

Q.2 a. How Interactive graphics system is more useful than non-interactive graphics systems?

Ans: Interactive Graphics: Computer graphics includes the process and outcomes associated with using computer technology to convert created or collected data into visual representations. The computer graphics field is motivated by the general need for interactive graphical user interfaces that support mouse, windows and widget functions. Other sources of inspiration include digital media technologies, scientific visualization, virtual reality, arts and entertainment. Computer graphics encompasses scientific, art, and engineering functions. Mathematical relationships or Geometry define many of the components in a particular computer graphics "scene" or composition. Physics fundamentals are the basis for lighting a scene. The layout, color, texture and harmony of a particular composition are established with Art and Perception principles. Computer graphics hardware and software are rooted in the Engineering fields. The successful integration of these concepts allows for the effective implementation of interactive and three-dimensional (3D) computer graphics.

Interactive 3D graphics provides the capability to produce moving pictures or animation. This is especially helpful when exploring time varying phenomena such as weather changes in the atmosphere, the deflection of an airplane wing in flight, or telecommunications usage patterns. Interaction provides individual users the ability to control parameters like the speed of animations and the geometric relationship between the objects in a scene to one another.

Primitives are the basic geometrical shapes used to construct computer graphics scenes and the resulting final images. Each primitive has attributes like size, color, line and width. For two dimensions, examples of primitives include: a line, circle, ellipse, arc, text, polyline, polygon, and spline.

For 3D space, examples of primitives include a cylinder, sphere, cube and cone. 3D viewing is complicated by the fact that computer display devices are only 2D. Projections resolve this issue by transforming 3D objects into 2D displays.

b. Give very brief description of

(i) Input Devices:

Two possible ways to see input devices:

- As a physical device – keyboard, mouse, trackball, etc.
- As a logical device – from a programmer perspective – with specified functionality, in graphics more complex

- The separation of physical and logical levels enable us to make programs more flexible, independent from the actual physical device

Physical input devices:

- Pointing device – allows to indicate position & send signals/interrupts to the computer – relative/absolute positioning
- Keyboard device – almost physical keyboard – returns character codes to a program

(ii) GUI

A GUI consists of one or more *windows* and each window contains one or more *controls*. GUIs are "event driven" which means you react to events - like a button that is clicked. You spend most of your time idling and waiting for an event to happen - this is a little different to a normal script where you are in control of what happens and when! Think of it as waiting by the door for the postman - you sit there until a letter pops through the post-box and then you look at the letters and decide what to do with them - this is exactly how GUIs work - you wait for the postman to come to you.

Of course, you may choose to do other tasks while the GUI is active - for example you might use the GUI functions to create an elaborate progress box that you update while your script performs complex actions.

GUI Controls

All users will be familiar with controls - anything you click on or interact with in a window is a type of control. The types of controls that can be created with AutoIt are listed below - you will have used most of them in other Windows programs.

- | | |
|------------|--|
| • Label | A plain piece of text. |
| • Button | A simple button. |
| • Input | A single line control that you can enter text into. |
| • Edit | A multi-line control that you can enter text into. |
| • Checkbox | A box that can be "checked" or "unchecked". |
| • Radio | A set of circular buttons - only one can be active at nce. |
| • Combo | A list with a dropdown box. |
| • List | A list. |
| • Date | A date picker. |
| • Pic | A picture. |
| • Icon | An icon. |

- Progress A progress bar.
- Tab A group of controls that are contained in tabs.
- UpDown A control that can be added to input controls.
- Avi Display an AVI format clip.
- Menu A menu across the top of the window.
- ContextMenu A menu that appears when you right click inside the window.
- TreeView A control similar to the windows file-explorer.
- Slider A control similar to the windows sound volume control.
- ListView A control displaying columns information.
- ListViewItem A control displaying item in a listview control.
- Graphic A control displaying graphics drawn with GUICtrlSetGraphic.
- Dummy A dummy user control.

(iii) Tablet

Digitizer is a graphic-input device that uses a special pen or PUCK and a special surface that would track the pen position. The tablet surface had rows of microphones along its edges and it triangulated the pen position from the speed of sound delay between the pen and the microphones. These devices are rarely used today because of the advent of the Rand tablet. In the Rand tablet, a mesh of wire runs under the surface and the pen acts as an antenna for signals that are sent through the wires. Modern digitizer is less expensive and quieter than spark pens.

Tablets typically have one or more buttons on the pointing device. When an actual pen is used there is a switch in the tip that can be activated by pressing down on the tablet surface. There may also be a puck that has a set of buttons on it (as many as 16). These devices can be programmed to do separate actions on the down stroke of a button, while it is down, when it is released, and while it is up. The coordinate values are, of course, absolute within the range of the tablet surface.

Tablet positions are displayed on the screen by drawing a cursor at the current position. This cursor is a symbol such as an X, an arrow, or crosshairs, which consist of two perpendicular lines drawn from one edge of the screen to another. Although some displays handle cursor tracking in hardware, many require saving, setting, and restoring of the image at each advance of the cursor. For the cursor to be visible over any part of the screen, it must be a unique color. This color can simply be the inverse of the value that is normally there so that each pixel of the cursor will stand out. Beware, however, of color maps that use the same color in complementary positions; a more suitable cursor-color algorithm may be necessary. Proper cursor tracking of the tablet will free the user from the need to look at the tablet, because the hand-eye coordination keeps the user's attention on the screen.

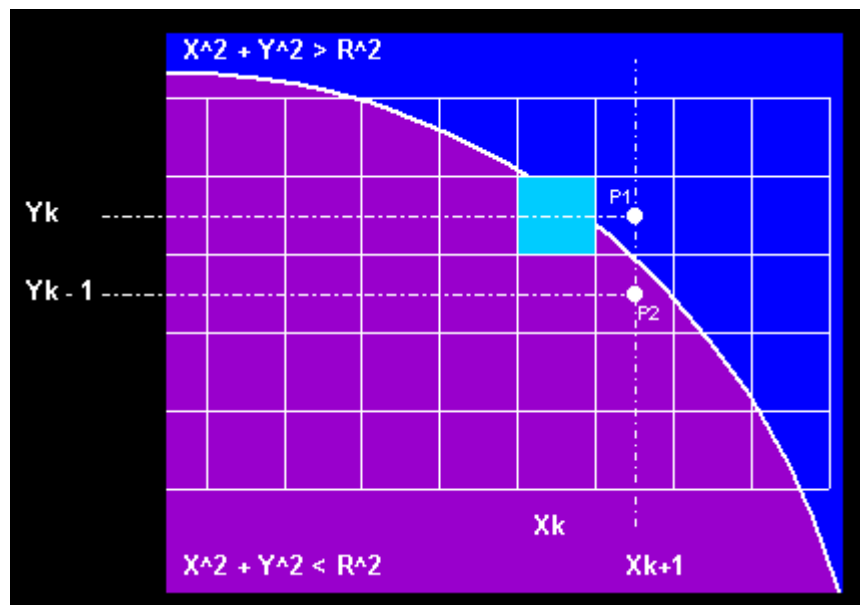
Q.3 a. Write an algorithm to draw a circle centred at (2, 3) and of radius 5.

Ans: We define a center (x0,y0) and a Radius. The mathematical circle is then the set of the points (x,y) such as:

$$(x-x_0)^2 + (y-y_0)^2 = \text{Radius}$$

We need to plot only one eighth of the circle, the other seven parts being symmetrical. Let's focus on the top right eighth, so.

Instead of computing an error along y as for the line, we compute for each of the possible choices (x_k+1, y_k) and (x_k+1, y_k-1) the error along the radius, i.e. (x²+y²-r²). If this function is positive, the point is outside the circle, if not, the point is inside the circle.



P1 is :

$$R1 = Y_k^2 + (X_k+1)^2 - \text{Radius}^2$$

The error for P2 is :

$$R2 = \text{Radius}^2 - (X_k+1)^2 - (Y_k-1)^2$$

After developing, we need to test the sign of the suite:

$$S_k = Y_k^2 + X_k^2 + 2 \cdot X_k - Y_k + 1 - \text{Radius}^2$$

The initial value in (0, Radius) is $S_0 = 1 - \text{Radius}$, hence:

- $S_0 = 1 - \text{Radius}$
- If $S(k) < 0$, then $Y(k+1) = Y(k)$ and $S(k+1) = S(k) + 2 * x + 3$
- If $S(k) \geq 0$, then $Y(k+1) = Y(k) - 1$ and $S(k+1) = S(k) + 2 * (x - y) + 5$

$x = 0$
 $y = 5$
 $s = 1 - 5$

DO WHILE $x < y$
 IF $s < 0$ THEN
 $s = s + 2 * x + 3$

ELSE

$s = s + 2 * (x - y) + 5$
 $y = y - 1$
 END IF

$x = x + 1$
 Pset (x, y)
 LOOP

Then translate the point with translation matrix $T(2, 3)$.

Q.4. a. Find the new points when the point (3, 4) is rotated by an angle of 45° anticlockwise and then translating the resultant point by 2 unit along x-axis and 5 units along y-axis.

Ans: Apply the transformation $T(2, 5) * R(45^\circ) * [3, 4, 1]^T$

b. Explain homogeneous co-ordinate system. Why is it required to be considered while transforming an object from one reference frame to other reference frame?

Ans: One of the many purposes of using homogeneous coordinates is to capture the concept of infinity. In the Euclidean coordinate system, infinity is something that does not exist. Let us consider two real numbers, a and w , and compute the value of a/w . Let us hold the value of a fixed and vary the value of w . As w getting smaller, the value of a/w is getting larger. If w approaches zero, a/w approaches to infinity! Thus, to capture the concept of infinity, we use two numbers a and w to represent a value v , $v = a/w$. If w is not zero, the value is exactly a/w . Otherwise, we identify the infinite value with $(a, 0)$.

Therefore, the concept of infinity can be represented with a number pair like (a, w) or as a quotient a/w .

For example, suppose we have a line $Ax + By + w = 0$. Dividing the equation by w and assuming x and y having w a factor, it yields $A(x/w) + B(y/w) + 1 = 0$. or

$$Ax' + By' + 1 = 0.$$

The coordinate (x, y) is then represented as $(x', y', 1)$ or in more general $(x, y, 1)$. This works for three-dimension as well. One can replace a point (x, y, z) with $(x, y, z, 1)$. This is of great help in expressing transformation as product of matrices.

Q.5 a. Write the Sutherland Hodgman Clipping algorithm and clip a line segment between points $(4, 3)$ to $(5, 7)$ so that it fits into view port with left bottom at $(4,4)$ and right top at $(5, 7)$.

Ans: Sutherland-Hodgman Clipping

- Polygon Clipping (Sutherland-Hodgman):
 - Window must be a convex polygon
 - Polygon to be clipped can be convex or not
- **Approach:**
 - Polygon to be clipped is given as a sequence of vertices
 - Each polygon edge is a pair
 - Don't forget wrap around; is also an edge
 - Process all polygon edges in succession against a window edge
 - Polygon in - polygon out
 - Repeat on resulting polygon with next sequential window edge
 - Sutherland-Hodgman algorithm deals with concave polygons efficiently
 - Built upon efficient line segment clipping

There are four cases to consider :

- $s = v_j$ is the polygon edge starting vertex
- $p = v_{j+1}$ is the polygon edge ending vertex
- i is a polygon-edge/window-edge intersection point
- w_j is the next polygon vertex to be output

The end points of the clipped line are $(4, 4)$ and $(5, 7)$. Candidate should show the steps to arrive at the solution.

b. Determine a transformation matrix to view a digital object of extent $(1, 1)$ to $(20, 20)$ on display area of width = 15 and height = 10 starting from $(0, 0)$. Unit of measurement is same for both object and display area.

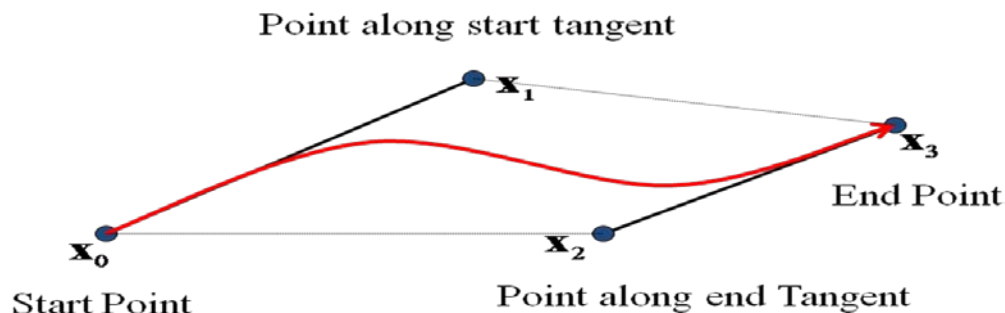
Ans: Use the transformation matrix to view x-z plane along y-axis.

Q6 b. Differentiate between spline and curve. Define a cubic Bezier curve and give the formula to compute its control points.

Ans: A spline function is a piecewise polynomial function joined together with certain continuity conditions satisfied whereas in a curve no such condition is required. Each piece in a spline is a curve of a particular type giving it a name of that type of spline e.g. linear spline (each piece is a line), quadratic spline (each piece is a quadratic curve) and so on.

Bezier curves

- Different choices of basis functions give different curves
 - Choice of basis determines how the control points influence the curve
 - In Hermite case, two control points define endpoints, and two more define parametric derivatives
- For Bezier curves, two control points define endpoints, and two control the tangents at the endpoints in a geometric way



- The user supplies d control points, p_i
- Write the curve as:

$$\mathbf{x}(t) = \sum_{i=0}^d \mathbf{p}_i B_i^d(t)$$

$$B_i^d(t) = \binom{d}{i} t^i (1-t)^{d-i}$$

- The functions B_i^d are the *Bernstein polynomials* of degree d
- This equation can be written as a matrix equation also
- There is a matrix to take Hermite control points to Bezier control points

Q.7 a. Write the z-buffer algorithm to detect visible surface. If the colour of the projection surface is same as that of visible surface, then how the surface will be made visible?

Ans: Z-Buffer Algorithm:

- We know how to scan-convert polygons onto the screen.
- What we want to do now is, only overwrite a pixel if the polygon element has a larger Z value than what is in the frame-buffer currently.
- We will create a Z-buffer to hold the depth values of each pixel in the frame-buffer.
- The general Algorithm:
- Initialize frame-buffer and Z-buffer
-
- for each polygon P
- for each pixel in poly's projection: x,y
- pz = the Z value of P at this pixel
- polycolor[x][y] =
- color of P at this pixel
- if Z-buffer[x][y] <= pz
- Z-buffer[x][y] = pz
- FB[x][y] = polycolor[x][y]
- end for each pixel
- end for each polygon

Z values calculated incrementally from the plane equation (dz/dx , dz/dy).

- Number of comparisons independent of number of polygons.
 - Works fine for patches & other primitives.
 - Requires a large amount of memory (1280*1024*32bits typically > 5Mb)
2. Often implemented in hardware.

Q.8 a. What is animation? What are the different methods to produce real time animation?

Ans: Real Time Animation is the rapid display of a sequence of images of 2-D or 3-D artwork or model positions in order to create an illusion of movement. It is an optical illusion of motion due to the phenomenon of persistence of vision, and can be created and demonstrated in a number of ways. The most common method of presenting animation is as a motion picture or video program, although several other forms of presenting animation also exist.

Computer animation (or CGI animation) is the art of creating moving images with the use of computers. It is a subfield of computer graphics and animation. Increasingly it is created by means of 3D computer graphics, though 2D computer graphics are still widely used for stylistic, low bandwidth, and faster real-time rendering needs. Sometimes the target of the animation is the computer itself, but sometimes the target is another medium, such as film. It is also referred to as CGI (computer-generated imagery or computer-generated imaging), especially when used in films.

To create the illusion of movement, an image is displayed on the computer screen and repeatedly replaced by a new image that is similar to the previous image, but advanced slightly in the time domain (usually at a rate of 24 or 30 frames/second). This technique is identical to how the illusion of movement is achieved with television and motion pictures.

Computer animation is essentially a digital successor to the art of stop motion animation of 3D models and frame-by-frame animation of 2D illustrations. For 3D animations, objects (models) are built on the computer monitor (modeled) and 3D figures are rigged with a virtual skeleton. For 2D figure animations, separate objects (illustrations) and separate transparent layers are used, with or without a virtual skeleton. Then the limbs, eyes, mouth, clothes, etc. of the figure are moved by the animator on key frames. The differences in appearance between key frames are automatically calculated by the computer in a process known as tweening or morphing. Finally, the animation is rendered.

For 3D animations, all frames must be rendered after modeling is complete. For 2D vector animations, the rendering process is the key frame illustration process, while tweened frames are rendered as needed. For pre-recorded presentations, the rendered frames are transferred to a different format or medium such as film or digital video. The frames may also be rendered in real time as they are presented to the end-user audience. Low bandwidth animations transmitted via the internet (e.g. 2D Flash, X3D) often use software on the end-users computer to render in real time as an alternative to streaming or pre-loaded high bandwidth animations.

There are several methods for generating the Avar values to obtain realistic motion. Traditionally, animators manipulate the Avars directly. Rather than set Avars for every frame, they usually set Avars at strategic points (frames) in time and let the computer interpolate or 'tween' between them, a process called keyframing. Keyframing puts control in the hands of the animator, and has roots in hand-drawn traditional animation.

In contrast, a newer method called motion capture makes use of live action. When computer animation is driven by motion capture, a real performer acts out the scene as if they were the character to be animated. His or her motion is recorded to a computer using video cameras and markers, and that performance is then applied to the animated character.

- b.** Define the term morphing and explain basic restrictions imposed on the number of features required to be maintained on two consecutive frames.

Ans: Morphing is a special effect in motion pictures and animations that changes (or morphs) one image into another through a seamless transition. Most often it is used to depict one person turning into another through technological means or as part of a fantasy or surreal sequence. Traditionally such a depiction would be achieved through cross-fading techniques on film.

A key frame in animation is a drawing that defines the starting and ending points of any smooth transition. They are called "frames" because their position in time is measured in frames on a strip of film. A sequence of keyframes defines *which* movement the spectator will see, whereas the position of the keyframes on the film, video or animation defines the timing of the movement. Because only two or three keyframes over the span of a second does not create the illusion of movement, the remaining frames are filled with inbetweens.

In computer animation this workflow is basically the same. The animator creates the important frames of a sequence, then the software fills in the gap. For example, in Adobe Systems Flash, the animator can specify, in keyframes, the starting and ending position of an object, such as a circle. Flash smoothly translates the object from the starting point to the ending point. This is called tweening. The animator can correct the result at any point in time, shifting keyframes back and forth to improve the timing and dynamics of a movement, or change an 'in between' into an additional keyframe to further refine the movement

Use of key frames as a means to change parameters

In software packages that support animation, especially 3D graphics packages, there are many parameters that can be changed for any one object. One example of such an object is a light. (In 3D graphics, lights function similarly to real world lights - They cause illumination, cast shadows, and create specular highlights.) Lights have many parameters including light intensity, beam size, light colour, and the texture cast by the light. Supposing that an animator wants the beam size of the light to change smoothly from one value to another within a pre-defined period of time, that could be achieved by using keyframes. At the start of the animation, a beam size value is set. Another value is set for the end of the animation. Thus, the software program automatically interpolates the two values, creating a smooth transition

- Q.9 b.** Briefly explain use of audio, visual and text elements in multimedia. How are they stored?

Ans: In computer science, the presentation of information using the combination of text, sound, pictures, animation, and video. Common multimedia computer applications include games, learning software, and reference materials, such as this encyclopedia. Most multimedia applications include predefined associations, known as *hyperlinks* that enable users to switch between media elements and topics.

Thoughtfully presented multimedia can enhance the scope of presentation in ways that are similar to the roving associations made by the human mind. Connectivity provided by hyperlinks transforms multimedia from static presentations with pictures and sound into an endlessly varying and informative interactive experience.

VISUAL ELEMENTS

The larger, sharper, and more colorful an image is, the harder it is to present and manipulate on a computer screen. Photographs, drawings, and other still images must be changed into a format that the computer can manipulate and display. Such formats include bit-mapped graphics and vector graphics.

Bit-mapped graphics store, manipulate, and represent images as rows and columns of tiny dots. In a bit-mapped graphic, each dot has a precise location described by its row and column, much like each house in a city has a precise address. Some of the most common bit-mapped graphics formats are called Graphical Interchange Format (GIF), Tagged Image File Format (TIFF), and Windows Bitmap (BMP).

Video files can be quite large, so they are usually reduced in size using *compression* technique. Common video compression formats are Audio Video Interleave (AVI), Quicktime, and Motion Picture Experts Group (MPEG or MPEG2). These formats can shrink video files by as much as 95 percent, but they introduce varying degrees of fuzziness in the images.

Animation can also be included in multimedia applications to add motion to images. Animations are particularly useful to simulate real-world situations, such as the flight of a jet airplane. Animation can also enhance existing graphics and video elements adding special effects such as morphing, the blending of one image seamlessly into another.

Sound Elements

Sound, like visual elements, must be recorded and formatted so the computer can understand and use it in presentations. Two common types of audio format are Waveform (WAV) and Musical Instrument Digital Interface (MIDI). WAV files store actual sounds, much as music CDs and tapes do. WAV files can be large and may require compression. MIDI files do not store the actual sounds, but rather instructions that enable devices called synthesizers to reproduce the sounds or music. MIDI files are much smaller than WAV files, but the quality of the sound reproduction is not nearly as good.

TEXTBOOK: Computer Graphics, Amarendra N.Sinha, Arun D Udai, TMH 2008