Q. 1 (A) DISCUSS SOME OF THE APPLICATION AREAS IN WHICH COMPUTER GRAPHICS IS USED. WRITE THE TWO METHODS USED FOR DISPLAYING THE COLOR PICTURES ON CRT MONITORS.

ANS. Application of computer graphics

- User interface: - Visual object which we observe on screen which communicates with user is one of the most useful applications of the computer graphics.
- Plotting of graphics and chart in industry, business, government and educational organizations drawing like bars, pie-charts, histogram's are very useful for quick and good decision making.
- Office automation and desktop publishing: - It is used for creation and dissemination of information. It is used in in-house creation and printing of documents which contains text, tables, graphs and other forms of drawn or scanned images or picture.
- Computer aided drafting and design: - It uses graphics to design components and system such as automobile bodies structures of building etc.
- Simulation and animation: - Use of graphics in simulation makes mathematic models and mechanical systems more realistic and easy to study.
- Art and commerce: - There are many tools provided by graphics which allows used to make their picture animated and attracted which are used in advertising.
- Process control: - Now a day's automation is used which is graphically displayed on the screen.
- Cartography: - Computer graphics is also used to represent geographic maps, weather maps, oceanographic charts etc.
- Education and training: - Computer graphics can be used to generate models of physical, financial and economic systems. These models can be used as educational aids.
- Image processing: - It is used to process image by changing property of the image.

The CRT Monitor display by using a combination of phosphors. The phosphors are different colors. There are two popular approaches for producing color displays with a CRT are:

1. Beam Penetration Method

The Beam-Penetration method has been used with random-scan monitors. In this method, the CRT screen is coated with two layers of phosphor, red and green and the displayed color depends on how far the electron beam penetrates the phosphor layers. This method produces four colors only, red, green, orange and yellow. A beam of slow electrons excites the outer red layer only; hence screen shows red color only. A beam of high-speed electrons excites the inner green layer. Thus screen shows a green color.

2. Shadow-Mask Method

Shadow Mask Method is commonly used in Raster-Scan System because they produce a much wider range of colors than the beam-penetration method.

It is used in the majority of color TV sets and monitors.

Construction: A shadow mask CRT has 3 phosphor color dots at each pixel position.

One phosphor dot emits:	red light
Another emits:	green light
Third emits:	blue light

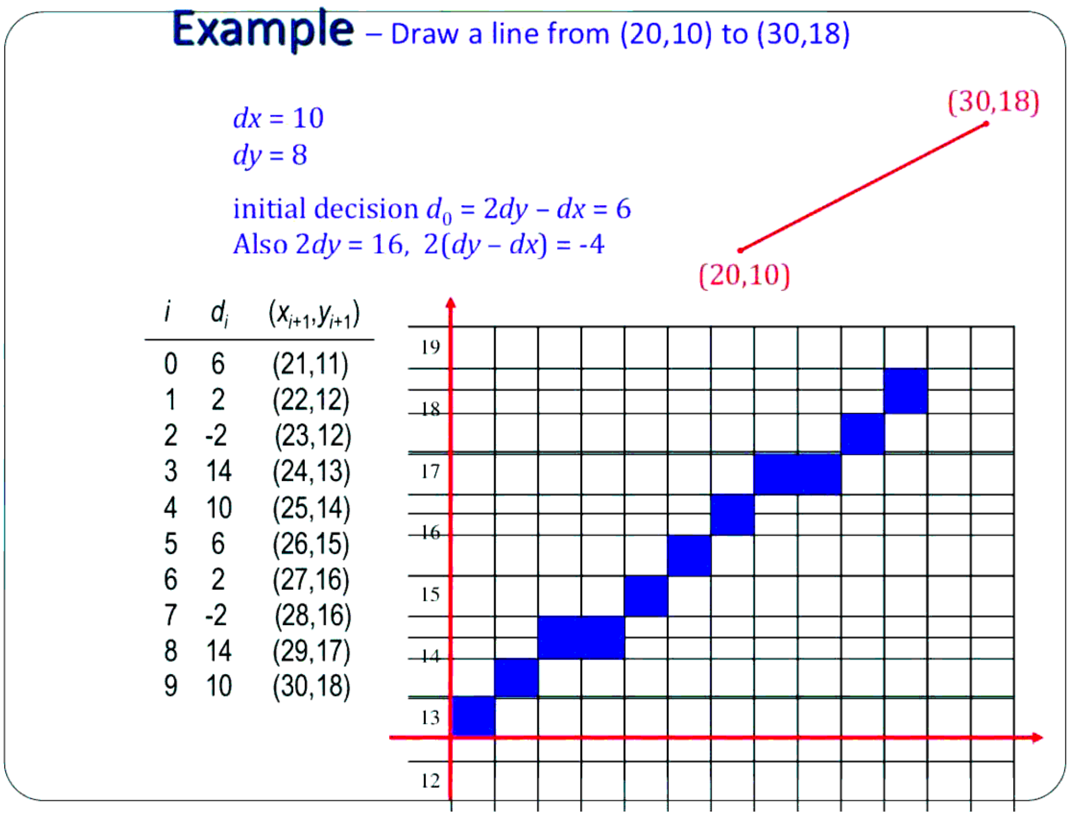
This type of CRT has 3 electron guns, one for each color dot and a shadow mask grid just behind the phosphor coated screen.

Shadow mask grid is pierced with small round holes in a triangular pattern.

Figure shows the delta-delta shadow mask method commonly used in color CRT system.

Q. 1 (B) USE BRESNEHAM'S ALGORITHM TO CHECK WHETHER PIXEL POSITION (28, 16) NEEDS TO BE PLOTTED OR NOT WHILE DRAWING A LINE FROM POINT (20,10) TO (30,18).

ANS.



Example – Draw a line from (20,10) to (30,18)

$dx = 10$
$dy = 8$

initial decision $d_0 = 2dy - dx = 6$
Also $2dy = 16$, $2(dy - dx) = -4$

i	d_i	(x_{i+1}, y_{i+1})
0	6	(21,11)
1	2	(22,12)
2	-2	(23,12)
3	14	(24,13)
4	10	(25,14)
5	6	(26,15)
6	2	(27,16)
7	-2	(28,16)
8	14	(29,17)
9	10	(30,18)

Q.2 (A) 3 EXPLAIN BRIEFLY THE ROLE OF COMPUTER GRAPHICS IN MOVIES AND CARTOON FILMS. FIND THE REFRESH RATE OF A 512 * 512 FRAME BUFFER, IF THE ACCESS TIME FOR EACH PIXEL IS 200 NANOSECONDS.

ANS. The film industry has been shifting and advancing in such a manner that even actors can now be substituted with robots or animated cartoons to represent human beings in films. Artificial sceneries are now becoming difficult to distinguish from real places. Computer graphics and animation have greatly affected the operations of film production globally and proven to be an inevitable source that create and enhance realistic images and sound in films that resembles a real environment. Because of its flexibility, most audiences are now interested in scientific films, and similar genres which highly use computer graphics and animation. The introduction of computer has also increased demand for many digitalized and online platforms for such films. This study focuses on the existence of computer graphics and use of animation in Zimbabwean film and television production. The paper will trace the development of computer graphics and animation, and its future prospects. Presently, the Zimbabwean film industry has been less appealing to most Zimbabwean audience leaving the industry less competitive.

In this f a s t d e v e l o p i n g i n f o r m a t i o n society computer graphics and animation has become the most important communication tool. The development of the film media and the progress of computer graphics and animation are intimately related to each other in many ways. The film media not only provide information and entertainment, but they also provide directly and indirectly many opportunities to the unemployed people in India. Every year, thousands of films are released with computer graphics and animation. Multibillion film projects are now very common in India. Computer graphics and visual effects are now becoming the part of all cinema projects. Even though 3D animated film and ordinary commercial film has differed, the visual effects are common. Many famous Indian film directors have used visual effects to communicate their dream effectively. Computer graphics have impact on the target audience, fuel to our knowledge, attitudes, and behaviors. 3D, Animations, graphics are inevitable resources in the creative industries. Advertising on Television, Dailies, and other electronic media at present fully depended upon the computerized format. The main aim of this research paper is to study and analyze the impact of computer graphics and animation in the main stream cinema. This paper also analyses the communicative aspects of computer graphics in the film media.

So if a pixel only takes 200ns, 512*512 are 262144 pixels wiil take

200ns/pixel * 262144 pixels = 52428800 ns

So if a second is 1000.000.000 ns, we should make a convertion to seconds

1(second/1000.000.00 ns) * 52428800 ns = 0,0524288 seconds, takes for one frame,

With a Simple 3 rule

if 0,0524288 takes 1 frame

1sg how many frames will take?

0,0524288 sec - 1 Frame

1 Sec ——————- X?

so X= (1Frames 1 Sec) /(0,0524288 sec) = 19,073486328125 Frames.

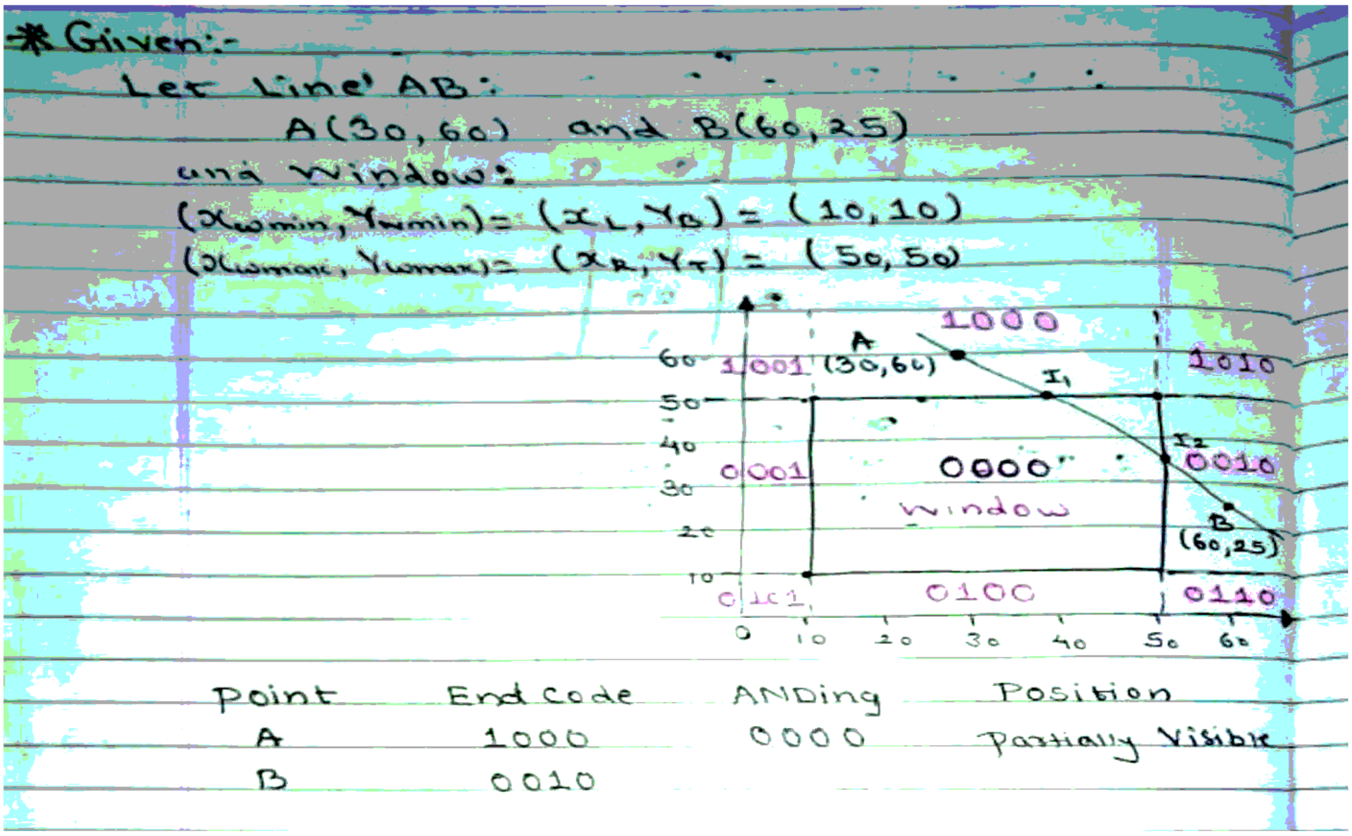
So 19,073486328125 Frames will be showed in a second

HENCE

The refresh rate of a 512*512 frame buffer, if acces time for each pixel is 200ns, is 19,07 Frames/Second.

Q. 2 (B) EXPLAIN ANY LINE CLIPPING ALGORITHM AGAINST A BOUNDRY WITH SUITABLE EXAMPLE.

ANS.



* Given:-

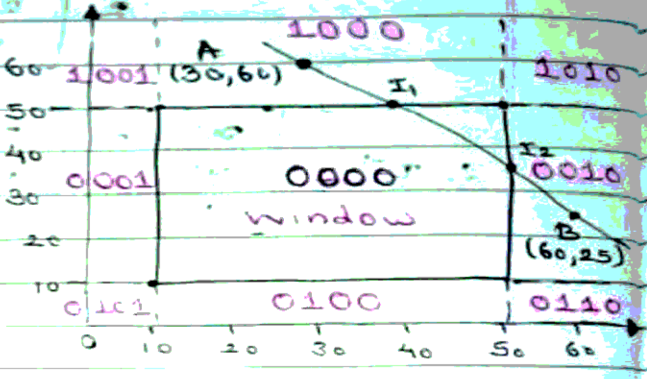
Let Line AB:

A(30, 60) and B(60, 25)

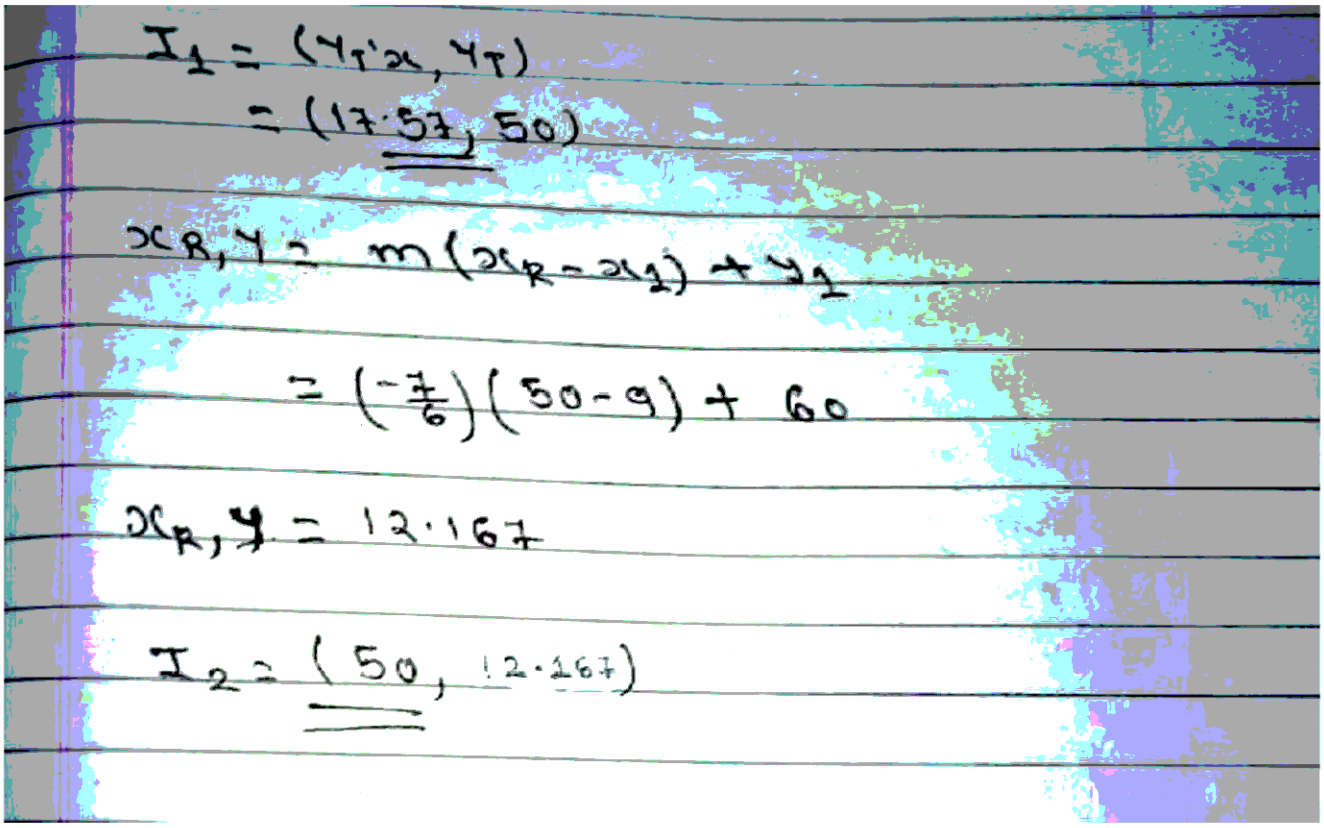
and Window:

$(x_{wmin}, y_{wmin}) = (x_L, y_B) = (10, 10)$

$(x_{wmax}, y_{wmax}) = (x_R, y_T) = (50, 50)$



Point	End code	ANDing	Position
A	1000	0000	Partially Visible
B	0010		



$I_1 = (y_T{'}x, y_T)$

$= (\underline{17.57}, 50)$

$x_R, y = m(x_R - x_1) + y_1$

$= \left(-\frac{7}{6}\right)(50 - 9) + 60$

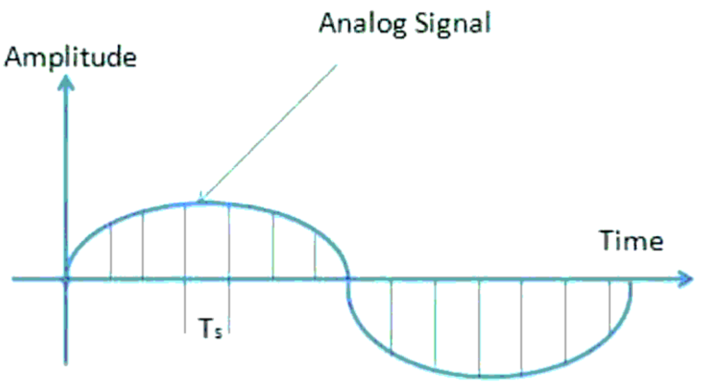
$x_R, y = 12.167$

$I_2 = (\underline{50}, 12.167)$

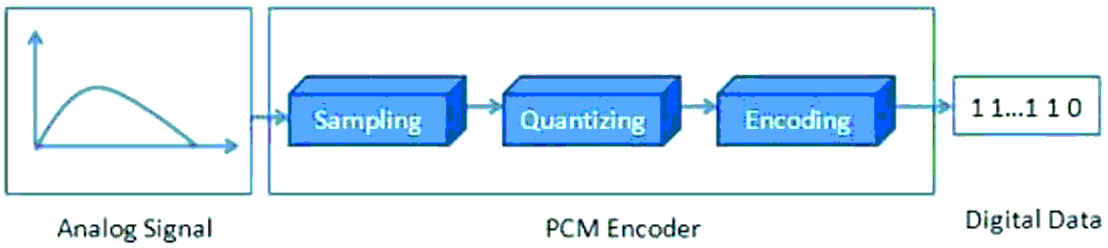
Q. 3 (A) BRIEFLY MENTION THE AUDIO DIGITIZATION PROCEDURE.

ANS Digitization of Sound

Digitization is a process of converting the analog signals to a digital signal. There are three steps of digitization of sound.



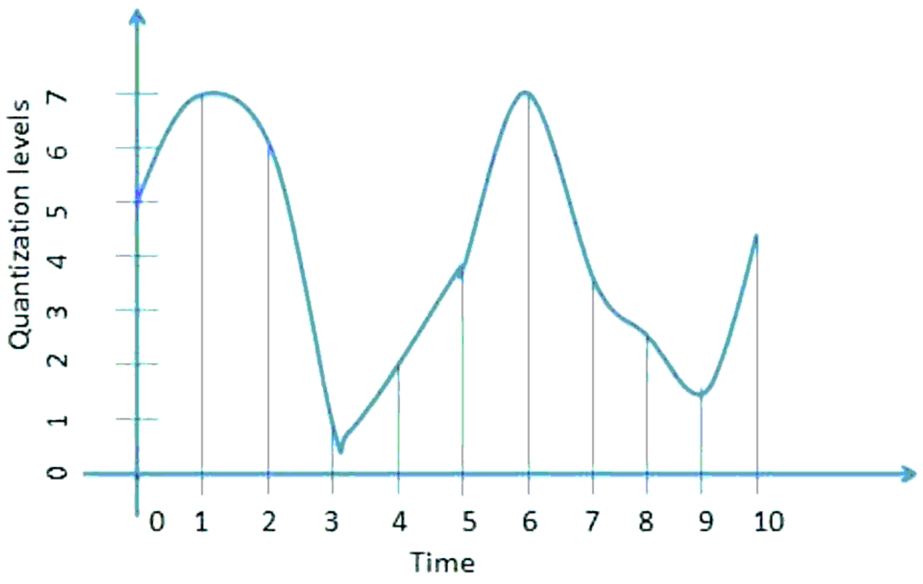
- **Sampling** - Sampling is a process of measuring air pressure amplitude at equally spaced moments in time, where each measurement constitutes a sample. A sampling rate is the number of times the analog sound is taken per second. A higher sampling rate implies that more samples are taken during the given time interval and ultimately, the quality of reconstruction is better. The sampling rate is measured in terms of Hertz, Hz in short, which is the term for Cycle per second. A sampling rate of 5000 Hz(or 5kHz,which is more common usage) implies that mt uj vu8i 9ikuhree sampling rates most often used in multimedia are 44.1kHz(CD-quality), 22.05kHz and 11.025kHz.
- **Quantization** - Quantization is a process of representing the amplitude of each sample as integers or numbers. How many numbers are used to represent the value of each sample known as sample size or bit depth or resolution. Commonly used sample sizes are either 8 bits or 16 bits. The larger the sample size, the more accurately the data will describe the recorded sound. An 8-bit sample size provides 256 equal measurement units to describe the level and frequency of the sound in that slice of time. A 16-bit sample size provides 65,536 equal units to describe the sound in that sample slice of time. The value of each sample is rounded off to the nearest integer (quantization) and if the amplitude is greater than the intervals available, clipping of the top and bottom of the wave occurs.
- **Encoding** - Encoding converts the integer base-10 number to a base-2 that is a binary number. The output is a binary expression in which each bit is either a 1(pulse) or a 0(no pulse).



Quantization of Audio

Quantization is a process to assign a discrete value from a range of possible values to each sample. Number of samples or ranges of values are dependent on the number of bits used to represent each sample. Quantization results in stepped waveform resembling the source signal.

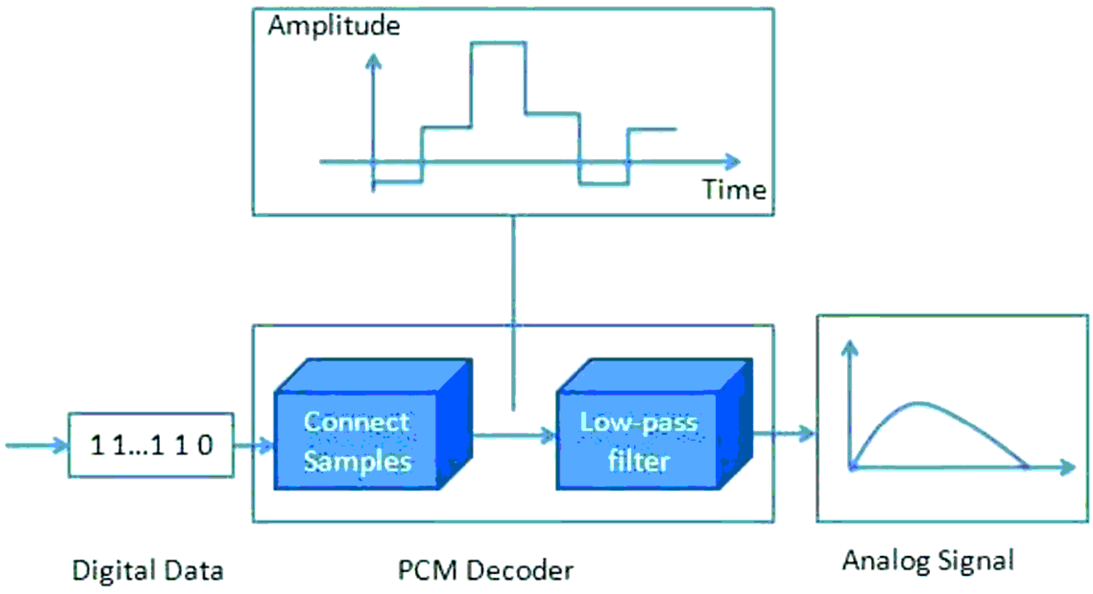
- **Quantization Error/Noise** - The difference between sample and the value assigned to it is known as quantization error or noise.
- **Signal to Noise Ratio (SNR)** - Signal to Ratio refers to signal quality versus quantization error. Higher the Signal to Noise ratio, the better the voice quality. Working with very small levels often introduces more error. So instead of uniform quantization, non-uniform quantization is used as companding. Companding is a process of distorting the analog signal in controlled way by compressing large values at the source and then expanding at receiving end before quantization takes place.



Transmission of Audio

In order to send the sampled digital sound/ audio over the wire that it to transmit the digital audio, it is first to be recovered as analog signal. This process is called de-modulation.

- **PCM Demodulation** - PCM Demodulator reads each sampled value then apply the analog filters to suppress energy outside the expected frequency range and outputs the analog signal as output which can be used to transmit the digital signal over the network.



Q. 3 (B) SHOW THE TRANSFORMATION MATRIX FOR A REFLECTION ABOUT A LINE Y = X , IS EQUIVALENT TO A REFLECTION RELATIVE TO THE Y – AXIS FOLLOWED BY A COUNTER CLOCKWISE ROTATION OF 90 DEGREE.

ANS :- **Reflection about the line** y=x
The effect of this reflection is to switch the x and y values of the reflected point. The matrix is:

- A=(0110)

CCW rotation of a point
For **CCW** rotations about origin by angle α:

- R(α)=(cosα−sinαsinαcosα)

If we combine these in the order suggested:
x'=A R(90_o) x
x'=(0110)(0−110)x
=(100−1)x
⇒(x'y')=(100−1)(xy)=(x−y)
That is equivalent to a reflection in **x-axis** .

Making it a **CW** rotation:
(x'y')=(0110)(01−10)(xy)
=(−1001)(xy)=(−xy)
That is a reflection in the **y-axis.**

Q. 4 (A) DERIVE EXPRESSION FOR CONVERTING RGB COLOR PARAMETER TO HSV VALUES.

ANS. I'm a programmer looking to build an RGB to HSV color converter. I found an algorithm, but I have very little mathematical background and I'm not quite sure what's going on with it. A step-by-step breakdown of exactly what is happening would be tremendously helpful so that I could code it. RGB and HSV are each sets of three values. R, G, and B are each 0-255, while H is 0-360° and S and V are each 0%-100%.

Here's the algorithm:

The R,G,B values are divided by 255 to change the range from 0..255 to 0..1:
R' = R/255

G' = G/255

B' = B/255

Cmax = max(R', G', B')

Cmin = min(R', G', B')

Δ = Cmax - Cmin

Hue calculation:

$$H = \begin{cases} 60^\circ \times \left(\frac{G'-B'}{\Delta} mod 6\right) & , Cmax = R' \\ 60^\circ \times \left(\frac{B'-R'}{\Delta} + 2\right) & , Cmax = G' \\ 60^\circ \times \left(\frac{R'-G'}{\Delta} + 4\right) & , Cmax = B' \end{cases}$$

Saturation calculation:

$$S = \begin{cases} 0 & , C_{max} = 0 \\ \frac{\Delta}{C_{max}} & , C_{max} \neq 0 \end{cases}$$

Value calculation: V = Cmax

Q. 4 (B) WRITE DOWN THE STEPS IN DESIGNING ANIMATION SEQUENCES.

ANS. the **6 basic steps of animation**, including:

1. Shooting reference video
2. Key posing
3. Blocking
4. Splining
5. Smoothing
6. Adding life

Step 1: Shooting Reference video

This is a very important and **overlooked** step. It's weird how people really think they know what certain actions look like and how long they take, but in reality they are often wrong.

Physical actions is something you **need to analyze** before animating, especially if you're a beginner.

You have a shot of a guy throwing a baseball? Better YouTube some reference video of pitchers throwing balls. Don't assume you know what an action looks like just because you've seen it before. Looking at an action as an **animator** is completely different than looking at it as a regular viewer.

Step 2: Posing

After shooting a reference, it's time to create the key poses of the shot.

These poses are called key poses because they are the most important poses of the shot. These are the poses that convey the story of the shot. We better make sure we **get those poses right**, because we're going to build on those for the rest of the process.

Step 3: Blocking

Once we're happy with our key poses, we start **breaking down** the movement from each pose to the next by adding 'in betweens' (also known as breakdown poses or passing poses). These are the poses that connect the key poses.

We keep adding more poses until the movement looks as good as it could, while still staying in **stepped mode** (stepped mode is when you don't allow interpolation between poses, which results in a very choppy/blocky motion).

Step 4: Splining

Splining is a 3D animation term. It's the process in which you convert the interpolation of the keys from stepped to spline. In other words – you make the computer connect the movement between each of your poses, and that makes the movement **look smoother.**

The problem is that the computer doesn't do a very good job at interpolating. It only works with what it has. That's why the better the blocking is – the better the splined version is going to look.

Step 5: Smoothing and offset

Now that all of our keys are on spline mode, we have to work on them. We need to clean up all the curves and make sure the movement looks smooth.

It's also a good idea to **offset** some of the actions so it doesn't look so **'stop and start'**, as if the character is doing all the motion at once. By the end of this step your shot should look pretty solid and almost finished.

Step 6: Adding life

This step is the a lot of fun. We've already finished with the grunt work of animating and it's time to add the fun stuff. In this step we add **small imperfections** that bring life to the character. Maybe an extra blink or a mouth twitch here and there. The difference between the last 2 steps is small but very noticeable.

Q. 5 (A) CONSTRUCT A B - SPLINE CURVE OF THIRD ORDER WITH FOUR POLYGON VERTICES P(1,1) , Q(2,3), R(4,3) AND S(6,4).

ANS.

The normalized B-spline blending functions are defined recursively by

$$N_{i,1}(t) = \begin{cases} 1 & \text{if } u \in [t_i, t_{i+1}) \\ 0 & \text{otherwise} \end{cases} \qquad (1)$$

and if $k > 1$,

$$N_{i,k}(t) = \left(\frac{t - t_i}{t_{i+k-1} - t_i} \right) N_{i,k-1}(t) + \left(\frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} \right) N_{i+1,k-1}(t) \qquad (2)$$

where $\{t_0, t_1, \ldots, t_{n+k}\}$ is a non-decreasing sequence of knots, and k is the order of the curve

These functions are difficult to calculate directly for a general knot sequence. However, if the knot sequence is uniform, it is quite straightforward to calculate these functions – and they have some suprising properties.

Calculating the Blending Functions using a Uniform Knot Sequence

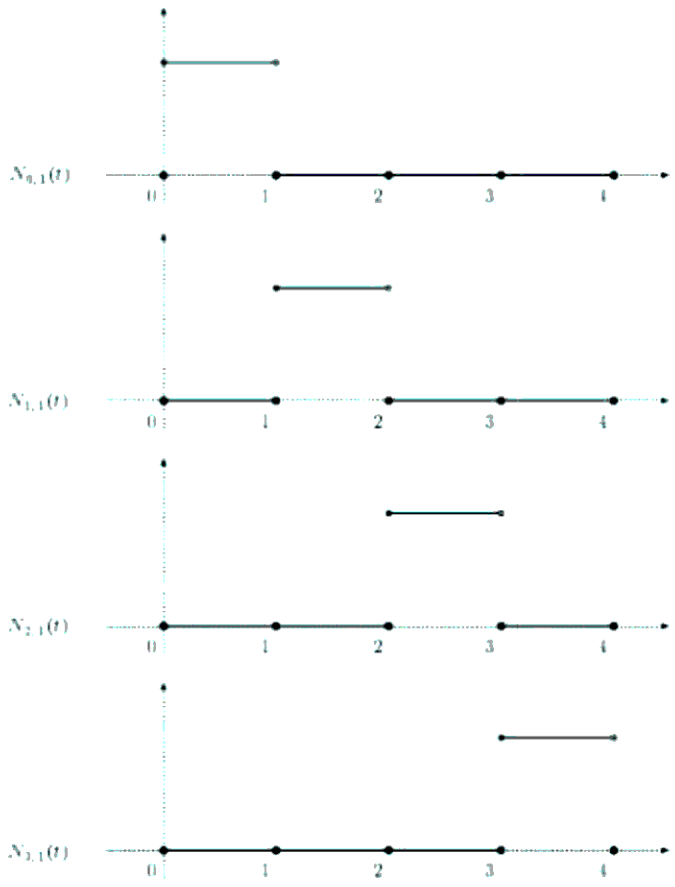
Assume that $\{t_0, t_1, t_2, \ldots, t_n\}$ is a uniform knot sequence, *i.e.*, $\{0, 1, 2, \ldots, n\}$. This will simplify the calculation of the blending functions, as $t_i = i$.

Blending Functions for k = 1

if $k = 1$, then by using equation (1), we can write the normalized blending functions as

$$N_{i,1}(t) = \begin{cases} 1 & \text{if } u \in [i, i+1) \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

These are shown together in the following figure, where we have plotted $N_{0,1}$, $N_{1,1}$, $N_{2,1}$, and $N_{3,1}$ respectively, over five of the knots. Note the white circle at the end of the line where the functions value is 1. This represents the affect of the "open-closed" interval found in equation (1).



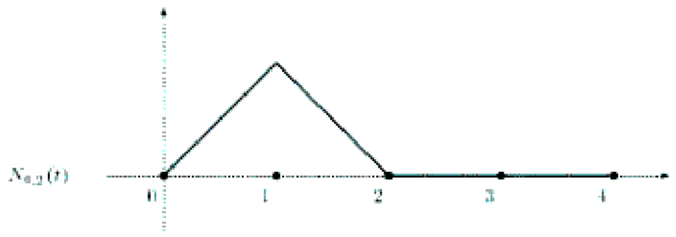
These functions have support (the region where the curve is nonzero) in an interval, with $N_{i,1}$ having support on $[i, i+1)$. They are also clearly shifted versions of each other – e.g., $N_{i+1,1}$ is just $N_{i,1}$ shifted one unit to the right. In fact, we can write $N_{i,1}(t) = N_{0,1}(t-i)$

Blending Functions for k = 2

If $k = 2$ then $N_{0,2}$ can be written as a weighted sum of $N_{0,1}$ and $N_{1,1}$ by equation (2). This gives

$$\begin{aligned} N_{0,2}(t) &= \frac{t - t_0}{t_1 - t_0} N_{0,1}(t) + \frac{t_2 - t}{t_2 - t_1} N_{1,1}(t) \\ &= t N_{0,1}(t) + (2 - t) N_{1,1}(t) \\ &= \begin{cases} t & \text{if } 0 \leq t < 1 \\ 2 - t & \text{if } 1 \leq t < 2 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

This curve is shown in the following figure. The curve is piecewise linear, with support in the interval $[0, 2]$.

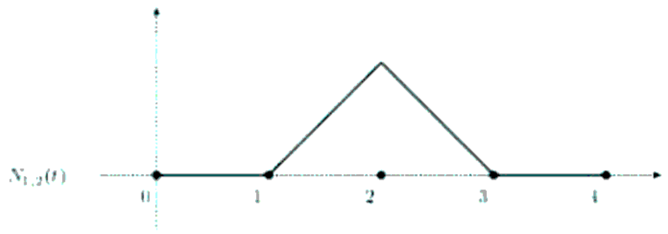


These functions are commonly referred to as "hat" functions and are used as blending functions in many linear interpolation problems.

Similarly, we can calculate $N_{1,2}$ to be

$$\begin{aligned} N_{1,2}(t) &= \frac{t - t_1}{t_2 - t_1} N_{1,1}(t) + \frac{t_3 - t}{t_3 - t_2} N_{2,1}(t) \\ &= (t - 1) N_{1,1}(t) + (3 - t) N_{2,1}(t) \\ &= \begin{cases} t - 1 & \text{if } 1 \leq t < 2 \\ 3 - t & \text{if } 2 \leq t < 3 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

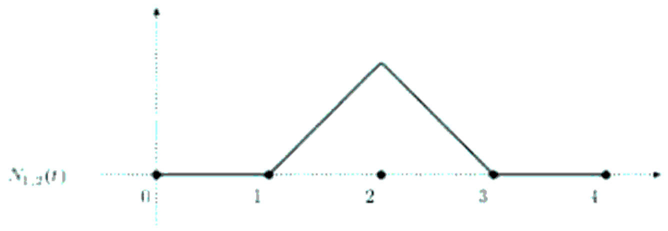
This curve is shown in the following figure, and it is easily seen to be a shifted version of $N_{0,2}$.



Finally, we have that

$$N_{1,2}(t) = \begin{cases} t - 2 & \text{if } 2 \leq t < 3 \\ 4 - t & \text{if } 3 \leq t < 4 \\ 0 & \text{otherwise} \end{cases}$$

which is shown in the following illustration.



These nonzero portion of these curves each cover the intervals spanned by three knots – e.g., $N_{1,2}$ spans the interval $[1, 3)$. The curves are piecewise linear, made up of two linear segments joined continuously.

Since the curves are shifted versions of each other, we can write

$$N_{i,2}(t) = N_{0,2}(t - i)$$

Blending Functions for k = 3

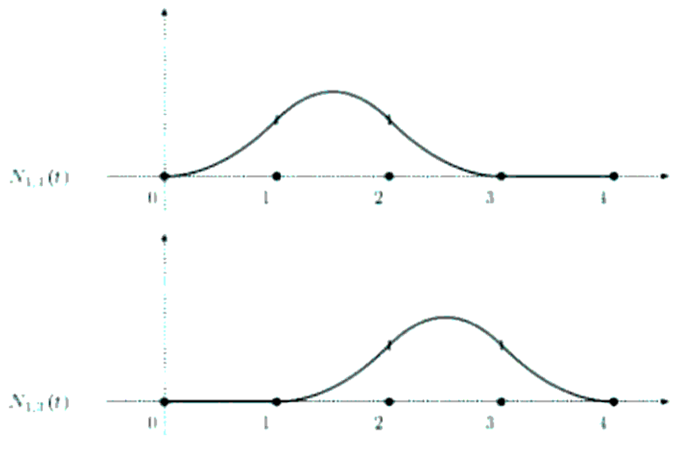
For the case $k = 3$, we again use equation (2) to obtain

$$\begin{aligned}
N_{0,3}(t) &= \frac{t - t_0}{t_2 - t_0} N_{0,2}(t) + \frac{t_3 - t}{t_3 - t_1} N_{1,2}(t) \\
&= \frac{t}{2} N_{0,2}(t) + \frac{3 - t}{2} N_{1,2}(t) \\
&= \begin{cases} \frac{t^2}{2} & \text{if } 0 \le t < 1 \\ \frac{t}{2}(2 - t) + \frac{3-t}{2}(t - 1) & \text{if } 1 \le t < 2 \\ \frac{(3-t)^2}{2} & \text{if } 2 \le t < 3 \\ 0 & \text{otherwise} \end{cases} \\
&= \begin{cases} \frac{t^2}{2} & \text{if } 0 \le t < 1 \\ \frac{-3+6t-2t^2}{2} & \text{if } 1 \le t < 2 \\ \frac{(3-t)^2}{2} & \text{if } 2 \le t < 3 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

and by nearly identical calculations,

$$\begin{aligned}
N_{1,3}(t) &= \frac{t - t_1}{t_3 - t_1} N_{1,2}(t) + \frac{t_4 - t}{t_4 - t_2} N_{2,2}(t) \\
&= \frac{t - 1}{2} N_{1,2}(t) + \frac{4 - t}{2} N_{2,2}(t) \\
&= \begin{cases} \frac{(t-1)^2}{2} & \text{if } 1 \le t < 2 \\ \frac{-11+10t-2t^2}{2} & \text{if } 2 \le t < 3 \\ \frac{(4-t)^2}{2} & \text{if } 3 \le t < 4 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

These curves are shown in the following figure. They are clearly piecewise quadratic curves, each made up of three parabolic segments that are joined at the knot values (we have placed tick marks on the curves to show where they join).



The nonzero portion of these two curves each span the interval between four consecutive knots – e.g., the nonzero portion of $N_{1,3}$ spans the interval [1, 4]. Again, $N_{1,3}$ can be seen visually to be a shifted version of $N_{0,3}$. (This fact can also be seen analytically by substituting $t + 1$ for t in the equation for $N_{1,3}$.) We can write

$$N_{i,3}(t) = N_{0,3}(t - i)$$

Q. 5 (B) WRITE ODWN THE METHOD FOR SIMULATING ACCELARATION IN COMPUTER GRAPHICS.

ANS. "Simulation Acceleration" or "Accelerated Verification" are terms commonly used to describe a verification environment in which the Device Under Test is synthesized and runs on the emulator, while the testbench runs on the simulator. This blog describes a technique for maximizing the performance of the simulator and thus increasing the overall performance of the Simulation Acceleration.

In Simulation Acceleration (SA), the verification environment is partitioned into two levels. The lower level contains the components that are connected to the DUT – the BFM, transactor, collector, and so on. These components are implemented in a synthesizable language and loaded onto the emulator with the DUT. The upper level of the verification environment implements functionalities such as sequences, generation constraints, and checkers. These components run on the simulator and are often modeled at the transaction level (TLM).

There is no real parallelism between these two platforms. When the synthesized code runs on the emulator, the simulator waits. When there is a context switch– for example, when the BFM calls a **get_next_item()**–the accelerator halts and the simulator runs. When the simulator performs another context switch, the emulator software continues execution and the simulator halts, and so on until the end of the test.

This flow is illustrated in the following diagram. When the test begins, the upper level generates the units and their fields, runs initialization methods, and so forth. After generating the first data item/s, it passes them to the lower level (for example, via SCEMI pipes). Now the lower level transactors interact with the DUT, and the upper level halts. When the lower level needs to pass a status to the higher level or a request for the next items, the upper level starts running again and the lower level halts, and so on until the end of the test.

Q. 6 (A) FIND THE POINTS ON BEZIER CURVE WHICH HAS STARTING AND ENDING POINTS P0(2,3) AND P3(4, -3) RESPECTIVELY AND IS CONTROLLED BY P1(6,6) AND P2(8,1) FOR U = 0,2, 0.5 AND 0.9.

ANS.

We have-

- The given curve is defined by 4 control points.
- So, the given curve is a cubic bezier curve.

The parametric equation for a cubic bezier curve is-

$$\mathbf{P(t) = B_0(1\text{-}t)^3 + B_1 3t(1\text{-}t)^2 + B_2 3t^2(1\text{-}t) + B_3 t^3}$$

Substituting the control points B_0, B_1, B_2 and B_3, we get-

$P(t) = [1\ 0](1\text{-}t)^3 + [3\ 3]3t(1\text{-}t)^2 + [6\ 3]3t^2(1\text{-}t) + [8\ 1]t^3$ ……..(1)

Now,

To get 5 points lying on the curve, assume any 5 values of t lying in the range 0 <= t <= 1.

Let 5 values of t are 0, 0.2, 0.5, 0.7, 1

For t = 0:

Substituting t=0 in (1), we get-

$P(0) = [1\ 0](1\text{-}0)^3 + [3\ 3]3(0)(1\text{-}t)^2 + [6\ 3]3(0)^2(1\text{-}0) + [8\ 1](0)^3$

$P(0) = [1\ 0] + 0 + 0 + 0$

$P(0) = [1\ 0]$

For t = 0.2:

Substituting t=0.2 in (1), we get-

$P(0.2) = [1\ 0](1\text{-}0.2)^3 + [3\ 3]3(0.2)(1\text{-}0.2)^2 + [6\ 3]3(0.2)^2(1\text{-}0.2) + [8\ 1](0.2)^3$

$P(0.2) = [1\ 0](0.8)^3 + [3\ 3]3(0.2)(0.8)^2 + [6\ 3]3(0.2)^2(0.8) + [8\ 1](0.2)^3$

$P(0.2) = [1\ 0] \times 0.512 + [3\ 3] \times 3 \times 0.2 \times 0.64 + [6\ 3] \times 3 \times 0.04 \times 0.8 + [8\ 1] \times 0.008$

$P(0.2) = [1\ 0] \times 0.512 + [3\ 3] \times 0.384 + [6\ 3] \times 0.096 + [8\ 1] \times 0.008$

$P(0.2) = [0.512\ 0] + [1.152\ 1.152] + [0.576\ 0.288] + [0.064\ 0.008]$

$P(0.2) = [2.304\ 1.448]$

For t = 0.5:

Substituting t=0.5 in (1), we get-

$P(0.5) = [1\ 0](1-0.5)^3 + [3\ 3]3(0.5)(1-0.5)^2 + [6\ 3]3(0.5)^2(1-0.5) + [8\ 1](0.5)^3$

$P(0.5) = [1\ 0](0.5)^3 + [3\ 3]3(0.5)(0.5)^2 + [6\ 3]3(0.5)^2(0.5) + [8\ 1](0.5)^3$

$P(0.5) = [1\ 0] \times 0.125 + [3\ 3] \times 3 \times 0.5 \times 0.25 + [6\ 3] \times 3 \times 0.25 \times 0.5 + [8\ 1] \times 0.125$

$P(0.5) = [1\ 0] \times 0.125 + [3\ 3] \times 0.375 + [6\ 3] \times 0.375 + [8\ 1] \times 0.125$

$P(0.5) = [0.125\ 0] + [1.125\ 1.125] + [2.25\ 1.125] + [1\ 0.125]$

$P(0.5) = [4.5\ 2.375]$

For t = 0.7:

Substituting t=0.7 in (1), we get-

$P(t) = [1\ 0](1-t)^3 + [3\ 3]3t(1-t)^2 + [6\ 3]3t^2(1-t) + [8\ 1]t^3$

$P(0.7) = [1\ 0](1-0.7)^3 + [3\ 3]3(0.7)(1-0.7)^2 + [6\ 3]3(0.7)^2(1-0.7) + [8\ 1](0.7)^3$

$P(0.7) = [1\ 0](0.3)^3 + [3\ 3]3(0.7)(0.3)^2 + [6\ 3]3(0.7)^2(0.3) + [8\ 1](0.7)^3$

$P(0.7) = [1\ 0] \times 0.027 + [3\ 3] \times 3 \times 0.7 \times 0.09 + [6\ 3] \times 3 \times 0.49 \times 0.3 + [8\ 1] \times 0.343$

$P(0.7) = [1\ 0] \times 0.027 + [3\ 3] \times 0.189 + [6\ 3] \times 0.441 + [8\ 1] \times 0.343$

$P(0.7) = [0.027\ 0] + [0.567\ 0.567] + [2.646\ 1.323] + [2.744\ 0.343]$

$P(0.7) = [5.984\ 2.233]$

Q. 6 (B) DEFINE FRACTAL DIMENTION. DISCUSS VARIOUS CONTINUITY CONDITIONS AT THE JOINING OF CURVES.

ANS A **fractal dimension** is an index for characterizing fractal patterns or sets by quantifying their complexity as a ratio of the change in detail to the change in scale. Several types of fractal dimension can be measured theoretically and empirically. Fractal dimensions are used to characterize a broad spectrum of objects ranging from the abstract to practical phenomena, including turbulence, river networks, urban growth human physiology, medicine, and market trends. The essential idea of *fractional* or *fractal* dimensions has a long history in mathematics that can be traced back to the 1600s, but the terms *fractal* and *fractal dimension* were coined by mathematician Benoit Mandelbrot in 1975

Fractal dimensions were first applied as an index characterizing complicated geometric forms for which the details seemed more important than the gross picture. For sets describing ordinary geometric shapes, the theoretical fractal dimension equals the set's familiar Euclidean or topological dimension. Thus, it is 0 for sets describing points (0-dimensional sets); 1 for sets describing lines (1-dimensional sets having length only); 2 for sets describing surfaces (2-dimensional sets having length and width); and 3 for sets describing volumes (3-dimensional sets having length, width, and height). But this changes for fractal sets. If the theoretical fractal dimension of a set exceeds its topological dimension, the set is considered to have fractal geometry.

Geometric continuity is another method to join two successive curve sections. G0 continuity is the same as parametric continuity (i.e., two curves sections to be joined must have same coordinate position at the boundary point) i.e., curve section are joined together such that they have same coordinates position at the boundary point. First order geometric continuity G(1) means that the tangent vectors are the same at join point of two successive curves i.e., the parametric first derivative are proportional at the intersection of two successive sections while second order geometric continuity is G2 means that both first and second order parametric derivates of the two curve sections are proportional at their boundary. In G2 curvature of the two successive curve sections will match at the joining position.

Q. 7 (A) DISCUSS DEPTH BUFFER AND DEPTH SORTING METHODS FOR HIDDEN SURFACE REMOVAL.

ANS. Z-Buffer Algorithm

It is also called a **Depth Buffer Algorithm**. Depth buffer algorithm is simplest image space algorithm. For each pixel on the display screen, we keep a record of the depth of an object within the pixel that lies closest to the observer. In addition to depth, we also record the intensity that should be displayed to show the object. Depth buffer is an extension of the frame buffer. Depth buffer algorithm requires 2 arrays, intensity and depth each of which is indexed by pixel coordinates (x, y).

Algorithm

For all pixels on the screen, set depth [x, y] to 1.0 and intensity [x, y] to a background value.

For each polygon in the scene, find all pixels (x, y) that lie within the boundaries of a polygon when projected onto the screen. For each of these pixels:

(a) Calculate the depth z of the polygon at (x, y)

(b) If z < depth [x, y], this polygon is closer to the observer than others already recorded for this pixel. In this case, set depth [x, y] to z and intensity [x, y] to a value corresponding to polygon's shading. If instead z > depth [x, y], the polygon already recorded at (x, y) lies closer to the observer than does this new polygon, and no action is taken.

3. After all, polygons have been processed; the intensity array will contain the solution.

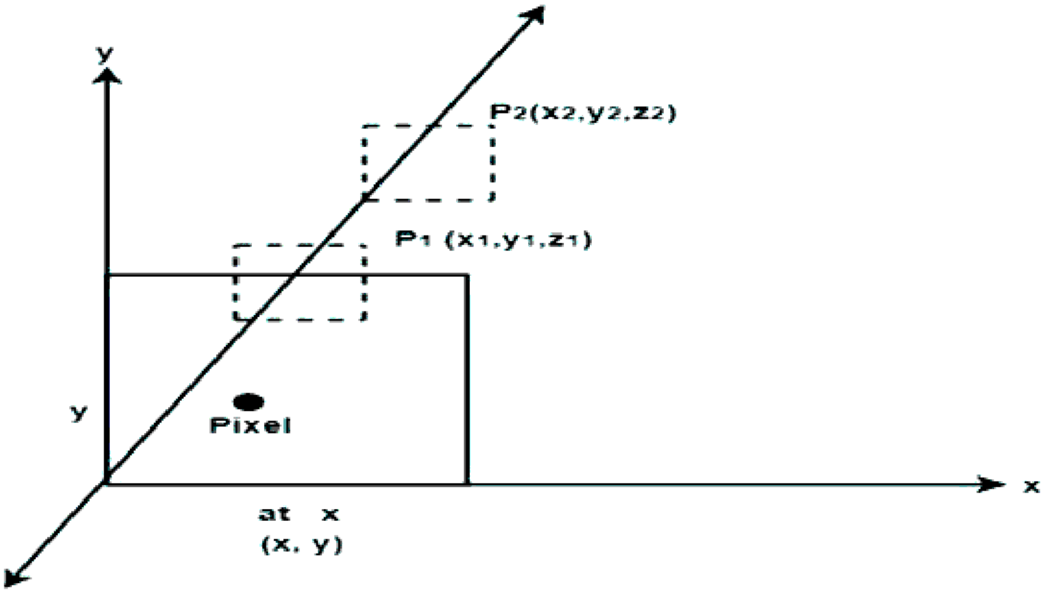
4. The depth buffer algorithm illustrates several features common to all hidden surface algorithms.

Painter Algorithm

It came under the category of list priority algorithm. It is also called a **depth-sort algorithm**. In this algorithm ordering of visibility of an object is done. If objects are reversed in a particular order, then correct picture results.

Objects are arranged in increasing order to z coordinate. Rendering is done in order of z coordinate. Further objects will obscure near one. Pixels of rear one will overwrite pixels of farther objects. If z values of two overlap, we can determine the correct order from Z value as shown in fig (a).

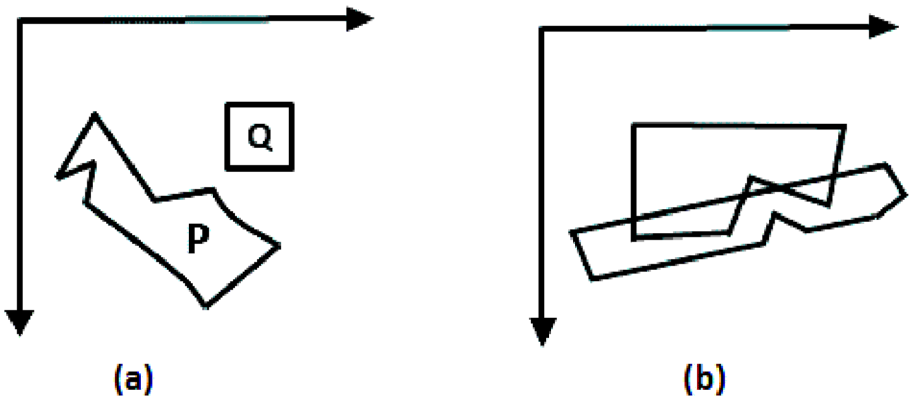
If z objects overlap each other as in fig (b) this correct order can be maintained by splitting of objects.



5. First, it requires a representation of all opaque surface in scene polygon in this case.

6. These polygons may be faces of polyhedral recorded in the model of scene or may simply represent thin opaque 'sheets' in the scene.

7. The IInd important feature of the algorithm is its use of a screen coordinate system. Before step 1, all polygons in the scene are transformed into a screen coordinate system using matrix multiplication.



Depth sort algorithm or painter algorithm was developed by Newell, sancha. It is called the painter algorithm because the painting of frame buffer is done in decreasing order of distance. The distance is from view plane. The polygons at more distance are painted firstly.

The concept has taken color from a painter or artist. When the painter makes a painting, first of all, he will paint the entire canvas with the background color. Then more distance objects like mountains, trees are added. Then rear or foreground objects are added to picture. Similar approach we will use. We will sort surfaces according to z values. The z values are stored in the refresh buffer.

Steps performed in-depth sort

1. Sort all polygons according to z coordinate.
2. Find ambiguities of any, find whether z coordinate overlap, split polygon if necessary.
3. Scan convert each polygon in increasing order of z coordinate.

Painter Algorithm

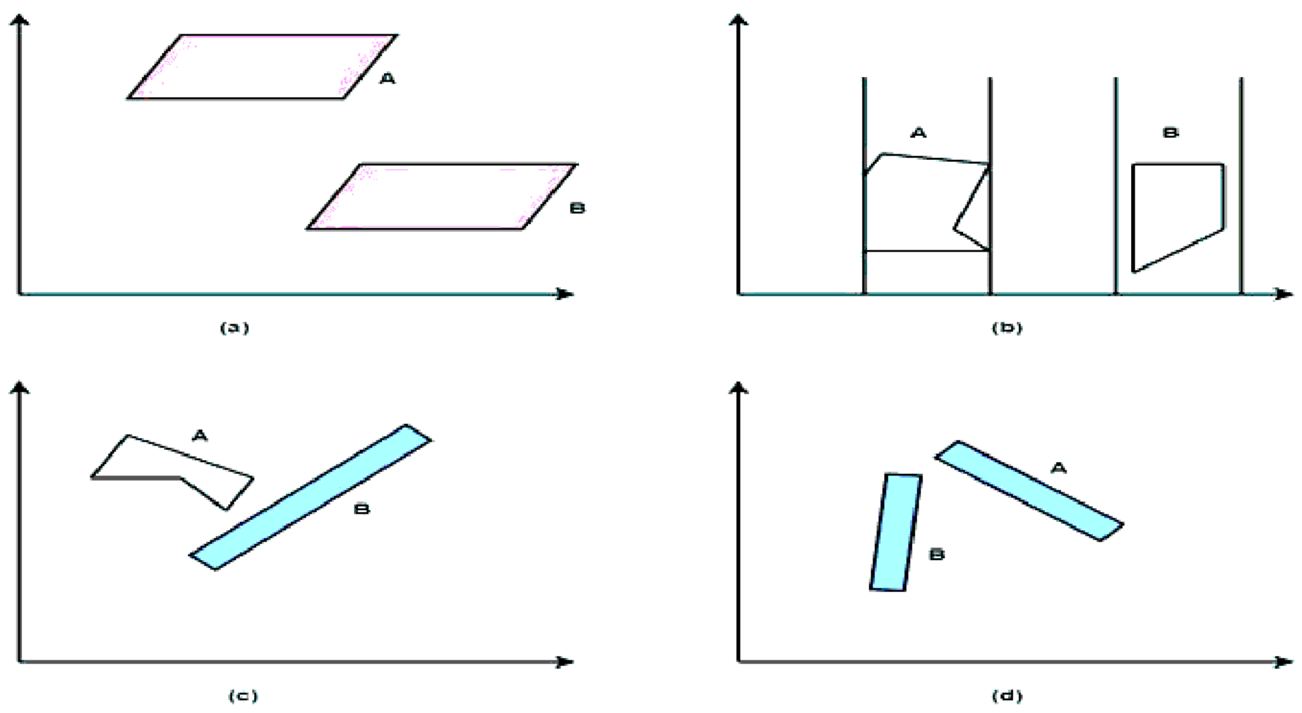
Step1: Start Algorithm

Step2: Sort all polygons by z value keep the largest value of z first.

Step3: Scan converts polygons in this order.
Test is applied

1. Does A is behind and non-overlapping B in the dimension of Z as shown in fig (a)
2. Does A is behind B in z and no overlapping in x or y as shown in fig (b)
3. If A is behind B in Z and totally outside B with respect to view plane as shown in fig (c)
4. If A is behind B in Z and B is totally inside A with respect to view plane as shown in fig (d)

The success of any test with single overlapping polygon allows F to be painted.



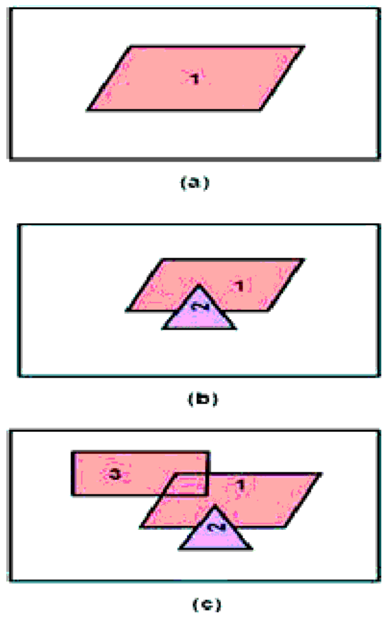
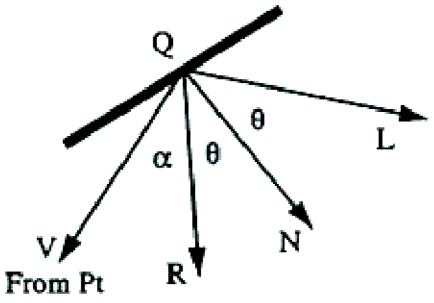


Figure showing addition of surface one by one and then painting done using painter algorithm

Q. 7 (B) EXPLAIN ILLUMINATION MODEL TO CALCULATE THE SURFACE INTENSITY DUE TO MULTIPLE SOURCES OF LIGHTS.

ANS *Three Components to Illumination of Objects*

- Ambient - Total of all the light bouncing.
- Diffuse reflection - Matted surfaces.
- Specular reflection - Mirror effect.



Ambient

$$I_a = \begin{bmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{bmatrix}$$ *- intensity of the ambient light.*

$$k_a = \begin{bmatrix} k_{ar} \\ k_{ag} \\ k_{ab} \end{bmatrix}$$ *- ambient reflection coefficient.*

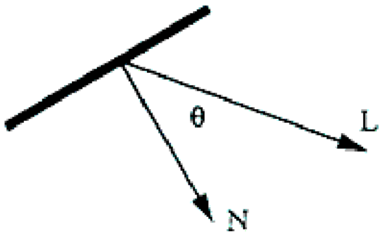
I_a is a property of the scene and

k_a is a material property (varies with object) Illumination (due to ambient).

$$I = I_a k_a = \begin{bmatrix} I_{ar}\ k_{ar} \\ I_{ag}\ k_{ag} \\ I_{ab}\ k_{ab} \end{bmatrix}$$

Diffuse Reflection

The light that gets reflected from the object's surface in all directions when a light source illuminates its surface. Based on Lamberts Law for reflected light off a matte surface.



I_p point light source's intensityk_d material's diffuse-reflection coefficient

Illumination: $I = I_p k_d \cos(\vartheta)$

Assumptions: $0 \le \vartheta \le \frac{pi}{2}$ otherwise diffuse contribution is zero.

Normalized vectors $||N|| = ||L|| = 1$

$\cos(\vartheta) = (N \cdot L)$

Ambient and Diffuse

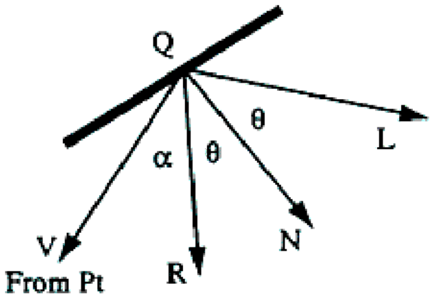
Illumination:

$$I = I_a k_a + I_p k_d (N \cdot L)$$

$$(N \cdot L) = \begin{cases} (N \cdot L), & \text{if } (N \cdot L) \ge 0 \\ 0, & \text{otherwise} \end{cases}$$

Specular reflection

The glare seen when a light source is mirrored on the surface of a shiny object Phong Model



k_s specular reflection coefficien

n Phong constant

Illumination:

$$I = I_a k_a + I_p \left[k_d \cdot \cos(\vartheta) + k_s \cdot \cos^n(\alpha) \right]$$

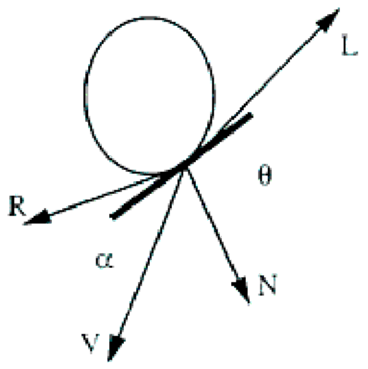
Calculate once for red, once for green and once for blue.

Computational considerations

$\cos(\vartheta) = \frac{(L \cdot N)}{}$ *(dot product)*

$||L||\cdot||N||$

$$\cos(\alpha) = \frac{(R\cdot V)}{||R||\cdot||V||} \quad \textit{(dot product)}$$

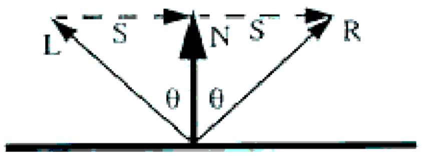


Diffuse requires: requires $\cos(\vartheta) \geq 0$

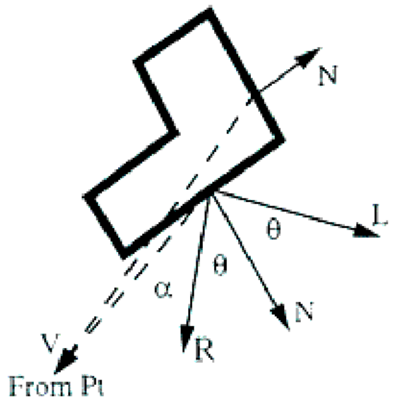
Specular requires: recuires $\cos(\alpha) \geq 0$ and $\cos(\vartheta) \geq 0$

Calculation Reflection Vector

$R = 2N(N\cdot L) - L$



Backface culling (and inside/otside coloring)



Backface culling:

If $(V\cdot N) < 0)$ *then the polygon cannot be seen.*

In/Out Coloring:

If $(V{\cdot}N) \geq 0$ then object is color.out
else object is color.in with (outward) normal $-N$

Painters Algorithm

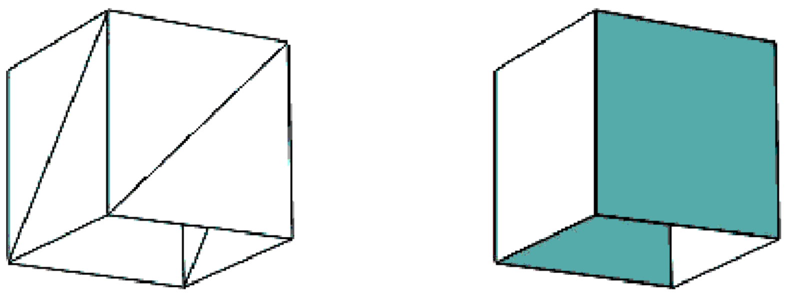
Sort polygons/surfaces by "distance" from the From Pt.

Then paint the polygons/surfaces starting with th one furthers away first.

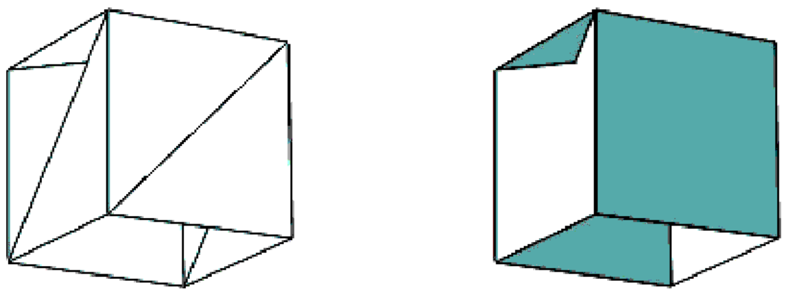
"distance" = cetroid, min. vertex??

Triangulate Open Box Example

OK

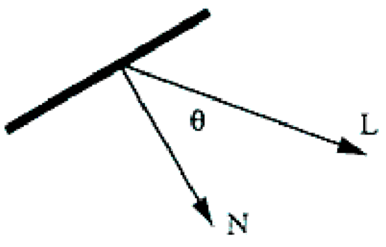


Not OK



Light Source

Point:



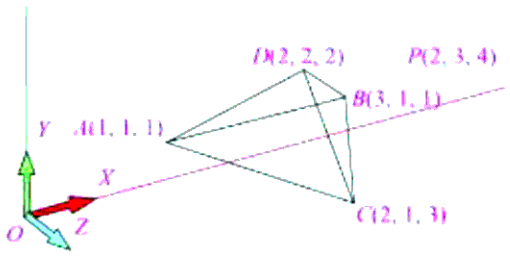
$L = P - Q$

P given in world coordinates.

Q. 8(A) A SOLID TETRAHEDRON IS GIVEN BY POSITION VECTORS A(1,1,1), B(3,1,1), C(2,1,3) AND D(2,2,2) AND A POINT LIGHT IS KEPT AT P(2,3,4). USING BACKFACE DETECTION METHOD FIND THE SURFACE ON WHICH THE LIGHT FALLS AND THE SURFACES WHICH ARE TO BE SHADOWED.

Solution

This is a convex object consisting of four flat triangular surfaces.



Calculating the surface normals projecting out of the object surfaces:

$$\mathbf{N}_{ACD} = \mathbf{AC} \times \mathbf{AD} = [(2-1)\mathbf{i} + (1-1)\mathbf{j} + (3-1)\mathbf{k}] \times [(2-1)\mathbf{i} + (2-1)\mathbf{j} + (2-1)\mathbf{k}]$$

$$= \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 1 & 0 & 2 \\ 1 & 1 & 1 \end{vmatrix} = -2\mathbf{i} + \mathbf{j} + \mathbf{k}.$$

$$\mathbf{N}_{CBD} = \mathbf{CB} \times \mathbf{CD} = [(3-2)\mathbf{i} + (1-1)\mathbf{j} + (1-3)\mathbf{k}] \times [(2-2)\mathbf{i} + (2-1)\mathbf{j} + (2-3)\mathbf{k}]$$

$$= \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 1 & 0 & -2 \\ 0 & 1 & -1 \end{vmatrix} = 2\mathbf{i} + \mathbf{j} + \mathbf{k}.$$

$$\mathbf{N}_{BAD} = \mathbf{BA} \times \mathbf{BD} = [(1-3)\mathbf{i} + (1-1)\mathbf{j} + (1-1)\mathbf{k}] \times [(2-3)\mathbf{i} + (2-1)\mathbf{j} + (2-1)\mathbf{k}]$$

$$= \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ -2 & 0 & 0 \\ -1 & 1 & 1 \end{vmatrix} = 0\mathbf{i} + 2\mathbf{j} - 2\mathbf{k}.$$

$$\mathbf{N}_{ACB} = \mathbf{AB} \times \mathbf{AC} = [(3 \quad 1)\mathbf{i} + (1 \quad 1)\mathbf{j} + (1 \quad 1)\mathbf{k}] \times [(2 \quad 1)\mathbf{i} + (1 \quad 1)\mathbf{j} + (3 \quad 1)\mathbf{k}]$$

$$= \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 2 & 0 & 0 \\ 1 & 0 & 2 \end{vmatrix} = 0\mathbf{i} - 4\mathbf{j} - 0\mathbf{k}.$$

.

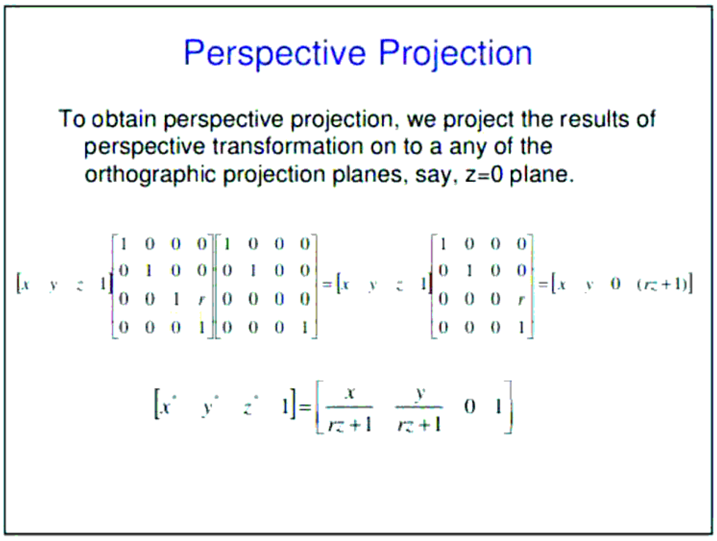
Q. 8 (B) EXPLAIN THE PERSPECTIVE PROJECTIONS WITH SUITABLE DIAGRAMS AND WRITE TRANSFORMATION MATRIX FOR THEM.

ANS.

Perspective Projection

In perspective projection farther away object from the viewer, small it appears. This property of projection gives an idea about depth. The artist use perspective projection from drawing three-dimensional scenes.

Two main characteristics of perspective are vanishing points and perspective foreshortening. Due to foreshortening object and lengths appear smaller from the center of projection. More we increase the distance from the center of projection, smaller will be the object appear.

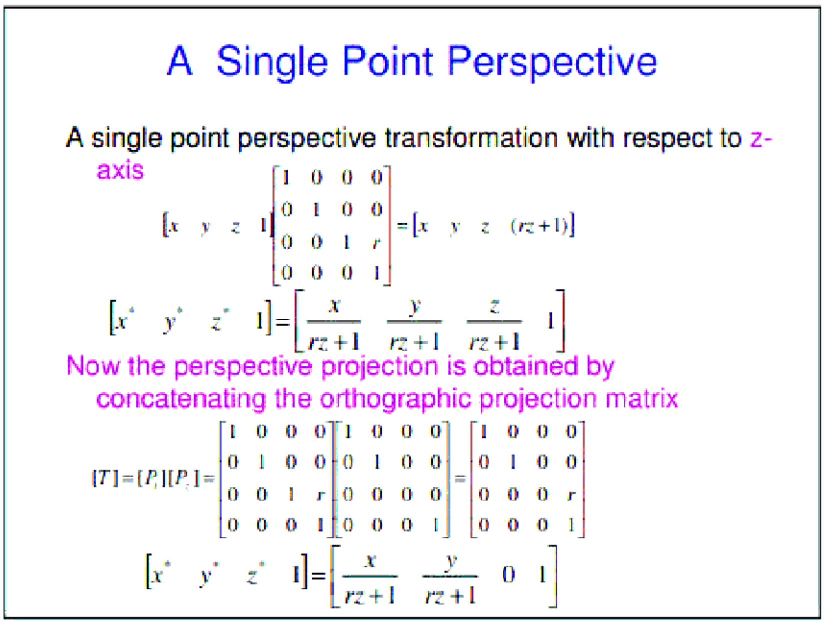


Perspective Projection

To obtain perspective projection, we project the results of perspective transformation on to a any of the orthographic projection planes, say, z=0 plane.

$$[x \ \ y \ \ z \ \ 1]\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&r\\0&0&0&1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&1\end{bmatrix}=[x \ \ y \ \ z \ \ 1]\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&r\\0&0&0&1\end{bmatrix}=[x \ \ y \ \ 0 \ \ (rz+1)]$$

$$[x^* \ \ y^* \ \ z^* \ \ 1]=\begin{bmatrix}\frac{x}{rz+1} & \frac{y}{rz+1} & 0 & 1\end{bmatrix}$$



A Single Point Perspective

A single point perspective transformation with respect to z-axis

$$[x \ \ y \ \ z \ \ 1]\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&r\\0&0&0&1\end{bmatrix}=[x \ \ y \ \ z \ \ (rz+1)]$$

$$[x^* \ \ y^* \ \ z^* \ \ 1]=\begin{bmatrix}\frac{x}{rz+1} & \frac{y}{rz+1} & \frac{z}{rz+1} & 1\end{bmatrix}$$

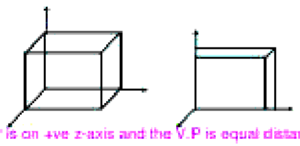
Now the perspective projection is obtained by concatenating the orthographic projection matrix

$$[T]=[P_r][P_z]=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&r\\0&0&0&1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&1\end{bmatrix}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&r\\0&0&0&1\end{bmatrix}$$

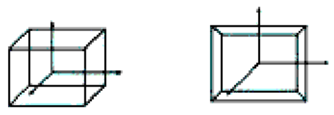
$$[x^* \ \ y^* \ \ z^* \ \ 1]=\begin{bmatrix}\frac{x}{rz+1} & \frac{y}{rz+1} & 0 & 1\end{bmatrix}$$

A Single Point Perspective

A single point perspective transformation with respect to z-axis



The COP is on +ve z-axis and the V.P is equal distance away on –ve z-axis



Perspective Projection of a centered cube

A Single Point Perspective

Numerical Example: A single point perspective transformation has to be performed on a unit cube from a center z_c=10 on the z-axis, followed by its projection on z=0 plane

Sol:Since z_c=10 , r=-1/ z_c= -1/10= -0.1

The perspective projection matrix for this problem can be written as

$$[U][T]=[U][P_r]=[U]\begin{bmatrix}1&0&0&0\\0&1&0&-0.1\\0&0&0&0\\0&0&0&1\end{bmatrix}$$

A Single Point Perspective

A single point perspective transformation with respect to x-axis and y-axis are given below respectively

$$[T]=[P_r][P_z]=\begin{bmatrix}1&0&0&r\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&1\end{bmatrix}=\begin{bmatrix}1&0&0&r\\0&1&0&0\\0&0&0&0\\0&0&0&1\end{bmatrix}$$

$$\begin{bmatrix}x^{*}&y^{*}&z^{*}&1\end{bmatrix}=\begin{bmatrix}\frac{x}{px+1}&\frac{y}{px+1}&0&1\end{bmatrix}$$

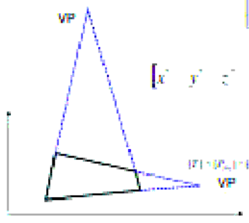
$$[T]=[P_r][P_z]=\begin{bmatrix}1&0&0&0\\0&1&0&q\\0&0&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&1\end{bmatrix}=\begin{bmatrix}1&0&0&0\\0&1&0&q\\0&0&0&0\\0&0&0&1\end{bmatrix}$$

$$\begin{bmatrix}x^{*}&y^{*}&z^{*}&1\end{bmatrix}=\begin{bmatrix}\frac{x}{qy+1}&\frac{y}{qy+1}&0&1\end{bmatrix}$$

Two Point Perspective Projection

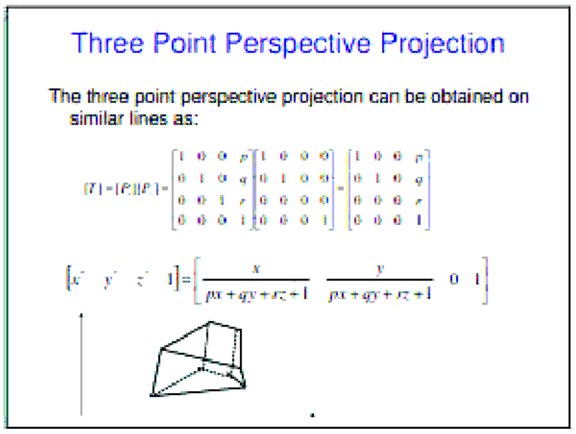
The two point perspective projection can directly be obtained on similar lines as:

$$[T]=[P_{pq}]=\begin{bmatrix}1&0&0&p\\0&1&0&q\\0&0&0&0\\0&0&0&1\end{bmatrix}=\begin{bmatrix}1&0&0&p\\0&1&0&q\\0&0&0&0\\0&0&0&1\end{bmatrix}$$



$$\begin{bmatrix}x^{*}&y^{*}&z^{*}&1\end{bmatrix}=\begin{bmatrix}\frac{x}{px+qy+1}&\frac{y}{px+qy+1}&0&1\end{bmatrix}$$

$$[T]=[P_{pq}]=[P_x][P_y]=\begin{bmatrix}1&0&0&p\\0&1&0&0\\0&0&0&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&q\\0&0&0&0\\0&0&0&1\end{bmatrix}=\begin{bmatrix}1&0&0&p\\0&1&0&q\\0&0&0&0\\0&0&0&1\end{bmatrix}$$

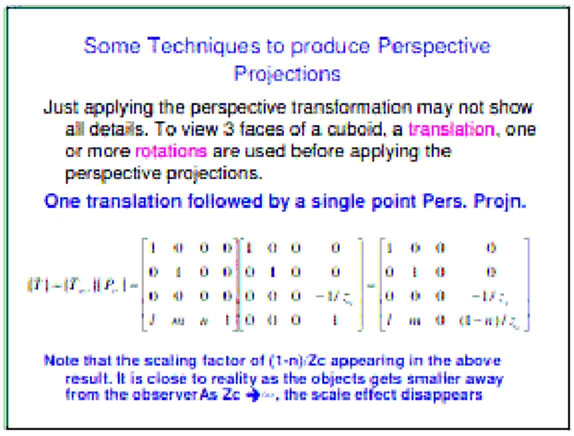


Three Point Perspective Projection

The three point perspective projection can be obtained on similar lines as:

$$[T] = [P_z][P_r] = \begin{bmatrix} 1 & 0 & 0 & p \\ 0 & 1 & 0 & q \\ 0 & 0 & 1 & r \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & p \\ 0 & 1 & 0 & q \\ 0 & 0 & 0 & r \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} \dfrac{x}{px+qy+rz+1} & \dfrac{y}{px+qy+rz+1} & 0 & 1 \end{bmatrix}$$



Some Techniques to produce Perspective Projections

Just applying the perspective transformation may not show all details. To view 3 faces of a cuboid, a translation, one or more rotations are used before applying the perspective projections.

One translation followed by a single point Pers. Projn.

$$[T] = [T_{tr}][P_z] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ l & m & n & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1/z_c \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1/z_c \\ l & m & 0 & (1-n)/z_c \end{bmatrix}$$

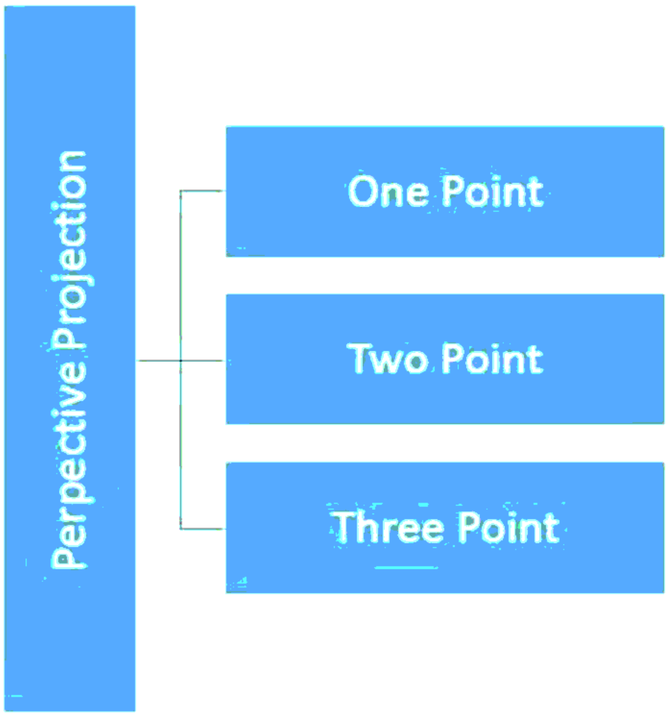
Note that the scaling factor of (1-n)/Zc appearing in the above result. It is close to reality as the objects gets smaller away from the observer As Zc → ∞, the scale effect disappears

Perspective Projection

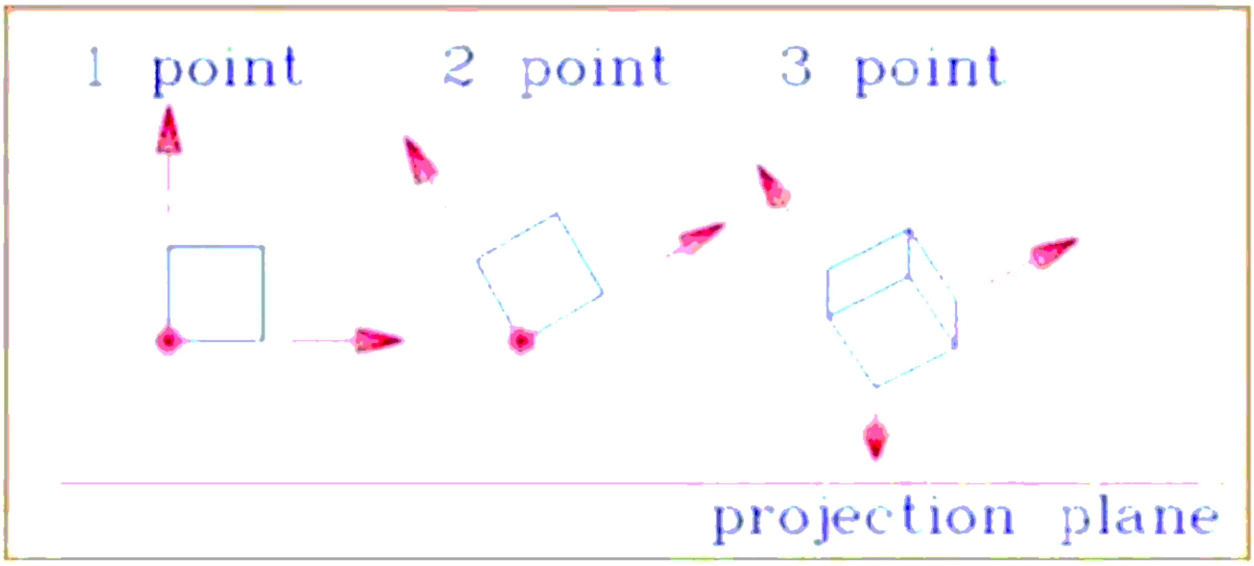
In perspective projection, the distance from the center of projection to project plane is finite and the size of the object varies inversely with distance which looks more realistic.

The distance and angles are not preserved and parallel lines do not remain parallel. Instead, they all converge at a single point called **center of projection** or **projection reference point**. There are 3 types of perspective projections which are shown in the following chart.

- **One point** perspective projection is simple to draw.
- **Two point** perspective projection gives better impression of depth.
- **Three point** perspective projection is most difficult to draw.



The following figure shows all the three types of perspective projection −

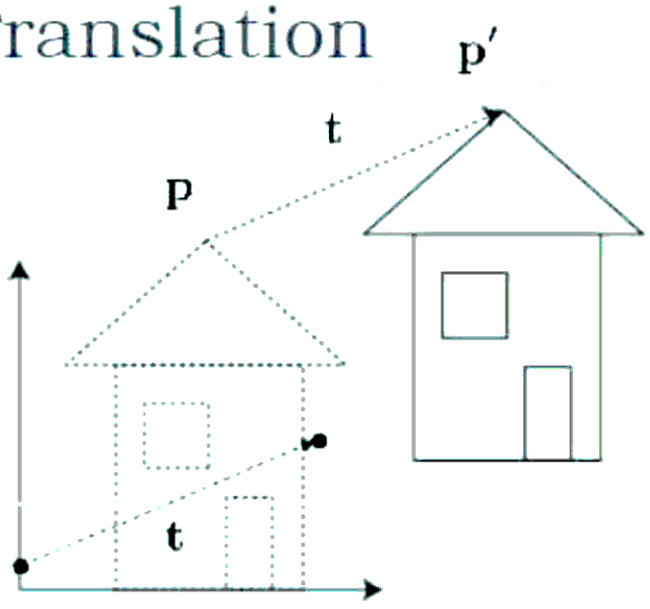


Translation

In 3D translation, we transfer the Z coordinate along with the X and Y coordinates. The process for translation in 3D is similar to 2D translation. A translation moves an object into a different position on the screen.

The following figure shows the effect of translation −

Translation



A point can be translated in 3D by adding translation coordinate (t_x, t_y, t_z) to the original coordinate X, Y, Z to get the new coordinate X', Y', Z' .

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix}$$

P' = P·T

$$[X' \ Y' \ Z' \ 1] = [X \ Y \ Z \ 1] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix}$$

$$= [X + t_x \quad Y + t_y \quad Z + t_z \quad 1]$$