time-consuming.
2. The algorithm is orientation dependent. Hence end point accuracy is poor.

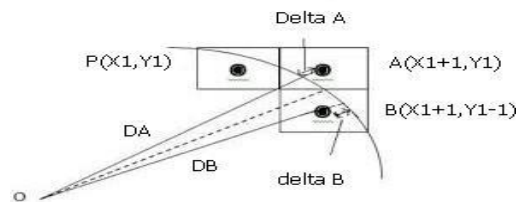
Circle

We can reduce our calculation drastically (8th fraction) by making use of the fact that a circle has 8 way symmetry. Thus after calculating a pixel position (x,y) to be plotted, we get 7 other points on the circle corresponding to it. These are:

$(x,y);(x,-y);(-x,y);(-x,-y);(y,x);(y,-x);(-y,x);(-y,-x);$



Bresenham's circle algorithm.



Bresenham's Circle Drawing algorithm

Algorithm :

1. Accept radius and center co-ordinates from user and plot first point on circumference of circle $(x,y) = (0,r)$
2. Calculate the initial value of decision parameter $d=3-2r$
3. If we are using octant symmetry property to plot the pixel then until $(x < y)$ we have to perform following steps

If $(d \leq 0)$

Update d by $d=d+4x+6$ and increase x by 1

Else

Update d by $d = d + 4(x - y) + 10$ and increase x by 1
and decrease y by 1

4. Determine the symmetry points in other octants also

5. Move and calculate the pixel position (x,y) on to the circular path .

OOP's concepts used

- 1. Class**
- 2. Function**
- 3. Encapsulation**

Test Cases

T_ID	T_Name	Condition to test	Expected outcome	Pass/Fail
T_P1	End point Checking	Given end points are within X max and Y max of screen	Output will be displayed on the screen	
		Given end points are beyond X max and Y max of screen	Output will not be displayed	
T_C1	Center testing	input center is origin	Complete circle will not be displayed	
		Center is shifted to mid point	Complete circle will be displayed	

CONCLUSION: Hence we have studied and implemented DDA line drawing and Bresenham's circle drawing algorithms.

FAQ's

- 1. Compare DDA and Bresenham's Line Drawing**

Algorithms Ans :

	DDA	Bresenham's
Arithmetic	DDA algorithm uses floating points i.e. Real Arithmetic .	Bresenham's algorithm uses fixed points i.e. Integer Arithmetic .
Operations	DDA algorithm uses multiplication and division in its operations.	Bresenham's algorithm uses only subtraction and addition in its operations.
Round Off	DDA algorithm round off the coordinates to integer that is nearest to the line.	Bresenham's algorithm does not round off but takes the incremental value in its operation.
Expensive	DDA algorithm uses an enormous number of floating-point multiplications so it is expensive.	Bresenham's algorithm is less expensive than DDA algorithm as it uses only addition and subtraction.

2. What is Aliasing?

Ans: Stair case or jagged line effect is called aliasing. Minimizing aliasing effect is called antialiasing. Super sampling and area sampling are techniques for antialiasing.

3. Use of Floor and ceil function in DDA algorithm

Ans: The C library function **double floor(double x)** returns the largest integer value less than or equal to **x**.

The C library function **double ceil(double x)** returns the smallest integer value greater than or equal to **x**.

4. What is Antialiasing?

Ans: In computer graphics, antialiasing is a technique for minimizing jaggies- stairstep-like. Aliasing is staircase or jagged line effect.

5. What is 8 way Symmetry?

Ans: Circle can be drawn by drawing $1/8^{\text{th}}$ part. Eight pixels can be drawn by one pixel only.

6. How symmetry will achieved by reflection ?

Ans: Reflection across $y=x$ and $y= -x$ axis gives 8 way symmetry.

7. What is error factor?

Ans: It is a value used to decide more appropriate pixel for rasterization

8. What is error factor d in Bresenham's and Midpoint Circle drawing Algorithm?

Ans: Brsenham's error factor

$d=3-2*r$

if $(d \leq 0)$ then $d=d+4*x+6$

else $d=d+4(x-y)+10$

Midpoint error factor

$d= 1.25-r$

if $(d \leq 0)$ then $d=d+2*x+1$

else $d=d+2(x-y)+1$

9. What is trigonometric equation for circle?

Ans: $X^2 + Y^2=R^2$ where $X= R\cos\theta$, $Y= R\sin\theta$

OUTPUT: Printout of program with output.

ASSIGNMENT NO: 2

AIM: To draw the polygon and fill the color inside the polygon

PROBLEM STATEMENT A : Write C++ program to draw a concave polygon and fill it with desired color using scan fill algorithm. Apply the concept of inheritance.

PREREQUISITES:

1. Knowledge of scan conversion, Raster scan, Polygon types
2. Basic mathematics

COURSE OBJECTIVE:

1. To acquaint the learner with the basic concepts of Computer Graphics
2. To get familiar with mathematics behind the graphical objects

COURSE OUTCOME:

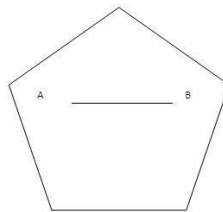
1. Develop scientific and strategic approach to solve complex problems in the domain of Computer Graphics

THEORY:

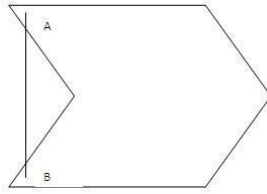
Polygon: polygon is a diagram in which start point and end point is connected.

Types of Polygons: Convex and Concave

Convex Polygons: In a convex polygon, any line segment joining any two inside points lies inside the polygon.

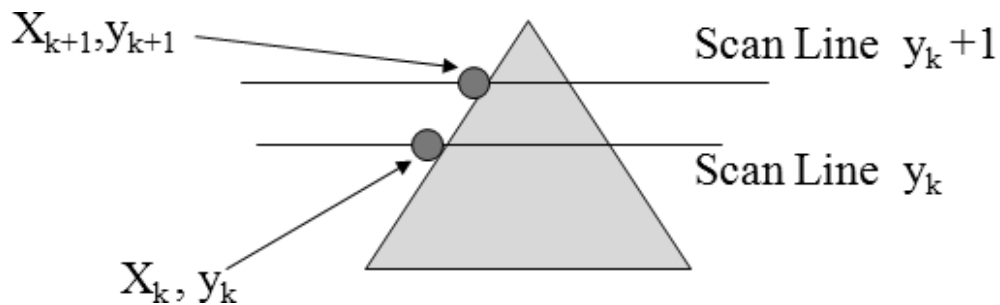


Concave Polygons: In a concave polygon, a line segment connecting any two points may or may not lie inside the polygon.



Scan Line Polygon Fill Algorithms

- A standard output primitive in general graphics package is a solid color or patterned polygon area:
- There are two basic approaches to filling on raster systems.
- Determine overlap Intervals for scan lines that cross that area.
- Start from a given interior point and paint outward from this point until we encounter the boundary
- The first approach is mostly used in general graphics packages, however second approach is used in applications having complex boundaries and interactive painting systems



- For each scan lines crossing a polygon are then sorted from left to right, and the corresponding frame buffer positions between each intersection pair are set to the specified color.
- These intersection points are then sorted from left to right , and the corresponding frame buffer positions between each intersection pair are set to specified color
- **Some scan-Line intersections at polygon vertices require special handling:**
The scan conversion algorithm works as follows

- i. Intersect each scanline with all edges
- ii. Sort intersections in x
- iii. Calculate parity of intersections to determine in/out
- iv. Fill the “in” pixels

Special cases to be handled:

- i. Horizontal edges should be excluded
 - ii. For vertices lying on scanlines,
 - i. count twice for a change in slope.
 - ii. Shorten edge by one scanline for no change in slope
- Coherence between scanlines tells us that
 - Edges that intersect scanline y are likely to intersect y + 1
 - X changes predictably from scan line y to y + 1

Test Cases

T_ID	T_Name	Condition to test	Expected outcome	Pass/Fail
T_F1	Seed point checking	Selected pixel is inside the polygon	Polygon will fill with given color	
		Selected pixel is outside the polygon	Polygon will not fill	
T_F2	Polygon Size testing	Polygon of small size	Get filled	
		Polygon of large size	Due to recursive fall system error will occur	

CONCLUSION: Scan Line algorithm can be used for filling the simple as well as complex polygon efficiently.

FAQs

1. What are active edges?

Ans: Edges that are inserted by current scan line are called active edges.

2. What is scan line?

Ans: Horizontal line across the y axis

3. What is span?

Ans: pixels between to active edges

4. What is complex polygon?

Ans: intersecting or overlapping polygons

5. Why to sort ymax and y min of polygon edges?

Ans: So algorithm will start at ymin or ymax of polygon instead of ymin and ymax of window.

6. Why to sort xmin and xmax?

Ans: To make pair for filling per active edge

OUTPUT: Program with output.

ASSIGNMENT NO: 3

AIM: To Study line clipping algorithm

PROBLEM STATEMENT: Write C++ program to implement Cohen Sutherland line clipping algorithm.

PREREQUISITES:

- Knowledge of windowing

COURSE OBJECTIVE:

1. To acquaint the learner with the basic concepts of Computer Graphics
2. To learn the various algorithms for generating and rendering graphical figures
3. To get familiar with mathematics behind the graphical transformations

COURSE OUTCOME:

1. Develop scientific and strategic approach to solve complex problems in the domain of Computer Graphics
2. Develop the competency to understand the concepts related to Computer Vision and Virtual reality

THEORY:

Clipping :

Clipping, in the context of computer graphics, is a method to selectively enable or disable rendering operations within a defined region of interest. A rendering algorithm only draws pixels in the intersection between the clip region and the scene model. In short clipping is cutting the part of scene which is not fitted in window.

Cohen-Sutherland line Clipping Algorithm:

1. Set outcodes to the regions



//above left below right

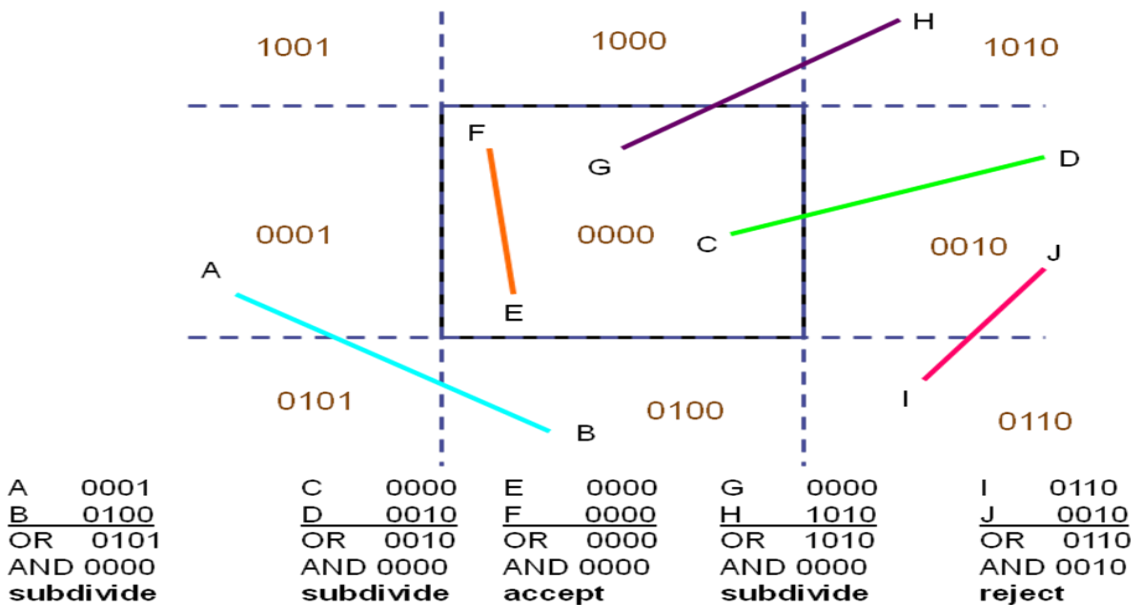
```
int outcode(x,y,xmin,xmax,ymin,ymax)
{
    int code=0;
    if(x > xmax) code |= 1; //right if(y < ymin) code |= 2; //below
    if(x < xmin) code |= 4; //left if(y > ymax) code |= 8; //above
    return(code);
}
```

2. Identify lines that are trivially accepted or trivially rejected.

1. Accept (and draw) lines that have both endpoints inside the region
2. Reject (and don't draw) lines that have both endpoints less than x_{min} or y_{min} or greater than x_{max} or y_{max}
3. Clip the remaining lines at a region boundary and repeat steps 1 and 2 on the clipped line segments

```
if(outcode(P)OR outcode(Q)==0000) accept
    else if(outcode(P)And outcode(Q))!=0)
        reject
else test further
```

3. If a line segment PQ falls into the “test further” category then
4. Stop
5. if (outcode(P) & 1000 != 0)
 replace P with PQ intersect y = top else if
 (outcode(Q) & 1000 != 0)
 replace Q with PQ intersect y = top go on to next boundary



CONCLUSION: Cohen Sutherland algorithm is implemented for line clipping.

FAQ:

1. What are types clipping algorithms?

Ans: Point clipping, Line Clipping, Polygon Clipping and Text Clipping

2. What is viewing transformation?

Ans: Mapping of picture coordinates to display coordinates is viewing transformation

3. What is windowing?

Ans: The process of selecting and viewing the picture with different views is called windowing.

4. What is window?

Ans: To perform viwing transformation we select a finite world co-ordinate area for display called a windowing.

5. What is viewport?

Ans: An area on device to which window is mapped

6. Name polygon clipping algorithm

Ans: Sudherland Hodgman polygon clipping algorithm

OUTPUT: Program with output.

GROUP: A

ASSIGNMENT NO: 4

AIM: To apply the basic 2D transformations such as translation, Scaling, Rotation for a given 2D object.

PROBLEM STATEMENT A : Write C++/Java program to draw 2-D object and perform following basic transformations,

- b) Scaling
- c) Translation
- d) Rotation

Use operator overloading.

PROBLEM STATEMENT B: Write C++/Java program to implement translation, sheer, rotation and scaling transformations on equilateral triangle and rhombus.

PROBLEM STATEMENT B: Write a program to implement scaling, translation on 3-D cube using OpenGL.

PREREQUISITES:

1. Knowledge of matrix fundamentals and basic transformations on polygon - translation, rotation & scaling.
2. Basic mathematics, vectors & matrices

COURSE OBJECTIVE:

1. To acquaint the learner with the basic concepts of Computer Graphics
2. To get familiar with mathematics behind the graphical transformations

COURSE OUTCOME:

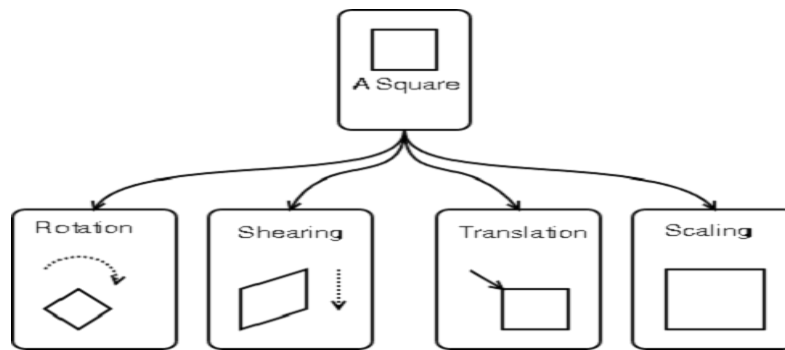
1. Develop scientific and strategic approach to solve complex problems in the domain of Computer Graphics

THEORY:

A. Transformations:

Transformations allow us to uniformly alter the entire picture. The geometric transformations considered here - translation, scaling and rotation are expressed in terms of matrix multiplication.

Example:



Homogenous Coordinates

To perform a sequence of transformation such as translation followed by rotation and scaling, we need to follow a sequential process –

- **Translate the coordinates,**
 - **Rotate the translated coordinates, and then**
- **Scale the rotated coordinates to complete the composite transformation.**

To shorten this process, we have to use 3×3 transformation matrix instead of 2×2 transformation matrix. To convert a 2×2 matrix to 3×3 matrix, we have to add an extra **dummy coordinate W**.

In this way, we can represent the point by 3 numbers instead of 2 numbers, which is called Homogenous Coordinate system. In this system, we can represent all the transformation equations in matrix multiplication.

Scaling

scaling refers to changing the size of the object either by increasing or decreasing. We will increase or decrease the size of the object based on scaling factors along x and y-axis.

Scaling can be achieved by multiplying the original coordinates of the object with the scaling factor to get the desired result.

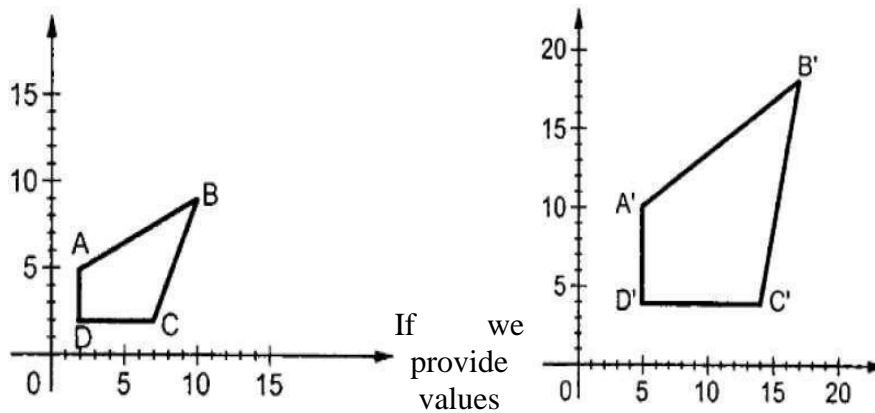
$$X' = X \cdot S_X \text{ and } Y' = Y \cdot S_Y$$

The scaling factor S_X, S_Y scales the object in X and Y direction respectively. The above equations can also be represented in matrix form as below –

$$\begin{array}{lcl} X' & X & S_x \quad 0 \\ Y' = & Y & 0 \quad S_y \end{array}$$

- scaling is performed about the origin (0,0) not about the center of the line/polygon/whatever

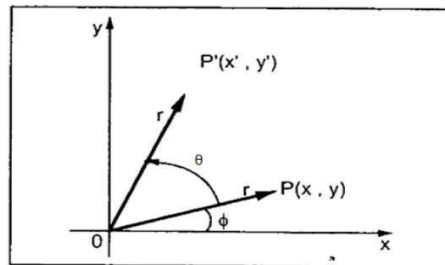
- uniform scaling: $S_x = S_y$
- differential scaling $S_x \neq S_y \rightarrow$ alters proportions



less than 1 to the scaling factor S , then we can reduce the size of the object. If we provide values greater than 1, then we can increase the size of the object.

Rotation

In rotation, we rotate the object at particular angle θ (theta) from its origin. From the following figure, we can see that the point $P(X, Y)$ is located at angle ϕ from the horizontal X coordinate with distance r from the origin.



Using standard trigonometric the original coordinate of point $P(X, Y)$ can be represented as –

$$X = r \cos \phi \quad (1)$$

$$Y = r \sin \phi \quad (2)$$

Same way we can represent the point $P'(X', Y')$ as –

$$x' = r \cos(\phi + \theta) = r \cos \phi \cos \theta - r \sin \phi \sin \theta \quad (3)$$

$$y' = r \sin(\phi + \theta) = r \cos \phi \sin \theta + r \sin \phi \cos \theta. \quad (4)$$

Substituting equation (1) & (2) in (3) & (4) respectively, we will get

$$x' = x \cos\theta - y \sin\theta$$

$$y' = x \sin\theta + y \cos\theta$$

Representing the above equation in matrix form,

$$\begin{matrix} X' \\ Y' \end{matrix} = \begin{matrix} X \\ Y \end{matrix}$$

OR $P' = P \cdot R$

Where R is the rotation matrix

$$R = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix}$$

The rotation angle can be positive and negative.

For positive rotation angle, we can use the above rotation matrix.

However, for negative angle rotation, the matrix will change as shown below –

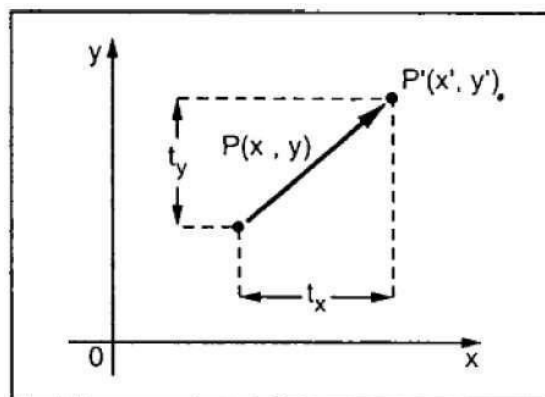
$$R = \begin{pmatrix} \cos(-\theta) & \sin(-\theta) \\ -\sin(-\theta) & \cos(-\theta) \end{pmatrix}$$

$$= \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \quad (\because \cos(-\theta) = \cos\theta \text{ and } \sin(-\theta) = -\sin\theta)$$

Translation

A translation moves an object to a different position on the screen. You can translate a point in 2D by adding translation coordinate (t_x, t_y) to the original coordinate (X, Y) to get the new coordinate (X', Y') .

Translation Distance: It is nothing but by how much units we should shift the object from one location to another along x, y-axis.



From the above figure, you can write that –

$$X' = X + t_x$$

$$Y' = Y + t_y$$

The pair (t_x, t_y) is called the translation vector or shift vector. The above equations can also be represented using the column vectors.

$$P = [X] / [Y]$$

$$p' = [X'] / [Y']$$

$$T = [t_x] / [t_y]$$

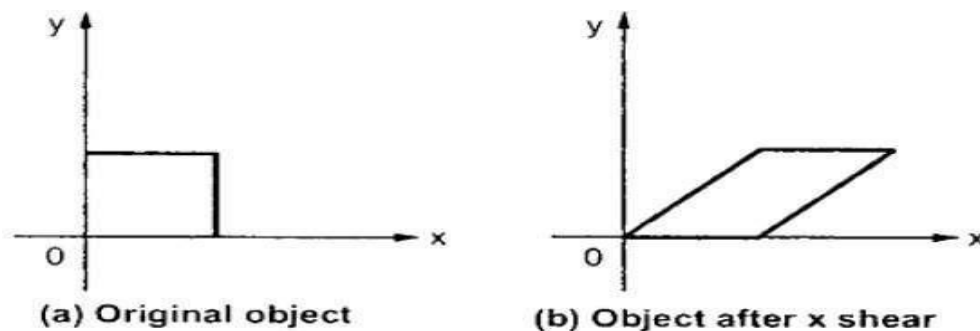
We can write it as - $P' = P + T$

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 as shown in below figure.



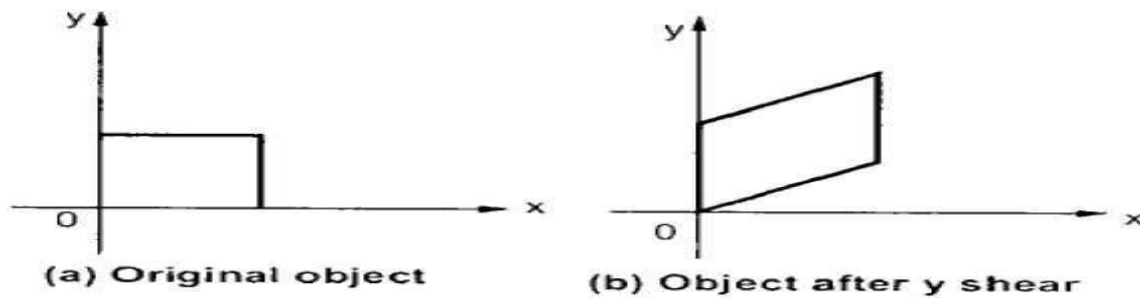
The transformation matrix for X-Shear can be represented as –

$$X_{sh} = X' = X + Sh_x \cdot Y \quad Y' = Y$$

Y-Shear

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

horizontal lines to transform into lines which slopes up or down as shown in the following figure.



The Y-Shear can be represented in matrix form as -

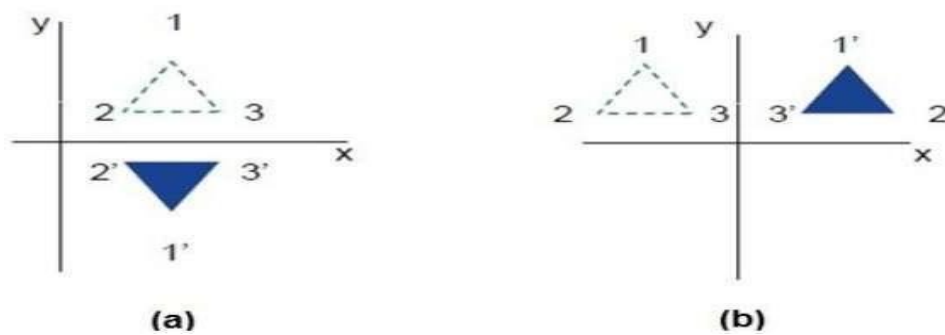
$$Y_{sh} = \begin{bmatrix} 1 & Sh_y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Y' = Y + Sh_y \cdot X$$

$$X' = X$$

Reflection

Reflection in computer graphics is used to emulate reflective objects like mirrors and shiny surfaces.



Homogeneous Matrices:

Translation $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ T_x & T_y & 1 \end{bmatrix}$	Scaling $\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$
Rotation $\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	Reflection (about origin) $\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
X- Shear $\begin{bmatrix} 1 & 0 & 0 \\ Sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	Y-Shear $\begin{bmatrix} 1 & Sh_y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Algorithm:

1. Read input object in the form of matrix
2. Multiply input matrix with transformation matrix
3. Repeat same for all transformations
4. Display transformed object
5. Stop

OOP's concepts used

- 1. Class**
- 2. Operator Overloading**

CONCLUSION: Hence we have studied and implemented 2D transformations such as translation, Scaling, Rotation for a given 2D object.

FAQs:

- 1. Perform a counterclockwise 45^0 rotation of triangle A(2,3), B(5,5), C(4,3) about point (1,1).**
- 2. A point (4,3) is rotated counterclockwise by an angle of 45^0 . Find the rotation matrix and the resultant point.**
- 3. Translate the polygon with co-ordinates A(2,5), B(7,10) and C(10,2) by 3 units in x direction and 4 units in y direction.**
- 4. Scale the polygon with co-ordinates A(2,5), B(7,10) and C(10,2) by 2 units in x direction and 2 units in y direction.**
- 5. Give a 3×3 homogeneous co-ordinate transformation matrix for each of following translations**
 - a) Shift image to right 3 units**
 - b) Shift image up 2 units**
 - c) Move image down 1 units**

OUTPUT: Printout of program with output.

ASSIGNMENT NO: 5

AIM: To study fractals

PROBLEM STATEMENT: Write C++ program to generate fractal patterns by using Koch curves.

PREREQUISITES:

- Basic knowledge of mathematic primitives

COURSE OBJECTIVE:

1. To acquaint the learner with the basic concepts of Computer Graphics
2. To learn the various algorithms for generating and rendering graphical figures

COURSE OUTCOME:

1. Develop the competency to understand the concepts related to Computer Vision and Virtual reality

THEORY:

Fractals : Fractals are very complex pictures generated by a computer from a single formula. They are created using iterations. This means one formula is repeated with slightly different values over and over again, taking into account the results from the previous iteration.

Fractals are used in many areas such as –


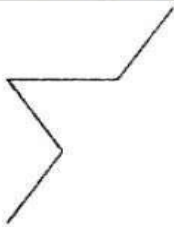
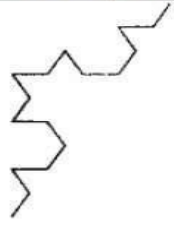
Astronomy – For analyzing galaxies, rings of Saturn, etc.

Biology/Chemistry – For depicting bacteria cultures, Chemical reactions, human anatomy, molecules, plants,

Others – For depicting clouds, coastline and borderlines, data compression, diffusion, economy, fractal art, fractal music, landscapes, special effect, etc.

Koch curve :

The Koch curve is also known as snowflake curve. It is drawn by dividing line into 4 equal segments with scaling factor $1/3$ & middle 2 segments are so adjusted that they form adjacent sides of equilateral triangle. Repeat same process for each of the 4 segments. Each repetition increases length by factor $4/3$.

Segment Length = 1	Segment Length = $1/3$	Segment Length = $1/9$
		
Length = 1	Length = $4/3$	Length = $16/9$

Fractal dimension:

$$4=3^D$$

$$D = \log_3 4 = \log 4 / \log 3 = 1.2618$$

Therefore for Koch curve topological dimension is 1, but fractal dimension is 1.2618 hence Koch curve is fractal.

Algorithm

1. Divide line into 4 equal segments with scaling factor $1/3$.
2. Adjust middle 2 segments that they form adjacent sides of equilateral triangle.
3. Repeat same process for each of the 4 segments.
4. Each repetition increases length by factor $4/3$.
5. Stop

CONCLUSION: Fractals are studied by implementing Koch curve.

FAQs:**1. What is topological dimension?**

Ans : IT specifies how the object is deform in the space. Ex. Line is 1D, Square is 2D & Cube is 3D.

2. How to compute Fractal dimension?

Ans: $D = \log N / \log s$

3. What are different types of fractals?

Ans: Self Similar, Self Affine & Invariant

4. Name any other

fractal. Ans: Hilberts curve

OUTPUT : Program with Output

ASSIGNMENT NO: 6

AIM: To create simple objects using OpenGL

PROBLEM STATEMENT: Write OpenGL program to draw Sun Rise and Sun Set

PREREQUISITES:

1. Basic programming skills of OpenGL
2. Graphics Primitive for OpenGL

COURSE OBJECTIVE:

1. To acquaint the learner with the basic concepts of Computer Graphics
2. To understand and apply various libraries and functions regarding OpenGL & its APIs

COURSE OUTCOME:

1. Apply the logic to develop sun set and sun rise

THEORY:

OpenGL

Open Graphics Library (OpenGL) is a cross-language, cross-platform application programming interface (API) for rendering 2D and 3D vector graphics. The API is typically used to interact with a graphics processing unit (GPU).

OpenGL is used in various applications of computer-aided design (CAD), virtual reality, scientific visualization, information visualization, flight simulation, and video games.

Library Files:

1. Basic GL: The fundamental OpenGL library. Each function starts with 'GL'
e.g. `include<gl/GL.h>`
2. GLUT: GL utility toolkit. Related with managing windows, menu and events
e.g. `include<gl/GLUT.h>`
3. GLU: Involves high level routines to handle matrix operation , drawing 3D surfaces.
e.g. `include<gl/GLU.h>`
4. GLUI: User interface library. Control and menus can be added to application.

OpenGL Primitives: Following are the graphics primitives present in OpenGL:

Vertex Primitives: Command for vertex representation is explained below.

e.g. glVertex2i(..)

gl: library **Vertex:** Basic command **2:** no. of arguments **i:** datatype

Point Primitives: This will cause OpenGL to interpret each individual vertex in the stream as a point.

- GL_POINTS

Line Primitives:

There are 3 kinds of line primitives, based on different interpretations of a vertex stream.

- ☐ GL_LINES: Vertices 0 and 1 are considered a line.
- ☐ GL_LINE_STRIP: The adjacent vertices are considered lines.
- ☐ GL_LINE_LOOP: As line strips, except that the first and last vertices are also used as a line.

CONCLUSION: Simple animation is created.

FAQs:

- 1. List all OpenGL APIs**
- 2. Explain OpenGL instruction format**

ASSIGNMENT NO: 7

AIM: To implement simple animation

PROBLEM STATEMENT: Write C++ program to draw man walking in the rain with an umbrella. Apply the concept of polymorphism.

PREREQUISITES:

1. Basic programming skills of C++
2. Graphics Primitive for C++
3. Basic steps to generate animation

COURSE OBJECTIVE:

3. To acquaint the learner with the basic concepts of Computer Graphics
4. To understand and apply various methods and techniques regarding projections, animation, shading, illumination and lighting

COURSE OUTCOME:

1. Apply the logic to develop animation and gaming programs

THEORY:

Animation means giving life to any object in computer graphics. Computer-assisted animation and computer-generated animation are two categories of computer animation. It can be presented via film or video.

The basic idea behind animation is to play back the recorded images at the rates fast enough to fool the human eye into interpreting them as continuous motion. Animation can be used in many areas like entertainment, computer aided-design, scientific visualization, training, education, e-commerce, and computer art.

Animation Techniques

Animators have invented and used a variety of different animation techniques. Basically there are six animation techniques which we would discuss one by one in this section.

1. Traditional Animation (frame by frame)

Traditionally most of the animation was done by hand. All the frames in an animation had to be

drawn by hand. Since each second of animation requires 24 frames (film), the amount of efforts required to create even the shortest of movies can be tremendous.

2. Keyframing

In this technique, a storyboard is laid out and then the artists draw the major frames of the animation. Major frames are the ones in which prominent changes take place. They are the key points of animation. Keyframing requires that the animator specifies critical or key positions for the objects. The computer then automatically fills in the missing frames by smoothly interpolating between those positions.

3. Procedural

In a procedural animation, the objects are animated by a procedure – a set of rules – not by keyframing. The animator specifies rules and initial conditions and runs simulation. Rules are often based on physical rules of the real world expressed by mathematical equations.

4. Behavioral

In behavioral animation, an autonomous character determines its own actions, at least to a certain extent. This gives the character some ability to improvise, and frees the animator from the need to specify each detail of every character's motion.

5. Performance Based (Motion Capture)

Another technique is Motion Capture, in which magnetic or vision-based sensors record the actions of a human or animal object in three dimensions. A computer then uses these data to animate the object.

This technology has enabled a number of famous athletes to supply the actions for characters in sports video games. Motion capture is pretty popular with the animators mainly because some of the commonplace human actions can be captured with relative ease. However, there can be

serious discrepancies between the shapes or dimensions of the subject and the graphical character and this may lead to problems of exact execution.

6. Physically Based (Dynamics)

Unlike key framing and motion picture, simulation uses the laws of physics to generate motion

of pictures and other objects. Simulations can be easily used to produce slightly different sequences while maintaining physical realism. Secondly, real-time simulations allow a higher degree of interactivity where the real person can maneuver the actions of the simulated character.

In contrast the applications based on key-framing and motion select and modify motions form a pre-computed library of motions. One drawback that simulation suffers from is the expertise and time required to handcraft the appropriate controls systems.

Design of Animation Sequences

1. Layout of Storyboard: Storyboard layout is the action outline utilized to illustrate the motion sequence as a set of storyboard comprises a set of rough sketches or a list of basic concepts for the motion.
2. Definition of Object & Path : The object definition is specified for all participant objects in action. The objects can be explained in terms of fundamental shapes, related movements or movement with shapes.
3. Specification of Key Frame: this is the detailed drawing of the scene at an exact time in the animation sequence. Inside each key frame, all objects are positioned as per to time for that frame. Several key frames are selected at the extreme positions in the action; More key frames are given for intricate motion than for easy, slowly varying motions.
4. In-between frames Generation: In-between frames are the middle frames among the key frames. In common, film needs twenty-four frames per second, and graphic terminals are refreshed on the rate of 30 to 60 frames per second. Classically the time interval for the motion is set up hence there are 3 to 5 among for each pair of key frames. Based upon the speed identified for the motion, several key frames can be duplicated.

Motion Specifications

In Various ways in which motions of objects can be specified as:

- Direct Motion Specification: Here the rotation angles and translation vectors are explicitly given. Then the geometric transformation matrices are applied to transform coordinate positions. We can approximate the path of a bouncing ball with a damped, rectified, sine curve
- Goal-Directed Systems : We can specify the motions that are to take place in general terms that abstractly describe the actions. These systems are referred to as goal directed because they determine specific motion parameters given the goals of the animation. For example, We could specify that we want an object to "walk " or to "run" to a particular destination. Or We could state that we want an object to "pick up " some other specified object.
- Kinematics and Dynamics :

Kinematics description, specify the animation by giving motion parameters (position, velocity and acceleration) No reference to causes or goals of the motions Inverse kinematics, specify the initial and final positions of object at specified times and motion parameters are computed by system. Dynamic description, require the specification of the forces that produce the velocities and accelerations.

Algorithm:

1. Draw object using basic graphics primitives
2. Create animation by increasing or decreasing in x direction or y direction
3. Keep delay of predefined milliseconds

CONCLUSION: Simple animation is created.

FAQs:

1. List down films based on animation
2. List down companies that produce animation
3. List down popular animation games

OUTPUT: Program with output

ASSIGNMENT NO: 8

AIM: To learn Open Source Tool for Animation

PROBLEM STATEMENT: A. Write a program to generate Bouncing ball animation using Blender

Content beyond Syllabus: B. *Create screen saver using Blender*

PREREQUISITES:

- Knowledge of animation tools

COURSE OBJECTIVE:

1. To understand and apply various methods and techniques regarding projections, animation, shading, illumination and lighting

COURSE OUTCOME:

1. Apply the logic to develop animation and gaming programs

THEORY:

Blender is a professional free and open-source 3D computer graphics software product used for creating animated films, visual effects, art, 3D printed models, interactive 3D applications and video games. Blender's features include 3D modeling, UV unwrapping, texturing, raster graphics editing, rigging and skinning, fluid and smoke simulation, particle simulation, soft body simulation, sculpting, animating, match moving, camera tracking, rendering, video editing and compositing.

Features

Official releases of Blender for Microsoft Windows, Mac OS X, and Linux, as well as a port for FreeBSD, are available in both 32-bit and 64-bit versions.

- Support for a variety of geometric primitives, including polygon meshes, fast subdivision surface modeling, Bezier curves, NURBS surfaces, metaballs, multi-res digital sculpting (including dynamic topology, maps baking, remeshing, resymetrize, decimation..), outline font, and a new n-gon modeling system called B-mesh.
- Keyframed animation tools including inverse kinematics, armature (skeletal), hook, curve and lattice-based deformations, shape keys (morph target animation), non-linear animation, constraints, and vertex weighting
- A particle system that includes support for particle-based hair.
- Modifiers to apply non-destructive effects.

- Python scripting for tool creation and prototyping, game logic, importing and/or exporting from other formats, task automation and custom tools.
- Basic non-linear video/audio editing.
- The Blender Game Engine, a sub-project, offers interactivity features such as collision detection, dynamics engine, and programmable logic. It also allows the creation of stand-alone, real-time applications ranging from architectural visualization to video game construction.

Algorithm:

- 1) Take UVSphere on Mesh from Add-> Mesh->UVSphere toolbar
- 2) Add key frame style movement to the ball. With the ball selected press i to insert a location key frame. Now press the up arrow we will see the number of key frames increase on the bottom of the screen 51 represents the number of frames. Go up to 75 key frames so that the ball has a smooth 3 second bounce.
- 3) Save the location of ball in every keyframe, press s.
- 4) Run the Animation ,click on start button or Press P Button from Keyboard

CONCLUSION:

- Blender is free and open-source 3D computer graphics software product used for creating animations.
- Screen Saver is created as a part of **content beyond syllabus**

FAQs:

1. What is MAYA software?

Ans: Maya 3D animation, modeling, simulation and rendering software offers artists a comprehensive creative tool set.

2. List Open Source Software for creating Animation

Ans: The Best Free / Open Source Animation Software: 2D Animation

- 1) Pencil
- 2) Synfig Studios
- 3) Stykz
- 4) CreaToon
- 5) Ajax Animator

The Best Free / Open Source Animation Software: 3D Animation

- 1) Blender
- 2) Bryce
- 3) DAZ Studio
- 4) Clara.io

3. What is animation?

Ans: Object in motion is called as animation

4. What is fps?

Ans: Frames per Second.

5. List out applications of animation.

Ans: Games, Cartoons, Movie making.

OUTPUT: program with output.