COMPUTER GRAPHICS (CCG3013)

LESSON 3

GEOMETRY IN 2D GRAPHICS

COURSE OUTLINE

Lesson	Topic
1	Introduction to computer graphics
2	Graphics hardware and software
3	Geometry in 2D graphics
4 & 5	Geometry in 3D graphics
6 & 7	User interfaces and interactions
8	Colour
9 & 10	Motion and animation
11	Lighting and rendering
12	Surface shadings

TOPIC LEARNING OUTCOMES

1. Describe and illustrate primitive shapes in two-dimensional (2D) space.

2. Compute matrix transformations in 2D space.

3. Explain and implement 2D drawing functions.

ASSESSMENTS

Structure	Marks (%)	Hand-out	Hand-in
Assignment 1 (Individual)	30	Week 1(Unofficial) Week 3(Official)	Week 6
Assignment 2 (Group up to four only)	30	Week 1(Unofficial) Week 3(Official)	Week 12
Final examination	40	Exam week	

REPLACEMENT CLASS

None.

CONTENT

No.	Topics	Duration (Minutes)
1	Mini lecture 1: 2D coordinates system	15
2	Exercise 1	10
3	Mini lecture 2: 2D primitive shapes	15
4	Exercise 2	10
5	Break	10
6	Mini lecture 3: 2D drawing functions	15
7	Exercise 3	10
8	Mini lecture 4: Matrix transformation in 2D space	15
9	Exercise 4	10

REVIEW I: GPU AND TOOLS

1. There are three main modules in GPU, which are driver, graphics accelerator, and projection.

2. The specifications for a GPU, include memory, resolution, texture fill rate, connectivity, and polygon counts.

3. Six components of computer graphics include display, renderer, matrices transformation, inputs and callbacks, motion and animations, and maps.

4. Four graphics accelerator included OpenGL, DirectX, Processing, and CUDA.

REVIEW II: GRAPHICS LIBRARIES

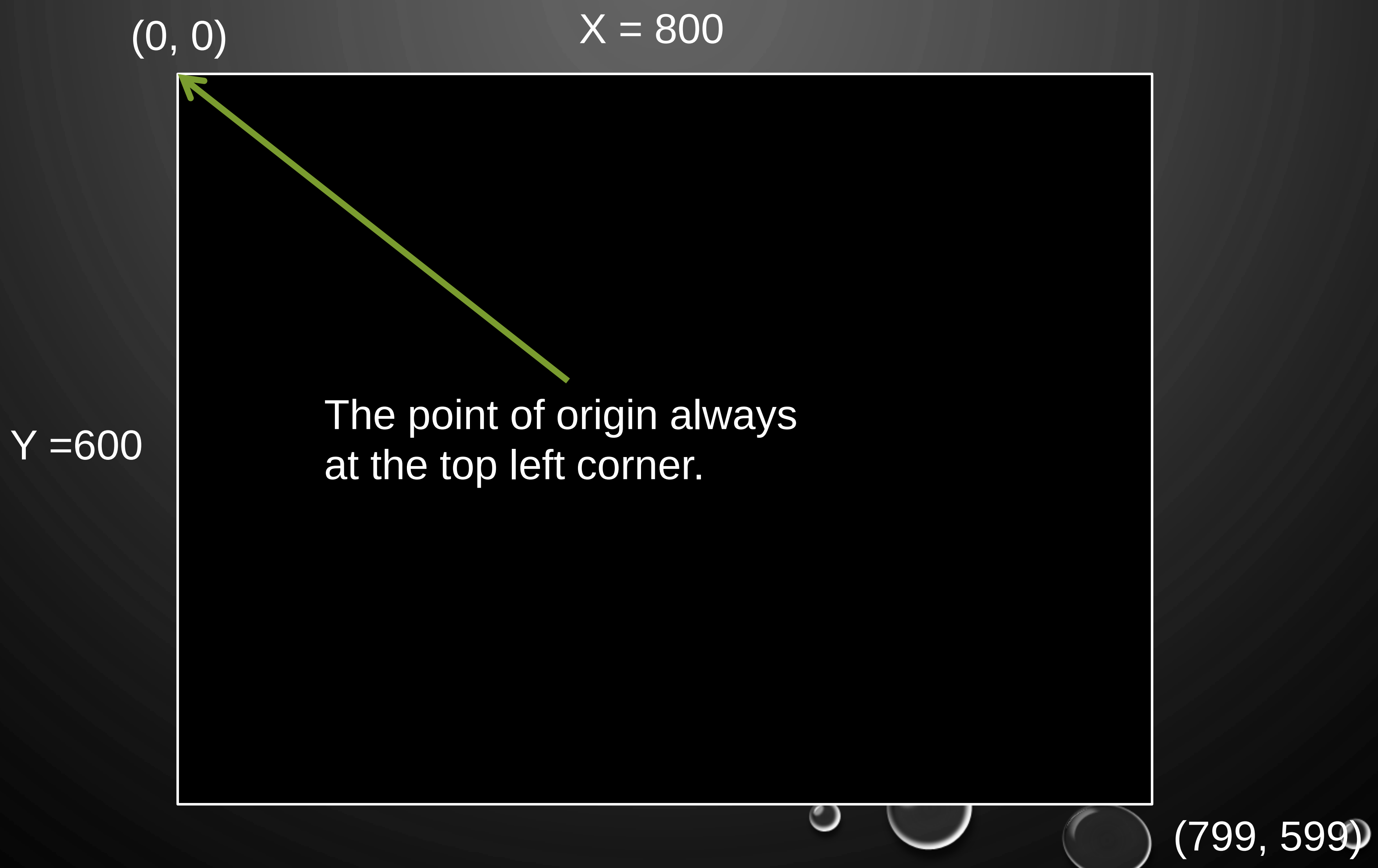
1. GLUT stands for OpenGL utility toolkit.
2. GLUT performs I/O controls for a host operating system.
3. GLU stands for OpenGL utility library.
4. GLU provides the primitive functions for OpenGL.
5. Functions of GLU included coordinates system, view settings, 2D/3D primitive modelling, and mappings.

REVIEW III: DISPLAY FUNCTIONS

Window	Canvas
glutInitDisplaymode()	glClearColor()
glutDisplayFunc()	glClear()
glutMainLoop()	glFlush()
glutCreateWindow()	glFinish()
glutInitWindowSize()	
glutInitWindowPosition()	

MINI LECTURE 1
2D COORDINATES SYSTEM
…

OPENGL WINDOW



2D COORDINATES SYSTEM

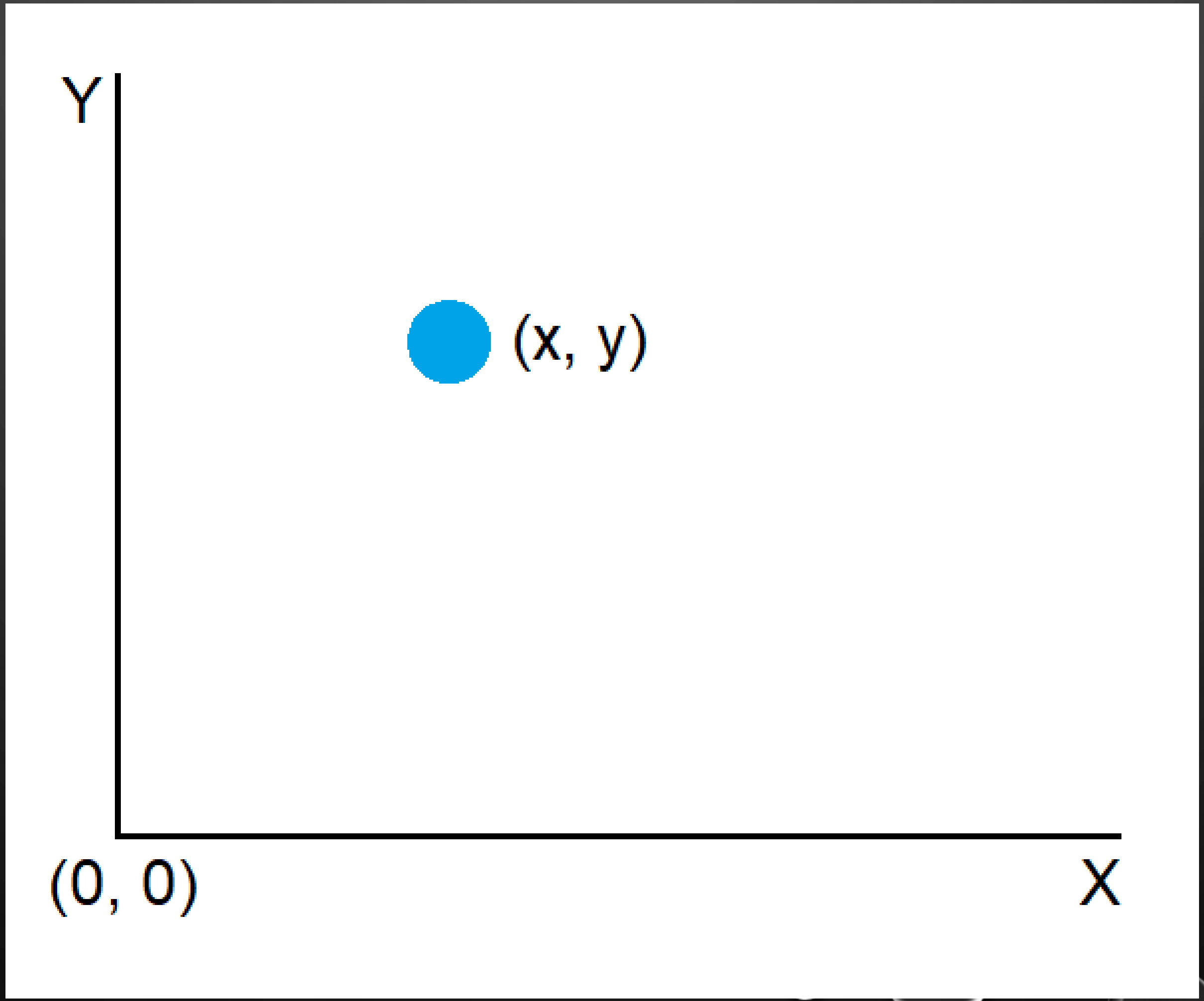


IMAGE RESOLUTION

1. It specifies the dimension of an image.
2. It is the width times the height of an image.
3. Image resolution, res = width × height.
4. It measures in pixels.

ASPECT RATIO

1. It is the ratio of the width to the height of an image.
2. Aspect ratio, ar = width : height.
3. For instance, a 500 × 400 digital image has the aspect ratio of 5 : 4.
4. The aspect ratio for a normal screen is 4 : 3.
5. The aspect ratio for a widescreen is 16 : 9.

EXERCISE 1

This activity will takes about ten minutes.

1. Define resolution.

2. Define aspect ratio.

3. Compute resolution for 640 × 480, 800 × 600, 720p, 1080p, 4K.

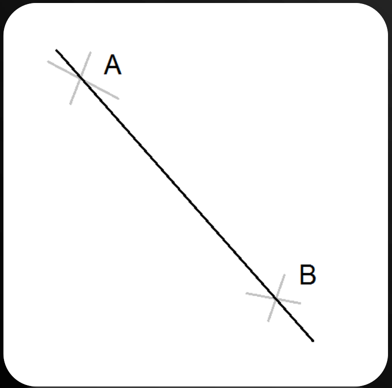
4. Compute aspect ratio for image sizes in Question 3.

MINI LECTURE 2
2D PRIMITIVE SHAPES

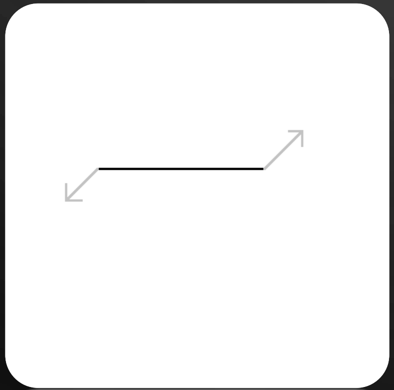
...

EUCLID'S POSTULATES/ EUCLID'S AXIOMS

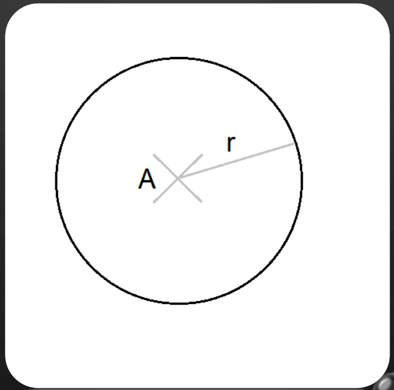
1. Given two points, it is possible to draw a right line.
2. The right line can be extended in both directions.
3. Given a center and a radius, we can draw a circle.
4. All right angles are equal.
5. Parallel postulate.



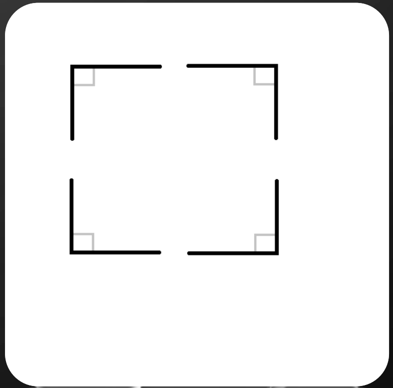
Postulate 1



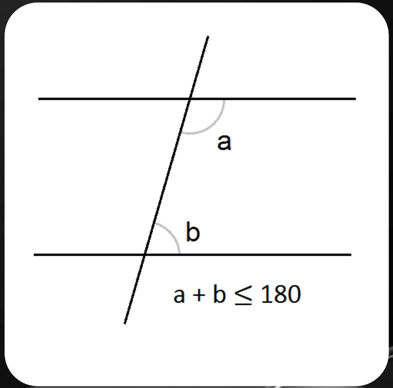
Postulate 2



Postulate 3



Postulate 4



Postulate 5

2D SHAPES

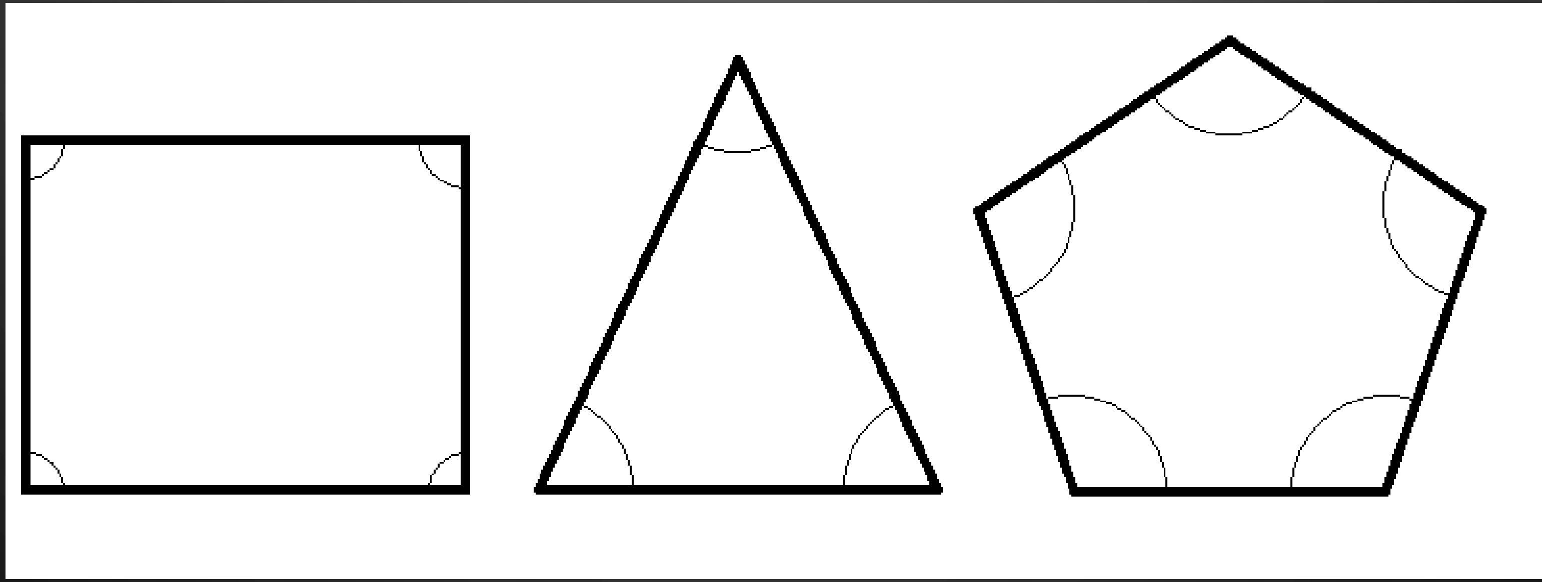
1. It is an outline that render in two-dimensional (2D) space.

2. There are two types of shape, which are convex shapes and concave shapes.

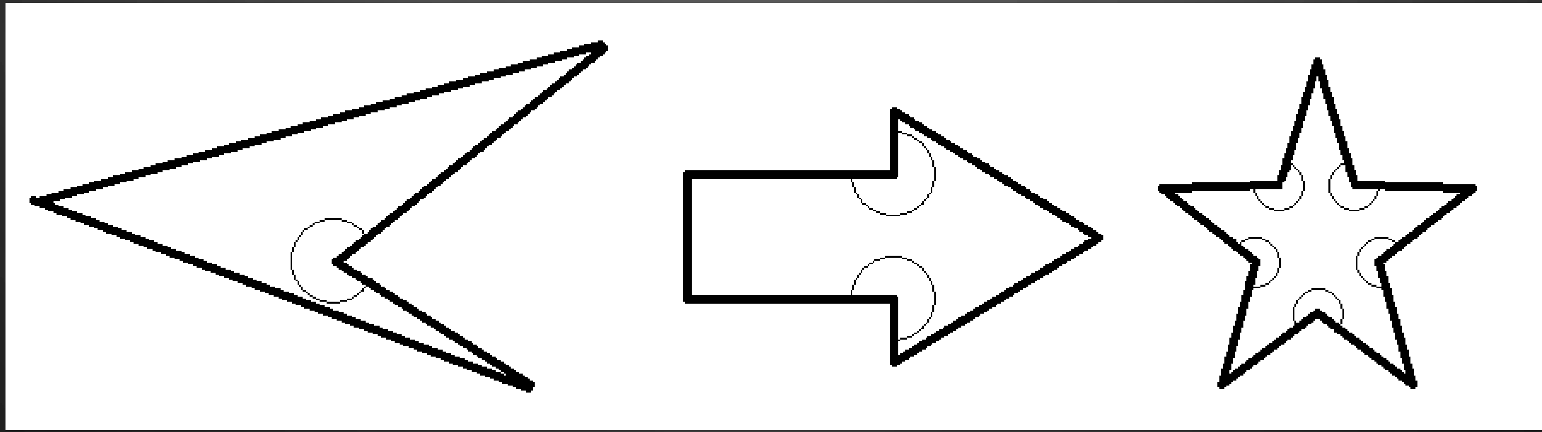
3. Convex shape, all inner angles should be less than 180 degrees.

4. Concave shape, at least one of the inner angles should be more than 180 degrees.

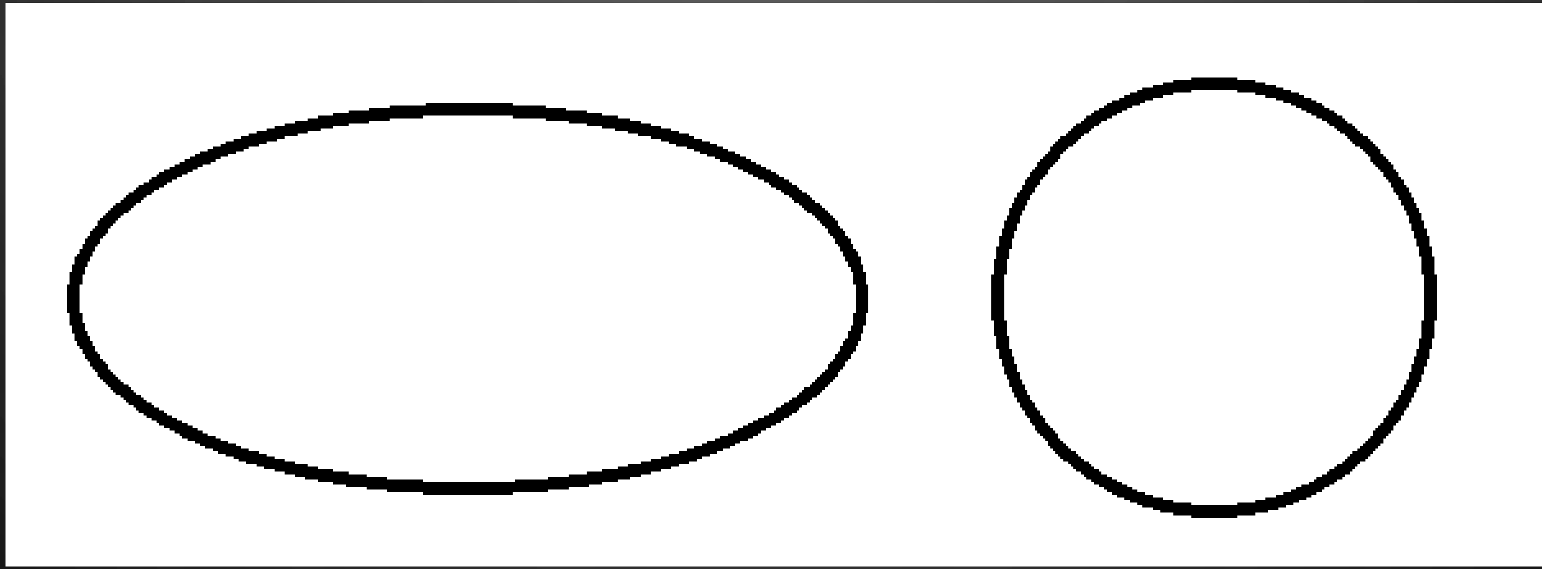
SHAPE TYPE: CONVEX



SHAPE TYPE: CONCAVE



SHAPE TYPE: CONVEX HULL

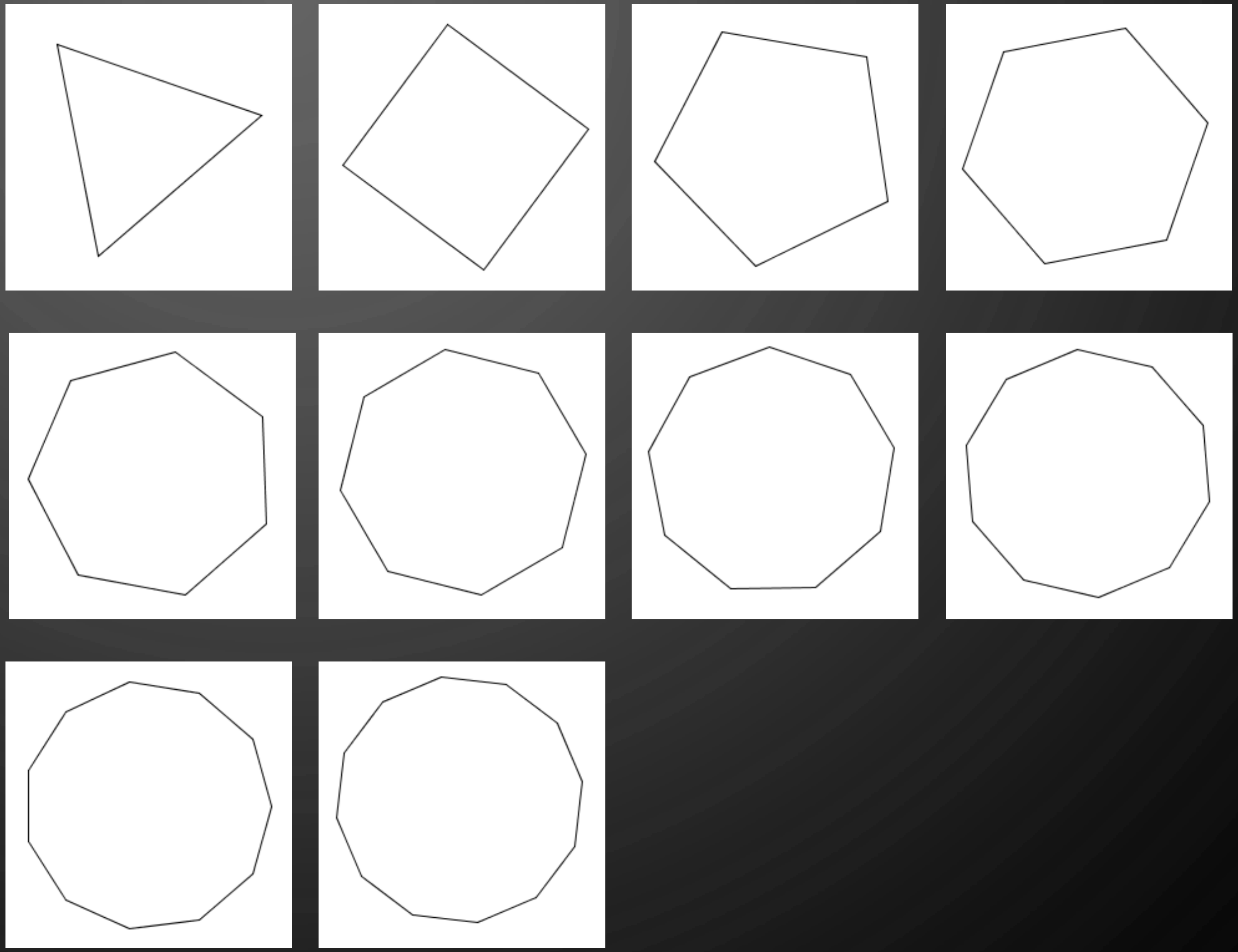


POLYGONS

1. It also called n-gons.
2. Poly means many, while gons mean angles.
3. It is geometrical term for shapes.
4. It can be convex or concave.
5. All regular polygons are of type convex.

REGULAR POLYGONS

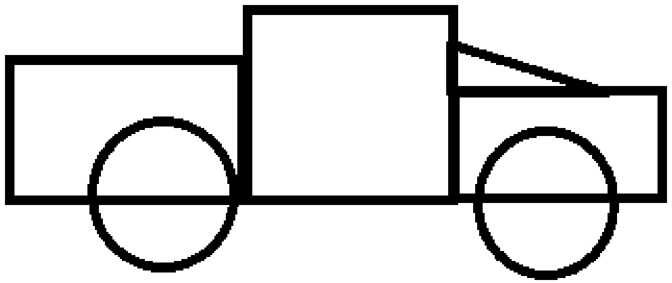
Shapes	**N-gons**
Triangle	3-gons
Square	4-gons
Pentagon	5-gons
Hexagon	6-gons
Heptagon	7-gons
Octagon	8-gons
Nanogon	9-gons
Decagon	10-gons
Hendecagon	11-gons
Dodecagon	12-gons



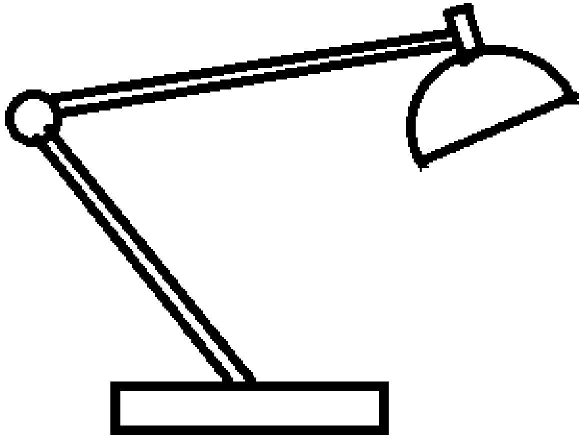
UNIT FORMS

1. A combination of simple objects that generate a more refined object.

2. It can be a composition of 2D shapes, such as the points, lines, triangles, rectangles, circles, ellipses, etc.



Unit form for a car



Unit form for a table lamp

EXERCISE 2

This activity will takes about ten minutes.

1. Illustrate the five Euclid's postulates.

2. All regular polygons are of type ______.

3. Name two shapes which satisfy convex hull.

10 MINUTES BREAK

…

MINI LECTURE 3
2D DRAWING FUNCTIONS

...

DRAWING TOOL

Function name	glBegin(), glEnd()
Purpose	It uses to define a group of vertices for one or more primitives.
Arguments or parameters	Drawing mode. Refer to next slide.
Return value	None

DRAWING MODE: POINT

GL_POINTS	It draws individual points.
glPointSize()	It sets the diameter of a point in pixels for GL_POINTS. Default value is one.

DRAWING MODE: LINE

GL_LINES	It draws lines for each specified pair of vertices.
GL_LINES_STRIP	It draws a line segment from the first vertex to the last vertex.
GL_LINE_LOOP	It draws an outline for the given vertices. Noted that it is the same as GL_LINES_STRIP with additional draw from last vertex to the first vertex.

DRAWING MODE: TRIANGLES

GL_TRIANGLES	It draws a triangle for each specified triplet of vertices.
GL_TRIANGLE_STRIP	It draws a set of triangles for each third vertex is defined.
GL_TRIANGLE_FAN	It draws a compute surface of triangles for each third vertex is defined.

DRAWING MODE: QUAD

GL_QUADS	It draws a quadrilateral plane for each specified quadro of vertices.
GL_QUAD_STRIP	It draws a mesh of quadrilateral plane for each fourth vertex is defined.

DRAWING MODE: POLYGON

GL_POLYGON	It draws a convex polygon.

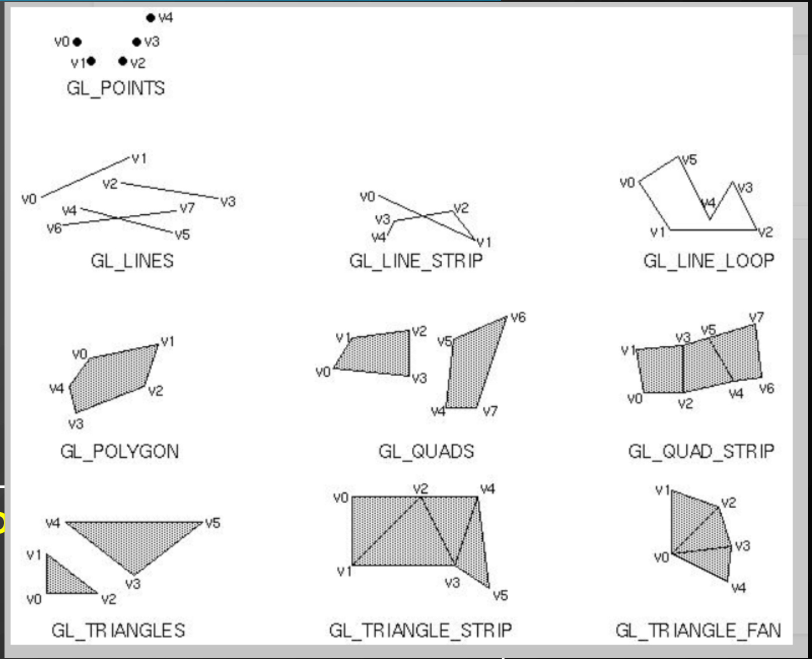
EXERCISE 3

This activity will takes about ten minutes.

1. Illustrate the ten drawing modes in OpenGL.

2. Save in 24 bits PNG file format.

GL_POINTS GL_LINES GL_LINES_STRIP GL_LINE_LOOP GL_TRIANGLES	GL_TRIANGLE_STRIP GL_TRIANGLE_FAN GL_QUADS GL_QUAD_STRIP GL_POLYGON



MINI LECTURE 4
MATRIX TRANSFORMATION IN 2D SPACE

…

2D TRANSLATION

1. To find an image point, (x', y', z') that translate an original point, (x, y) with a translation vector, (t_x, t_y).

2. Image point, $\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

2D ROTATION IN CLOCKWISE

1. To find an image point, (x', y', z') that rotates an original point, (x, y, 0) at certain degrees in clockwise (CW) direction with respect to point of origin.

2. Image point, $\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos(t) & \sin(t) & 0 \\ -\sin(t) & \cos(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$

2D ROTATION IN COUNTER-CLOCKWISE

1. To find an image point, (x’, y’, z’) that rotates an original point, (x, y, 0) at certain degrees in counter-clockwise (CCW) direction with respect to point of origin.

2. Image point, $\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos(t) & -\sin(t) & 0 \\ \sin(t) & \cos(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$

SCALE

1. To find an image point, (x', y', z') that scale an original point, (x, y, 0) on a scaling factor, (s_x, s_y).

2. Image point, $\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$

EXERCISE 4

This activity will takes about ten minutes.

Given a 2D original point at (15, 25) in the 2D space. Compute the corresponding image point with the following matrix transformations.

(a) Translate with a vector of (20, -47).

(b) Rotate clockwise (CW) at 30 degrees.

(c) Rotate counter-clockwise (CCW) at 90 degree

(d) Scale with a factor of (0.5 , 2.5).

1. $$\begin{bmatrix}1 & 0 & 20\\ 0 & 1 & -47\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}15\\ 25\\ 1\end{bmatrix} = \begin{bmatrix}15+0+20\\ 0+25-47\\ 0+0+1\end{bmatrix} = \begin{bmatrix}35\\ -22\\ 1\end{bmatrix}$$

2. $$\begin{bmatrix}\cos(30) & \sin(30) & 0\\ -\sin(30) & \cos(30) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}15\\ 25\\ 0\end{bmatrix} = \begin{bmatrix}15\cos(30) + 25\sin(30) + 0\\ -15\sin(30) + 25\cos(30) + 0\\ 0+0+0\end{bmatrix} =$$
$$\begin{bmatrix}15(0.866) + 25(0.5)\\ -15(0.5) + 25(0.866)\\ 0\end{bmatrix} = \begin{bmatrix}25.49\\ 14.15\\ 0\end{bmatrix}$$

3. $$\begin{bmatrix}\cos(90) & -\sin(90) & 0\\ \sin(90) & \cos(90) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}15\\ 25\\ 0\end{bmatrix} = \begin{bmatrix}0-25+0\\ 15+0+0\\ 0+0+0\end{bmatrix} = \begin{bmatrix}-25\\ 15\\ 0\end{bmatrix}$$

4. $$\begin{bmatrix}0.5 & 0 & 0\\ 0 & 2.5 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}15\\ 25\\ 0\end{bmatrix} = \begin{bmatrix}15(0.5)+0+0\\ 0+2.5(25)+0\\ 0+0+0\end{bmatrix} = \begin{bmatrix}7.5\\ 62.5\\ 0\end{bmatrix}$$

REFERENCES

Main reference:

Hajek, D. (2019). Introduction to Computer Graphics 2019 Edition. Independently Published.

Additional reference:

Marschner, S. and Shirley, P. (2021). Fundamentals of Computer Graphics, 5th Edn. CRC Press: Taylor's & Francis.