Animation and Computer Graphics Realism

5

Chapter in a Nutshell

- **Ray Tracing Methods :** Ray tracing is an extension of this basic idea. Instead of merely looking for the visible surface for each pixel, We continue to bounce the ray around the scene collecting intensity contributions. This provides a simple and powerful rendering technique for obtaining global reflection and transmission effects.

Previous Years Questions

Part-A

Q.1 *What is computer graphics realism?*

Ans. Computer Graphics Realism : The creation of realistic picture in computer graphics is known as realism. It is important in fields such as simulation, design, entertainments, advertising, research, education, command, and control.

Q.2 *How realistic pictures are created in computer graphics?*

Ans. To create a realistic picture, it must be process the scene or picture through viewing coordinate transformations and projection that transform 3D viewing coordinates onto 2D device coordinates.

Q.3 *What is Fractals?*

Ans. Fractal : A fractal is an image or a geometric object with self-similar, infinite properties produced by recursive or iterative algorithms.

Q.4 *What is Koch Curve?*

Ans. Koch Curve : The Koch curve is a curve that is generated by a simple geometric procedure which can be iterated an infinite number of times by dividing a straight line segment into three equal parts and substituting the intermediate part with two segment of the same length.

Q.5 *What is turtle graphics program?*

Ans. Turtle Graphics Program : The turtle program is a Robert that can move in two dimensions and it has a pencil for drawing. The turtle is defined by the following parameters :

Position of the turtle(x, y)

Heading of the turtle( ) the angle from the x-axis.

Part-B

Q.6 *Write short note on simple recursive ray tracing without antialiasing.* ***[R.T.U. Dec. 2013]***

Ans. Recursive Ray Tracing

1. Ray tracing handles global reflections and refractions (transmissions), hidden surface removal, shadows, and other special effects
2. We want to consider
   - How to build a ray tracer
   - How to make it efficient
   - How to improve the image quality with special effects
3. Whitted presented the first general recursive ray tracing model in 1980 (others had used some of the ideas before)
4. Ray tracing follows specularly reflected and refracted rays through a scene
5. The rays are infinitely thin, so aliasing is a problem
6. Ray tracing can be used as a basic technique for volume rendering
7. Ray tracing is recursive; secondary rays are followed recursively from primary rays
8. The basic algorithm fires an *eye ray* through the center of a pixel to find the intersection with the closest object
9. A (specular) *reflected ray* is followed from the intersection
10. A (specular) *refracted ray* is followed from the intersection
11. To calculate shadows, fire a *shadow ray* (or *shadow feeler*) from the intersection point to each light source
12. Reflection and refraction are handled by *secondary rays* from the point of intersection
13. Note that is first an eye ray, then for secondary rays it can be a reflected or refracted ray; its role changes as rays are cast from ray-object intersection points.

Q.7 *Explain the concept of motion specification.*

Ans. Motion Specifications : There are several ways in which the motions of objects can be specified in an animation system. We can define motions in very explicit terms, or we can use more abstract or more general approaches.

Direct Motion Specification : The most straightforward method for defining a motion sequence is direct specification of the motion-parameters. Here, we explicitly give the rotation angles and translation vectors. Then the geometric transformation matrices are applied to transform coordinate positions. Alternatively, we could use an approximating equation to specify certain kinds of motions.

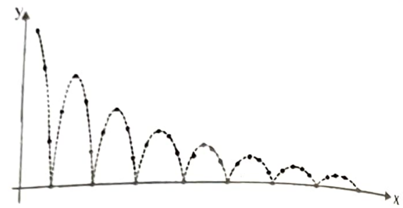

Fig. : Approximating the motion of a bouncing ball with a damped sine function

We can approximate the path of a bouncing ball, for instance, with a damped, rectified, sine curve (Fig.).

$$y(x) = A\,|\sin(\omega x + \theta_0)|e^{kx}$$

where A is the initial amplitude, ω is the angular frequency, θ_0 is the phase angle, and k is the damping constant. These methods can be used for simple user-programmed animation sequences.

Goal-Directed Systems : At the opposite extreme, 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. The input directives are then interpreted in terms of component motions that will accomplish the selected task. Human motions, for instance, can be defined as a hierarchical structure of sub-motions for the torso, limbs, and so forth.

Kinematics and Dynamics : We can also construct animation sequences using kinematic or dynamic descriptions. With a kinematic description, we specify the animation by giving motion parameters (position, velocity, and acceleration) without reference to the forces that cause the motion. For constant velocity (zero acceleration), we designate the motions of rigid bodies in a scene by giving an initial positions and velocity vector for each object. As an example, if a velocity is specified as (3, 0, -4) km/sec, then this vector gives the direction for the straight-line motion path and the speed (magnitude of velocity) is 5 km/sec. If we also specify accelerations (rate of change of velocity), we can generate speed-ups, slow-downs, and curved motion paths. Kinematic specifications of a motion can also be given by simply describing the motion path. This is often done using spline curves.

Q.8 *What is tiling the plane? Write its types.*

Ans. Tiling the Plane

- Use one or more geometric shapes
- Tessellation (without gaps) of flat surface
- Shape repeated
- Moving infinity
- Covering entire plane
- Used arts, mosaics, wall papers, tiled floor

Types of Tiling

- Monohedral tiling
- Cairo tiling
- Drawing tiling
- Reptiles

Monohedral Tiling : This is based on single polygon

Cairo Tiling

- Four pentagon fit together to form hexagon
- Used to tile the plane
- Many street in cairo, Egypt in this pattern

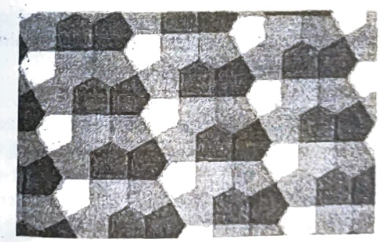

Fig. : Cairo Tiling

Drawing Tiling

- Large window setup
- Tiles grouped together into single figure
- Single figure drawn again and again
- Non periodic figure include
- Small to large and large to small

Reptiles

- Non periodic tiling
- Based on square, equilateral triangle

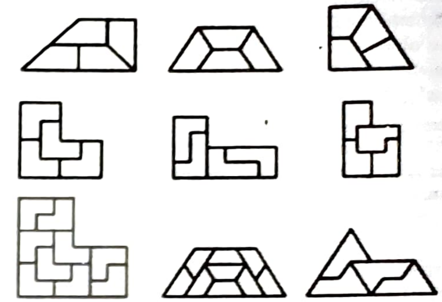

Fig. : Reptiles

Q.9 *Write short note on raster animations.*

Ans. Raster Animations : On raster systems, we can generate real-time animation in limited applications using raster operations. A simple method for translation in the xy plane is to transfer a rectangular block of pixel values from one location to another. Two-dimensional rotations in multiples of 90° are also simple to perform, although we can rotate rectangular blocks of pixels through arbitrary angles using antialiasing procedures. To rotate a block of pixels, we need to determine the percent of area coverage for those pixels that overlap the rotated block. Sequences of raster operations can be executed to produce real-time animation of either two-dimensional or three-dimensional objects, as long as we restrict the animation to motions in the projection plane. Then no viewing or visible-surface algorithms need be invoked.

We can also animate objects along two-dimensional motion paths using the color-table transformations. Here we predefine the object at successive positions along the motion path, and set the successive blocks of pixel values to color-table entries.

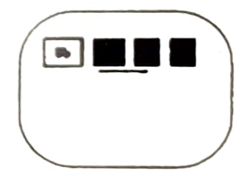

Fig. : Real time raster color table animation

We set the pixels at the first position of the objects to "on" values and we set the pixels at the other object positions to the background color. The animation is then accomplished by changing the color-table values so that the object is "on" at successively positions along the animation path as the preceding position is set to the background intensity (Fig.).

Q.10 ***Write note on Random fractals?***

Ans. Random Fractals : Fractal is the term associated with randomly generated curves and surfaces that exhibit a degree of self-similarity. These curves are used to provide naturalistic shapes for representing objects such as coastlines, rugged mountains, grass and fire.

Fractalizing a Segment : The simplest random fractal is formed by recursively roughening or fractalizing a line segment. At each step, each line segment is replaced with a random elbow.

This process applied to the line segment S having endpoints A and B. S is replaced by the two segments from A to C and from C to B. For a fractal curve, point C is randomly chosen along the perpendicular bisector L of S. the elbow lies randomly on one or the other side of the parent segment AB.

Three stages are required in the fractalization of a segment. In the first stage, the midpoint of AB is perturbed to form point C. In the second stage, each of the two segment has its midpoints perturbed to form points D and E. In the third and final stage, the new points F are added.

PART-C

Q.11 ***What is ray tracing? How can you render polygon surface using Gourand shading?***

[R.T.U. 2018, R.U. 2007]

OR

What is ray tracing? Explain basic ray tracing algorithm. ***[R.T.U. 2017]***

OR

Write short note on Ray Tracing algorithm.

[R.T.U. 2015, 2013, R.U. 2006, 2002]

OR

Explain Ray Tracing. ***[R.U. 2010]***

Ans. Ray Tracing : In computer graphics, ray tracing is a technique for generating an image by tracing the path of light through pixels in an plane and simulating the effects of its encounters with virtual objects. The technique is capable of producing a very high degree of visual realism, usually higher than that of typical scan line rendering methods, but at a greater computational cost. This makes ray tracing best suited for applications where the image can be rendered slowly ahead of time, such as in still images and film and television visual effects, and more poorly suited for real-time applications like video games where speed is critical. Ray tracing is capable of simulating a wide variety of optical effects, such as reflection and refraction, scattering, and dispersion phenomena (such as chromatic aberration).

Optical ray tracing describes a method for producing visual images constructed in 3D computer graphics environments, with more photorealism than either ray casting or scan line rendering techniques. It works by tracing a path from an imaginary eye through each pixel in a virtual screen, and calculating the color of the object visible through it. Scenes in ray tracing are described mathematically by a programmer or by a visual artist (typically using intermediary tools). Scenes may also incorporate data from images and models captured by means such as digital photography. Typically, each ray must be tested for intersection with some subset of all the objects in the scene. Once the nearest object has been identified, the algorithm will estimate the incoming light at the point of intersection, examine the material properties of the object, and combine this information to calculate the final color of the pixel. Certain illumination algorithms and reflective or translucent materials may require more rays to be re-cast into the scene.

Ray Tracing Algorithm

(1) For each pixel ray, we test each surface in the scene to determine if it is intersected by the ray.

(2) If a surface is intersected, we calculated the distance from the pixel to the surface intersection point.

(3) The smallest calculated intersection distance identifies the visible surface for that pixel.

(4) Reflect the ray off the visible surface along a specular path (angle of reflection equal to angle of incidence).

(5) If the surface is transparent, we also send a ray through the surface in the refraction direction. Reflection and refraction ray are referred to as secondary rays.

(6) This procedure is repeated for each secondary ray.

(7) As the rays from a pixel ricochet through the scene, each successively intersected surface is added to a binary ray tracing tree.

(8) Left branches in the tree are used to represent reflection path and right branches represent transmission paths.

(9) Maximum depth of the ray tracing tree can be set as a user option, or it can be determined by the amount of storage available. A path in the tree is then terminated if it reaches the present maximum or if the ray strikes a light source.

(10) The intensity assigned to a pixel is then determined by accumulating the intensity contributions starting at the bottom (terminal nodes) of its ray tracing tree.

Pixel intensity is then the sum of the attenuated intensities at the root node of the ray tree.

Gourand Shading : Each polygon surface rendered with Gourand shading by performing the following calculations :

(1) Determine the average unit normal vector at each polygon vertex.

(2) Apply an illumination model to each vertex to calculate the vertex intensity.

(3) Linearly interpolate the vertex intensities over the surface of the polygon.

For any vertex position V, obtain the unit vertex normal with the calculation :

$$N_v = \frac{\sum_{k=1}^{n} N_k}{\left|\sum_{k=1}^{n} N_k\right|}$$

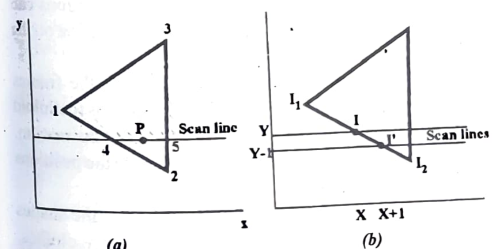

Fig. : (a) Linear Interpolation of intensities (b) Incremental interpolation of intensity values along a polygon edge for successive scan lines

The next step is to interpolate intensities along the polygon edges. For each scan line, the intensity at the intersection of the scan with the polygon edge is linearly interpolated from the intensities at the edge end-points.

$$I_4 = \frac{y_4 - y_2}{y_1 - y_1} I_1 + \frac{y_1 - y_4}{y_1 - y_2} I_2$$

Similarly, intensity at the right intersection of this scan line (point 5) is interpolated from intensity values at vertices 2 and 3.

An interior point 'P' is interpolated from the bounding intensities at point 4 and 5 as :

$$I_p = \frac{x_5 - x_p}{x_5 - x_4} I_4 + \frac{x_p - x_4}{x_5 - x_4} I_5$$

Incremental calculations are used to obtain successive edge intensity values between scan lines and to obtain successive intensities along a scan line.

If the intensity at edge position (x,y) is interpolated as :

$$I = \frac{y - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y}{y_1 - y_2} I_1$$

Then, intensity along this edge for the next scan line, y–1 is :

$$I' = I + \frac{I_2 - I_1}{y_1 - y_2}$$

Q.12 ***Write the strategy for design of animation sequences. Explain animation function also.***

Ans. Design of Animation Sequences : In general, an animation sequence is designed with the following steps:

- Storyboard layout
- Object definitions
- Key-frame specifications
- Generation of in-between frames

This standard approach for animated cartoons is applied to other animation applications as well, although there are many special applications that do not follow this sequence. Real-time computer animations produced by flight simulators, for instance, display motion sequences in response to settings on the aircraft controls. And visualization applications are generated by the solutions of the numerical models. For frame-by-frame animation, each frame of the scene is separately generated and stored. Later, the frames can be recorded on film or they can be consecutively displayed in "real-time playback" mode.

The storyboard is an outline of the action. It defines the motion sequence as a set of basic events that are to take place. Depending on the type of animation to be produced, the storyboard could consist of a set of rough sketches or it could be a list of the basic ideas for the motion.

An object definition is given for each participant in the action. Objects can be defined in terms of basic shapes, such as polygons or splines. In addition, the associated movements for each object are specified along with the shape.

A key frame is a detailed drawing of the scene at a certain time in the animation sequence. Within each key frame, each object is positioned according to the time for that frame. Some key frames are chosen at extreme positions in the action; others are spaced so that the time interval between key frames is not too great. More key frames are specified for intricate motions than for simple, slowly varing motions.

In betweens are the intermediate frames between the key frames. The number of in-betweens needed is determined by the media to be used to display the animation. Film requires 24 frames per second, and graphics terminals are refreshed at the rate of 30 to 60 frames per second. Typically, time intervals for the motion are set up so that there are from three to five in-betweens for each pair of key frames can be duplicated. For a 1-minute film sequence with no duplication, we would need 1440 frames. With five in-between for each pair of key frames, we would need 288 key frames. If the motion is not too complicated, we could space the key frames a little further apart.

There are several other tasks that may be required, depending on the application. They include motion verification, editing, and production and synchronization of a soundtrack. Many of the functions needed to produce general animations are now computer-generated. Fig. show example of computer-generated frames for animation sequences.

Fig. : Computer generated frame for animation sequence

General Computer Animation Functions : Some steps in the development of an animation sequence are well-suited to computer solution. These include object manipulations and rendering, camera motions, and the generations of in-betweens. Animation packages, such as Wave-front, for example, provide special functions for designing the animation and processing individual objects.

One function available in animation packages is provided to store and manage the object database. Object shapes and associated parameters are stored and updated in the database. Other object functions include those for motion generation and those for object rendering. Motions can be generated according to specified constraints using two-dimensional or three-dimensional transformations. Standard functions can then be applied to identify visible surfaces and apply the rendering algorithms.

Another typical function simulates camera movements. Standard motions are zooming, panning, and tilting. Finally, given the specifications for the key frames, the in-betweens can be automatically generated.

Q.13 *(a) Define key-frame systems and also explain morphing process.*

(b) Explain different types of fractals and how to construct fractals and use of fractals in computer graphics?

Ans. (a) Key-Frame Systems : We generate each set of in-betweens from the specification of two (or more) key frames. Motion paths can be given with a kinematic description as a set of spline curves, or the motions can be physically based by specifying the forces acting on the objects to be animated.

For complex scenes, we can separate the frames into individual components or objects called cels (celluloid transparencies), an acronym from cartoon animation. Given the animation paths, we can interpolate the positions of individual objects between any two times.

With complex, object transformations, the shapes of objects may change over time. Examples are clothes, facial features, magnified detail, evolving shapes, exploding or disintegrating objects, and transforming one object into another object. If all surfaces are described with polygon meshes, then the number of edges per polygon can change from one frame to the next. Thus, the total number of line segments can be different in different frames.

Morphing : Transformation of object shapes from one

form to another is called morphing, which is shortened form of metamorphosis. Morphing methods can be applied to any motion or transition involving a change in shape.

Given two key frames for an object transformation, we first adjust the object specification in one of the frames so that the number of polygon edges (or the number of vertices) is the same for the two frames. This preprocessing step is illustrated in fig. 1. A straight-line segment in key frame k is transformed into two line segments in key frame k + 1. Since key frame k + 1 has an extra vertex, we add a vertex between vertices 1 and 2 in key frame k to balance the number of vertices (and edges) in the two key frames. Using linear interpolation to generate the in-betweens, we transition the added vertex in key frame k into vertex 3 along the straight-line path shown in Fig. 2. An example of a triangle linearly expanding into a quadrilateral is given in Fig. 3.

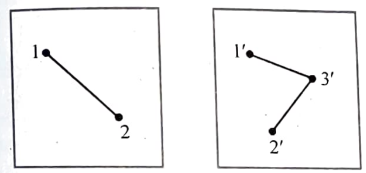

Fig. 1 : An edge with vertex position 1 and 2 in frame k evolves two connected edges in key frame k+1

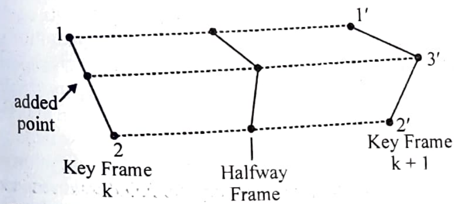

Fig. 2 : Lineat interpolation for transforming a line segment in key frame k into two connected line segments in key frame k+1

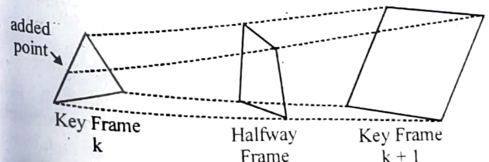

Fig. 3 : Linear interpolation for transforming a triangle into a quadrilateral

We can state general preprocessing rules for equalizing key frames in terms of either the number of edges or the number of vertices to be added to a key frame. Suppose we equalize the edge count, and parameters L_k and L_{k+1} denote the number of line segments in two consecutive frames. We then define

$$L_{max} = \max (L_k, L_{k+1}),$$
$$L_{min} = \min (L_k, L_{k-1})$$

Then the preprocessing is accomplished by

1. dividing N_e edges of keyframe$_{min}$ into $N_s + 1$ sections
2. dividing the remaining lines of keyframe$_{min}$ into N_s sections

As an example, if $L_k = 15$ and $L_{k+1} = 11$, we would divide 4 lines of keyframe$_{k+1}$ into 2 sections each. The remaining lines of keyframe$_{k+1}$ are left intact.

If we equalize the vertex count, we can use parameters V_k and V_{k+1} to denote the number of vertices in the two consecutive frames. In this case, we define

$$V_{max} = \max(V_k, V_{k+1}),$$
$$V_{min} = \min(V_k, V_{k+1}) \quad \text{... (i)}$$

and $$N_{ls} = (V_{max} - 1) \bmod (V_{min} - 1)$$

$$N_p = \text{int}\left(\frac{V_{max} - 1}{V_{min} - 1}\right) \quad \text{... (ii)}$$

Preprocessing using vertex count is performed by

1. adding N_p points to N_{ls} line sections of keyframe$_{min}$
2. adding $N_p - 1$ points to the remaining edges of keyframe$_{min}$

For the triangle-to-quadrilateral example, $V_k = 3$ and $V_{k+1} = 4$. Both N_{ls} and N_p are 1, so we would add one point to one edge of keyframe$_k$. No points would be added to the remaining lines of keyframe$_{k+1}$.

Ans. (b) Fractals can appear in anything. They are not only computerized but also appear in nature such as lightening ferns, shells, rivers, clouds and many more. These are a few of the different types of fractals.

(i) Mandelbrot Sets : The Mandelbrot set is the subset of the complex plane consisting of those parameters for which the Julia set of is connected. The Mandelbrot set can also be defined as the set of parameters for which the set has a finite upper bound.

Fig. 1 : Mandelbrot sets

(ii) Julia Sets : Julia Sets are produced with the same formula as the Mandelbrot set, but the starting values are different. That is c is more constant, and z0 is the starting point on the plane. From this definition, an infinite number of Julia Sets; one for each value of c. In fact, there is a Julia Set that corresponds to each point on complex plane. There is an interesting relationship between the Mandelbrot Set and the Julia Sets. In a way, you can think of the Mandelbrot Set as an index for the Julia Sets. For values of c that are inside the Mandelbrot Set, you will get connected Julia Sets. That is all the black regions are connected. Conversely, those values of c outside the Mandelbrot Set, you get unconnected sets.

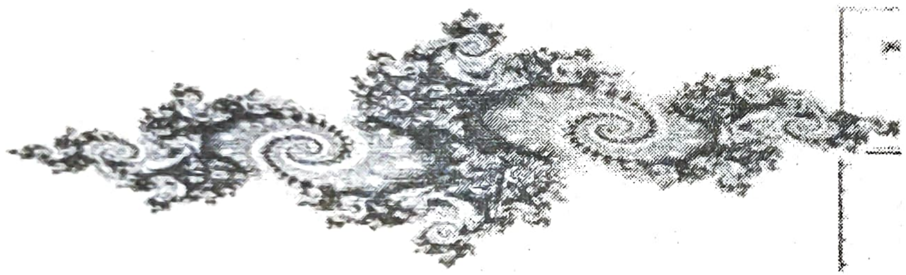

Fig. 2 : Julia sets

(iii) Leinian Group Fractals : Kleinian Group fractals are fractals based on 2 pairs of Mobius transformations and allow you to produce Quasifuchsian, Single Cusp, and Double Cusp.

Fig. 3 : Leinian Group Fractals

(iv) Newton Method Fractals : Isaac Newton discovered what we now call Newton's method around 1670. Although Newton's method is an old application of calculus, it was discovered relatively recently that extending it to the complex plane leads to a very interesting fractal pattern.

(v) Quaternion 3D Fractals : Quaternion Julia fractals are created by the same principle as the more traditional Julia set except that it uses 4 dimensional complex numbers instead of 2 dimensional complex numbers.

❑❑❑