

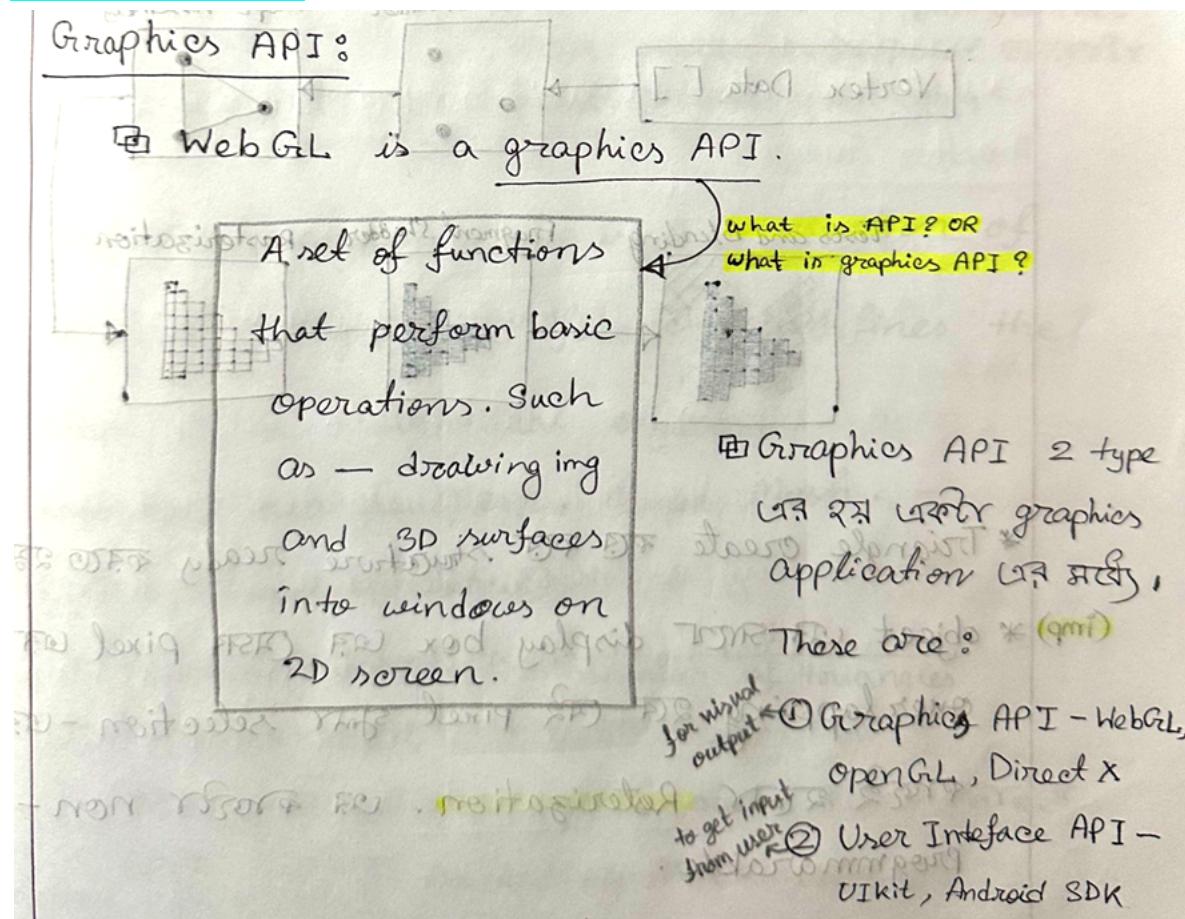
By Sowpnil Roy

Chapter 1

Graphics API

গ্রাফিক্স API (Application Programming Interface) হলো সফটওয়্যারে গ্রাফিক্স তৈরি বা প্রদর্শনের জন্য ব্যবহৃত একটি টুলকিট। এটি গ্রাফিক্স হার্ডওয়্যারের সাথে যোগাযোগের জন্য ব্যবহার করা হয়। উদাহরণস্বরূপ, WebGL, OpenGL বা Direct3D গ্রাফিক্স API এর উদাহরণ। এদের মাধ্যমে, প্রোগ্রামাররা গ্রাফিক্স কার্ডের ক্ষমতা ব্যবহার করে খ্রি-ডি অবজেক্ট রেন্ডার করতে পারেন।

A set of functions that perform basic Operations, Such as - Drawing and 3d surfaces into windows on 2d Screen



Graphics Pipeline (Quiz 3)

গ্রাফিক্স পাইপলাইন হলো একটি ধাপ-ভিত্তিক প্রক্রিয়া, যেখানে থ্রি-ডি মডেলকে টু-ডি স্ক্লিনে রেন্ডার করা হয়। এটি সাধারণত তিনটি প্রধান ধাপে ভাগ করা হয়:

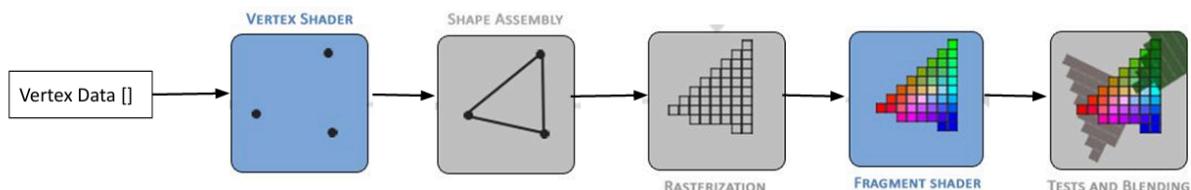
গ্রাফিক্স API তে, ভার্টেক্স ডেটা প্রথমে ভার্টেক্স শেডারে যায়। সেখানে, ভার্টেক্সের পজিশন এবং অন্যান্য বৈশিষ্ট্য প্রসেস করা হয়। এর পর, তা শেপ অ্যাসেম্বলিতে যায়, যেখানে ভের্টেক্সগুলোকে যোগ করে প্রিমিটিভ গঠন করা হয়।

এরপর আসে রাস্টারাইজেশন, যেখানে প্রিমিটিভগুলোকে পিক্সেলে রূপান্তর করা হয়। রাস্টারাইজেশনের পর, ডেটা যায় ফ্র্যাগমেন্ট শেডারে, যেখানে প্রতিটি পিক্সেলের রঙ এবং অন্যান্য বৈশিষ্ট্য নির্ধারিত হয়।

শেষে, টেস্ট অ্যান্ড রেন্ডারিং ধাপে, ফ্র্যাগমেন্টগুলোর মান যাচাই এবং প্রযোজনমত কস্বাইন করা হয়, যেমন আলফা রেন্ডিংয়ের জন্য। এইভাবেই, পুরো প্রক্রিয়া শেষে, থ্রি-ডি অবজেক্ট টু-ডি স্ক্লিনে প্রদর্শিত হয়।

■ **Graphics pipeline is a set of stages or processes that every graphics application goes through.**

6 Stages are:



(Imp) * object এর সাথে display box এর মেঝে pixel এর overlapping হলে মেঝে pixel এর selection - এর প্রক্রিয়া হলো **Rasterization**. এর ক্ষেত্রে non-programmable.

* Fragment Shader এর কাছে colour দিতে হবে।

Why we use Triangles?

Decipher 44
+
(Quiz)

- It is the simplest universal surface element. → line, points ~~বিন্দু~~ surface নয়।

It is the Convex Hull of 3 points.

→ ৩ points নিজে এমন একটা surface
তৈরি করা যাতে Points দ্বারা connect
করলে কোথায় একটা point আসেকো

যাতে কোথায় নাও হোস : যাতে দ্রুত প্রস্তাৱণা point connect কৰা যাব।

Rendering

রেন্ডারিং হল কম্পিউটারে শ্রি-ডি দৃশ্য তৈরি বা প্রদর্শনের প্রক্রিয়া। এটি বিভিন্ন ধাপে বিভক্ত, যেখানে শ্রি-ডি মডেল, আলো, ছায়া, টেক্সচার এবং ক্যামেরার মত উপাদানগুলো একত্রে কাজ করে। রেন্ডারিং এর ফলে আমরা ক্লিনে শ্রি-ডি অবজেক্টের ছবি বা অ্যানিমেশন দেখতে পাই।

রেন্ডারিংয়ের প্রক্রিয়াতে প্রথমে জ্যামিতিক হিসাব করে অবজেক্টের পজিশন নির্ধারণ করা হয়, তারপর আলো এবং ছায়ার হিসাব করে রং এবং উজ্জ্বলতা নির্ধারণ করা হয়।

Mesh

মেশ হলো একটি শ্রি-ডি অবজেক্টের জ্যামিতিক প্রতিনিষিদ্ধ, যা ভেটেক্স, এজ এবং ফেস দিয়ে গঠিত। এটি মূলত শ্রি-ডি অবজেক্টের বাহ্যিক পৃষ্ঠাগুলোকে সংজ্ঞায়িত করে। মেশের প্রাথমিক উপাদান ভেটেক্স, যা সংযোগ করে এড় তৈরি করে, এবং এড় গুলো সংযোগ করে ফেস তৈরি করা হয়। এই ফেসগুলো সাধারণত ত্রিভুজ বা চতুর্ভুজ আকারের হয়, যা একসাথে মিলে শ্রি-ডি অবজেক্টের আকৃতি গঠন করে।

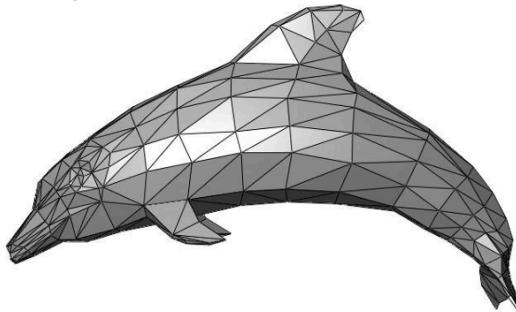
Mesh: A polygon mesh is a collection of vertices, edges and faces that defines the shape of a polyhedral object.

Example: Triangle Mesh, Quad Mesh.

- Mesh:

A polygon mesh is a collection of vertices, edges and faces that defines the shape of a polyhedral object.

– Ex. Quad mesh, Triangle mesh.



LOD

LOD, বা Level of Detail, হলো একটি প্রযুক্তি, যা থ্রি-ডি গ্রাফিক্সে ব্যবহৃত হয়, যাতে দৃশ্যমান অবজেক্টের জটিলতা বা বিশদ তথ্য, দেখার দূরস্থের ওপর ভিত্তি করে, কমানো বা বাড়ানো হয়।

নির্দিষ্ট করে বললে, **দূরে** থাকা অবজেক্টগুলোর জন্য কম পলিগন ও টেক্সচার রেজোলিউশন ব্যবহার করে কম্পিউটেশনের চাপ কমানো হয়, আর কাছাকাছি অবজেক্টগুলোর জন্য বেশি বিশদ তথ্য ব্যবহার করা হয়, যাতে মানসম্পন্ন রেন্ডারিং নির্ণিত করা যায়।

2) LOD: (Level of Detail)

- Rendering Speed $\propto \frac{1}{\text{number of triangles}}$
- more triangle: more storage
- কাছের জিনিস ক্রিয়া detail show করে। অন্তর্ভুক্ত
দূরের জিনিস detailing করতে পারে।

Chapter 2

Raster Image

রাস্টার ইমেজ হলো একটি পিক্সেলভিত্তিক চিত্র। এতে অসংখ্য পিক্সেল বা ডট থাকে, এবং প্রতিটি পিক্সেলের একটি নির্দিষ্ট রং থাকে। এই পিক্সেলগুলো একত্রে চিত্র তৈরি করে। রাস্টার ইমেজের মান নির্ভর করে এর রেজোলিউশনের ওপর, অর্থাৎ পিক্সেলের সংখ্যা এবং ঘনত্বের ওপর। উদাহরণ হিসেবে, JPEG বা PNG ফর্ম্যাটের ছবি হল রাস্টার ইমেজ। রাস্টার ইমেজকে বড় করলে এর পিক্সেলগুলো দৃশ্যমান হয়, ফলে ছবির মান কমে যেতে পারে।

- **Raster Image:**

- used to store and process images, as rasters are common in devices

- simply a **2D array**
- stores the pixel value for each pixel
- usually a color stored as **three numbers (r, g, b)**

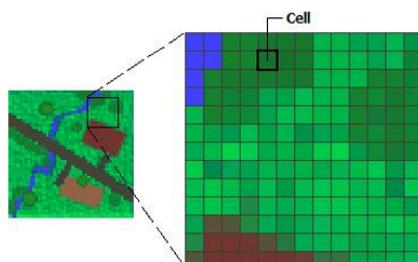


Image Compression

ইমেজ কম্প্রেশন হল ছবি বা গ্রাফিকসের সাইজ কমানোর প্রক্রিয়া, যাতে কম্পিউটার স্টোরেজ এবং ডেটা ট্রান্সফার সহজ হয়।

এই প্রক্রিয়া দুইভাবে করা যায়:

- লুজি কম্প্রেশন: এতে কিছু তথ্য বাদ দেওয়া হয়, ফলে ইমেজের গুণগত মান কিছুটা কমে যায়, কিন্তু সাইজ বেশ কমে যায়। যেমন JPEG।
- লসলেস কম্প্রেশন: এতে কোনো তথ্য বাদ দেওয়া হয় না, তাই ইমেজের গুণগত মান অপরিবর্তিত থাকে, কিন্তু সাইজ কমানো সম্ভব হয়। যেমন PNG।

এই দুই ধরনের কম্প্রেশন ইমেজের উদ্দেশ্য এবং প্রয়োগের ওপর নির্ভর করে ব্যবহার করা হয়।

Lossless Compression

- Don't impact the image quality
- Only removes additional, non-essential data automatically added by the device used to take the photo
- No significant reduction in file size
- Lossless Compression algorithms: Run-length encoding, Huffman coding
- Lossless formats are .RAW, .BMP, .GIF, and .PNG

Lossy Compression

- Reduces the file size considerably by removing image data
- Quality might degrade
- This process is irreversible - can't get back to the original file
- Common algorithms - discrete wavelet transform, fractal compression, transform encryption etc.
- Lossy format; JPEG, MPEG, AVC

Difference Between Lossless & Lossy Compression

Difference between Lossless and Lossy :

(Quiz) + Deciphering

Lossless Compression	Lossy Compression
1. get back original file.	1. can not get back original file.
2. Quality does not decrease	2. Quality might decrease.
3. Algorithms: RLE, Huffman coding.	3. Algorithms: transform encryption, fractal compression etc.
4. Formats: .RAW, .PNG, .GIF	4. Formats: JPEG, AVI, MPEG
5. File size doesn't reduce significantly	5. File size reduce significantly.

How RLE (Run Length Encoding) Works?

How RLE (Run-Length Encoding) works?

→  → 4x5 image

After compressed → Here,

→ if 1 pixel = 8 bit
= 1 byte
then it's a $(4 \times 5 \times 1) = 20$ byte image

→ 4x3 image
→ It's a $(4 \times 3 \times 1) = 12$ byte image.

2	2	1
1	3	1
1	1	3
3	1	1

2R 2W 1B
1W 3B 1W
1B 1W 3R
3W 1B 1R
(W=White, G=Grey, B=Black)

Transmissive display

Transmissive display এমন একটি ডিসপ্লে প্রযুক্তি যেখানে ব্যাকলাইটের আলো সরাসরি প্যানেলের মধ্য দিয়ে পাস করে এবং ডিসপ্লেতে দৃশ্যমান ইমেজ তৈরি করে। এটি সাধারণত LCD (Liquid Crystal Display) স্ক্রিনে ব্যবহৃত হয়।

এখানে LCD প্যানেল নিজে আলো তৈরি করতে পারে না, তাই ব্যাকলাইট থেকে আসা আলো ক্রিস্টালের মধ্য দিয়ে পাস করে রঙ এবং চিত্র তৈরি করে। এটি বাইরে বা উচ্চতার আলোর পরিবেশে কিছুটা কম স্পষ্ট হতে পারে, কারণ ব্যাকলাইটের আলো পরিবেশের আলোতে মিশে যেতে পারে।

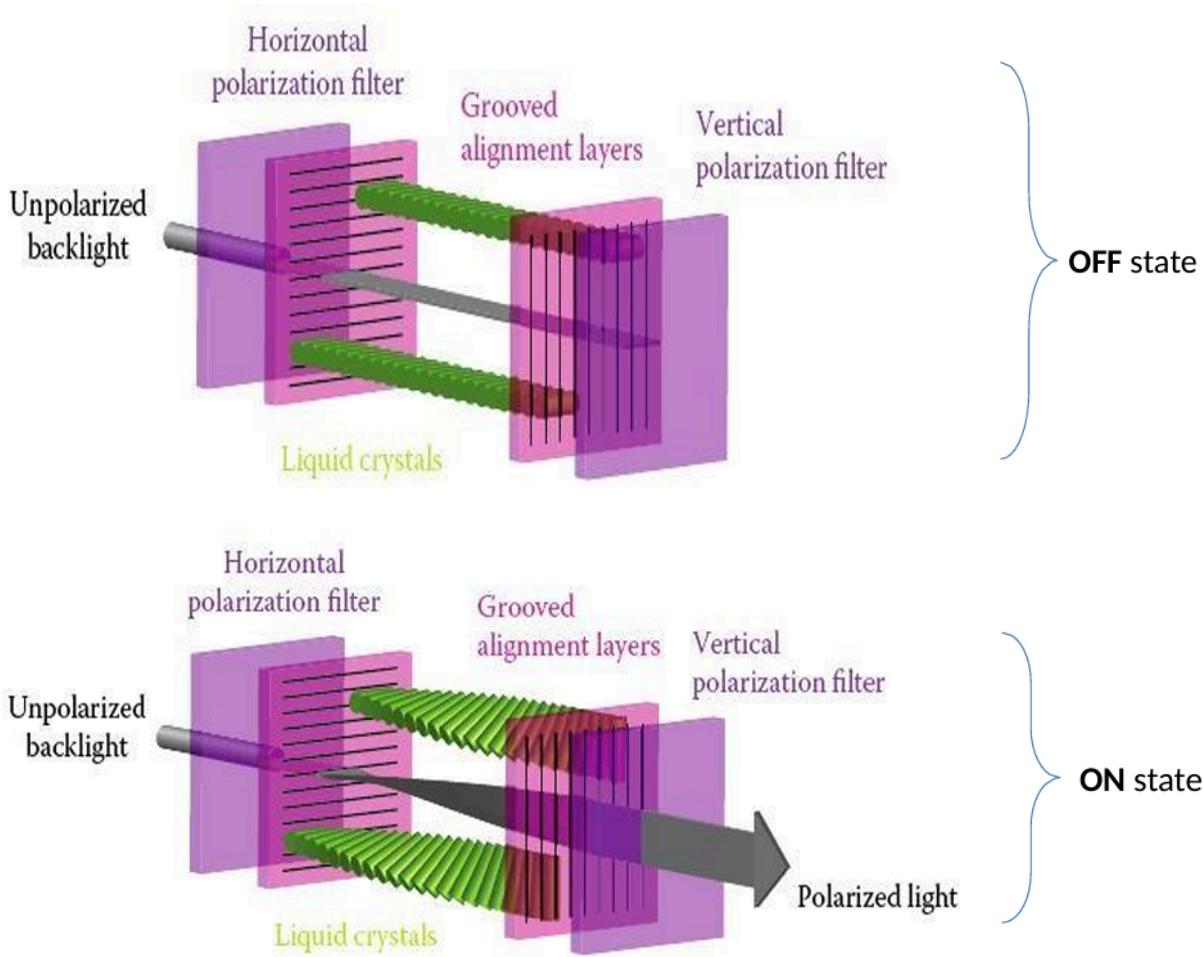
Quiz + Decipher 44

Emissive display

Emissive display এমন একটি প্রযুক্তি যেখানে প্রতিটি পিক্সেল নিজেই আলো উৎপন্ন করে এবং এর জন্য আলাদা ব্যাকলাইটের প্রয়োজন হয় না। এই ধরনের ডিসপ্লেতে, পিক্সেলগুলো নিজেদের আলোর মাধ্যমে ইমেজ তৈরি করে, ফলে উচ্চতা এবং কন্ট্রাস্ট অনেক ভালো হয়।

Emissive display-এর উদাহরণ হলো OLED (Organic Light Emitting Diode) এবং MicroLED ডিসপ্লে। এই ডিসপ্লে প্রযুক্তি সাধারণত উচ্চ কন্ট্রাস্ট এবং গভীর কালো রঙ প্রদর্শনে সক্ষম, কারণ পিক্সেলগুলোকে স্বাধীনভাবে চালু বা বন্ধ করা যায়।

Feature	Transmissive Display	Emissive Display
Light Source	Comes from a backlight	Pixels produce their own light
Examples	LCD (Liquid Crystal Display)	OLED, MicroLED
Contrast	Lower contrast due to uniform backlighting	High contrast; pixels can individually turn on/off
Black Level Depth	Less deep blacks	Can display deep blacks
Visibility in Bright Environments	Lower, especially outdoors or under bright light	Generally better visibility
Power Consumption	Higher, as the backlight is always on	Lower, as individual pixels can be off or on
Image Quality	Less vibrant colors	Bright and vibrant colors



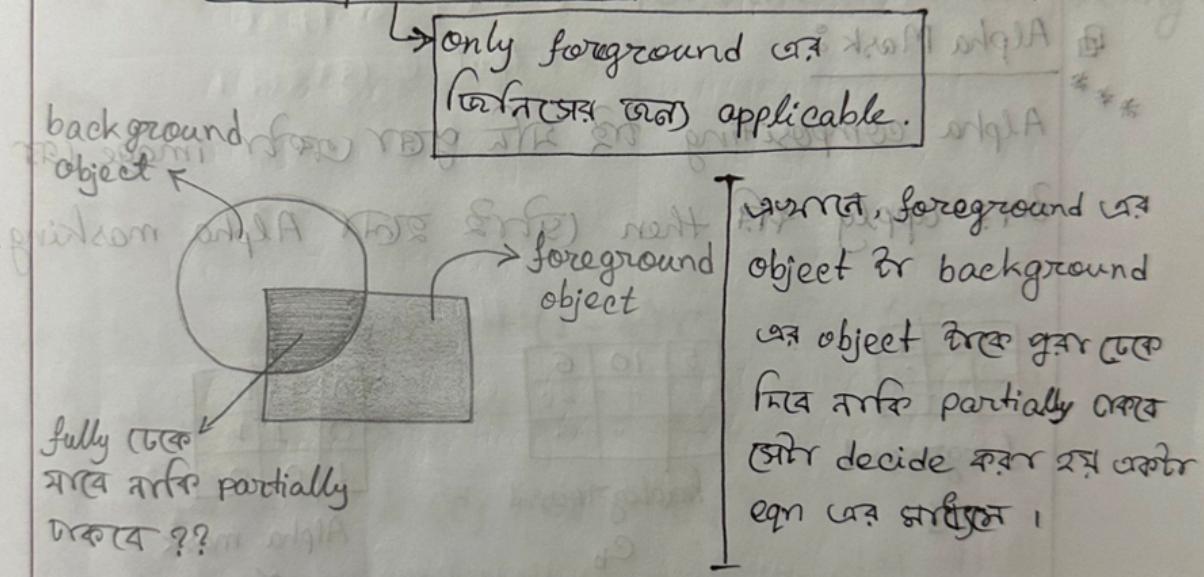
Alpha Compositing

Alpha Compositing হল গ্রাফিক্সের একটি পদ্ধতি, যেখানে বিভিন্ন লেয়ারের ছবি একসাথে মেশানো হয়। "Alpha" চ্যানেল স্বচ্ছতা (transparency) নির্দেশ করে—যেখানে 0 মানে সম্পূর্ণ স্বচ্ছ এবং 1 মানে সম্পূর্ণ অস্বচ্ছ।

Alpha Compositing

↳ Partially overwriting the contents of a pixel.

(foreground एवं जिन्हें background एवं जिन्हें
background पूरी तरह छोड़ना नहीं ब्रॉडकास्ट करता है बल्कि वह अपने
transparent issue, यह इस विषय पर विवरण देता है।)



Equation:

$$C = \alpha C_f + (1 - \alpha) C_b$$

output
color

foreground
color

background
color

α = Fraction of the pixel covered
by the foreground layer

transparency
(scalar value)

$$\text{if, } C_f = \text{Red} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$C_b = \text{Green} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\alpha = 0.5 \quad [\text{Ques. - 2 दूसरा थार्ड्वेयर}]$$

$$\begin{aligned} \therefore C &= \alpha C_f + (1 - \alpha) C_b \\ &= 0.5 \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + (1 - 0.5) \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} 0.5 \\ 0.5 \\ 0 \end{bmatrix} \rightarrow \text{red, green वा combine उत्तरी color. (Ans).} \end{aligned}$$

Alpha Mask

Alpha Mask

Alpha compositing නිසු මත යුතු ඇත්තා image නේ
background apply කරී then සැකි හැනු Alpha masking.

5	10	6
-	-	-
-	-	-

background

$C_f = \alpha f + (1 - \alpha) C_b$

0	1	1
0	1	1
1	0	0

Alpha mask α

calculation
5th element
mix RGA

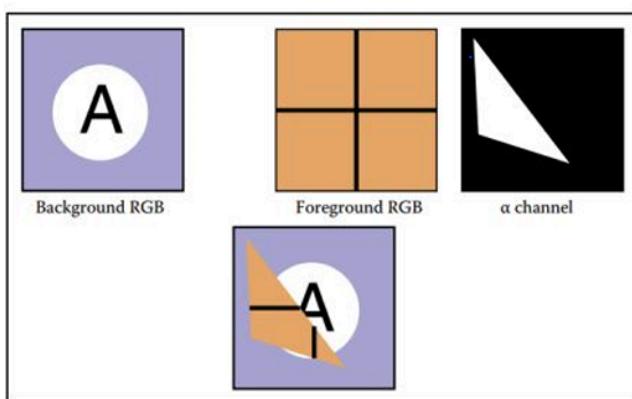
$$C = \alpha f + (1 - \alpha) C_b$$

$$= \begin{bmatrix} \checkmark \\ \alpha \end{bmatrix} \cdot \begin{bmatrix} 1 \\ C_f \end{bmatrix} + \begin{bmatrix} \checkmark \\ 1 - \alpha \end{bmatrix} \begin{bmatrix} 1 \\ C_b \end{bmatrix}$$

$$= \begin{bmatrix} \checkmark \\ \text{sum of foreground} \end{bmatrix} \rightarrow C$$

Alpha Mask:

- The α values for all the pixels is stored in a separate gray scale image.



Chapter 3

Vector Image

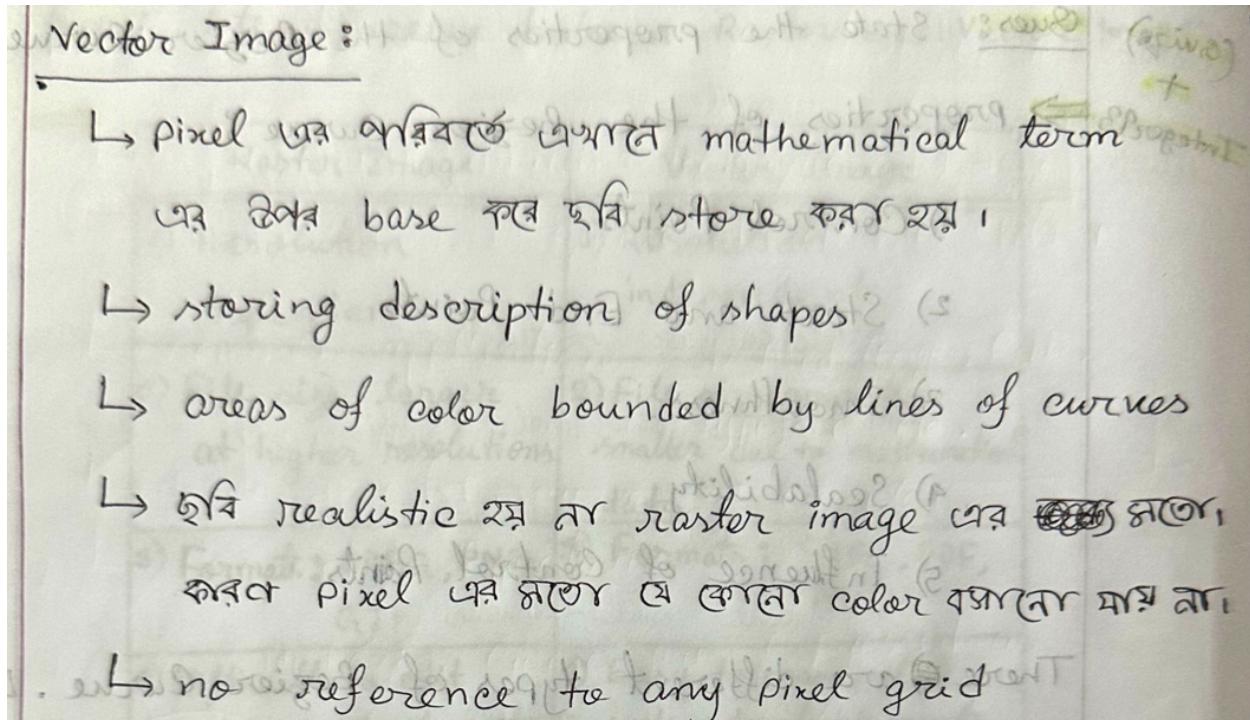
Vector Image হলো একটি ধরণের গ্রাফিক্স ফাইল, যা গাণিতিক সমীকরণের মাধ্যমে তৈরি হয়। এতে ছবির বিভিন্ন উপাদান (যেমন লাইন, শেপ, এবং পয়েন্ট) ভেক্টর হিসেবে সংরক্ষিত থাকে। এই কারণেই, Vector Image বড় বা ছোট করলেও এর গুণগত মান অপরিবর্তিত থাকে, মানে এটি রেজোলিউশন-স্বাধীন।

Vector Image-এর বৈশিষ্ট্য:

- **স্কেলেবল:** যেকোনো সাইজে বাড়ানো বা কমানো যায়, কিন্তু মান কমে না।
- **কম ফাইল সাইজ:** সাধারণত ছোট ফাইল সাইজ, কারণ পিক্সেলের বদলে গাণিতিক ফর্মুলায় তথ্য সংরক্ষণ করা হয়।
- **সম্পাদনাযোগ্য:** সহজে রঙ, আকৃতি, এবং আকার পরিবর্তন করা যায়।

উদাহরণ:

Vector Image সাধারণত লোগো, আইকন, এবং ইলাস্ট্রেশন তৈরিতে ব্যবহৃত হয়। এর ফাইল ফর্ম্যাটের মধ্যে SVG, AI, এবং EPS অন্যতম।



Properties of vector Image:

④ Properties of Vector Image:

1. Resolution independent - scaling the image will not affect the quality.
2. Typically small in size compared to raster image.
3. Generates smooth curves and edges.
4. Limited photorealism.
5. Must be rasterized before they can be displayed.
6. File Formats : SVG, AI, PDF.

What is a Curve?

④ What is a Curve ?

- ⇒ - Curve is the continuous image of some interval in an n-dimensional space
- a continuous map from a one dimensional space to an n-dimensional space.
- it lies on 2D plane, but actually, it's 1D.

Differences between raster and vector images?

Raster Image	Vector Image
1) Resolution dependent	1) Resolution independent
2) File size larger at higher resolutions	2) File size generally smaller due to mathematical data.
3) Formats: JPEG, PNG, GIF	3) Formats: SVG, PDF, AI
4) Difficult to edit individual elements	4) Easy to manipulate individual elements
5) Best for photographs, complex images	5) Best for Logos, icons, scalable illustrations

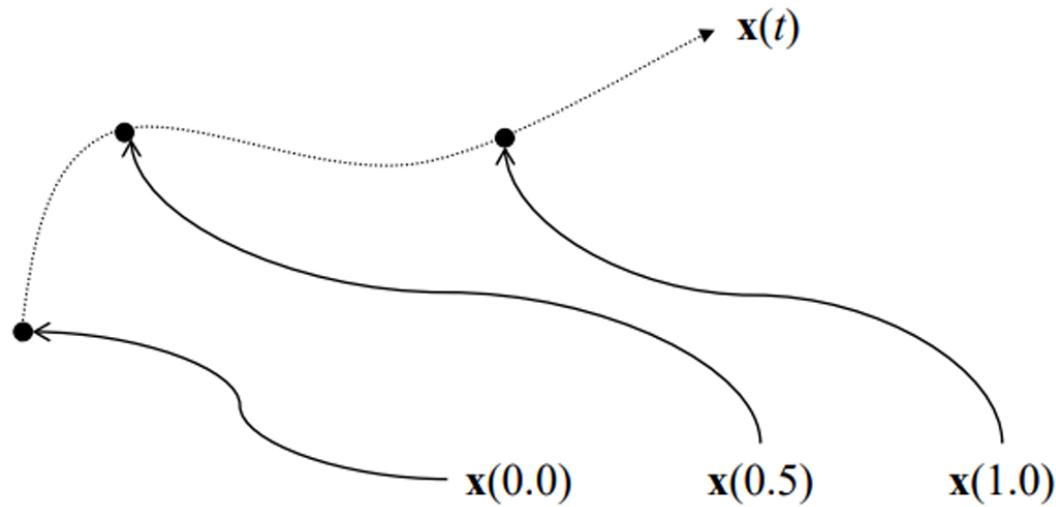
Feature	Raster Image	Vector Image
Data Representation	Made up of pixels	Made up of mathematical equations
Resolution Dependence	Resolution-dependent; loses quality when enlarged	Resolution-independent; retains quality when scaled
Scalability	Quality decreases when scaled up	Can be scaled to any size without loss of quality
File Size	Usually larger, as it stores each pixel	Usually smaller, as it stores mathematical data
Usage	Photos, web images, detailed graphics	Logos, icons, illustrations, and text-based graphics
File Formats	JPEG, PNG, BMP, GIF	SVG, AI, EPS, PDF

Curves

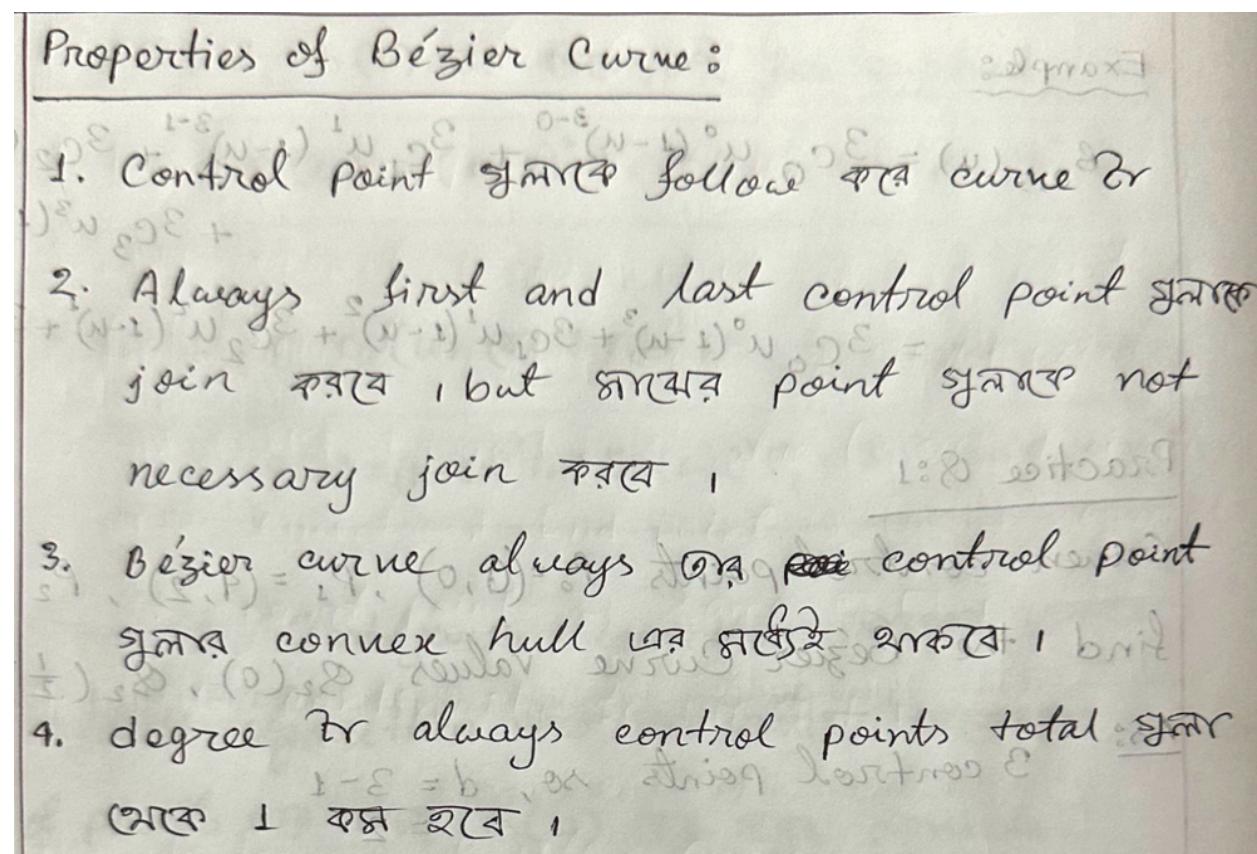
- Curve is the continuous image of some interval in an n-dimensional space
- a continuous map from a one-dimensional space to an n-dimensional space.

How many dimensions in the curve?

- It lies on 2D plane, but actually it's 1D



Properties of Bezier Curves

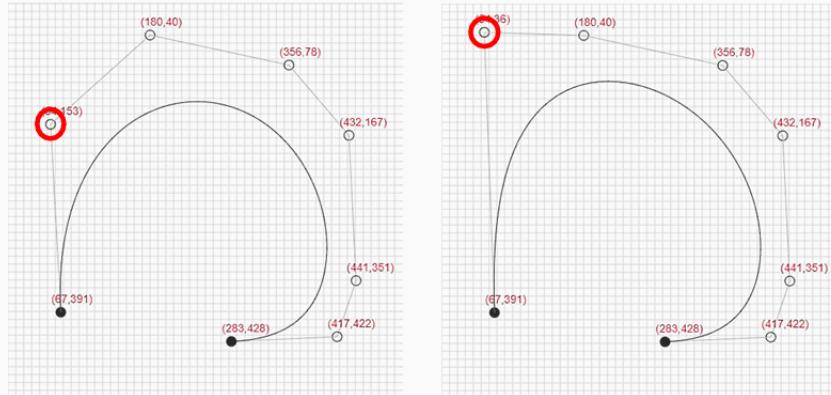


Properties of Bezier Curves

- They generally follow the shape of the control polygon, which consists of the segments joining the control points
- They always pass through the first and last control points
- They are contained in the convex hull of their defining control points
- The degree of the polynomial defining the curve segment (d) is one less than that the number of defining polygon point (n) i.e. $n = d+1$

Disadvantages

- A change to any of the control point alters the entire curve.
- Having a large number of control points requires high polynomials to be evaluated. This is expensive to compute.



Disadvantages:

1. Point শূলো নিয়ে হত্তয়া করে কর্তৃপক্ষের curve
এবং shape পুরু নয়। ইষ্টে এমন,
2. Higher number of control points করে compute

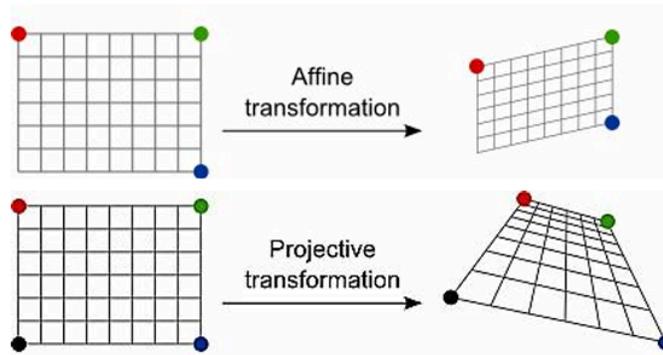
Bézier Curve এর অসম্ভাব্য disadvantages = শূলোর
solution হলো B-Spline Curve

Chapter 4

Affine transformation

Affine transformation (1/2)

- Maps points to points, lines to lines, planes to planes.
- Preserves the ratio of lengths of parallel line segments.
- Sets of parallel lines remain parallel.
- Does not necessarily preserve angles between lines or distances between points.



Chapter 5

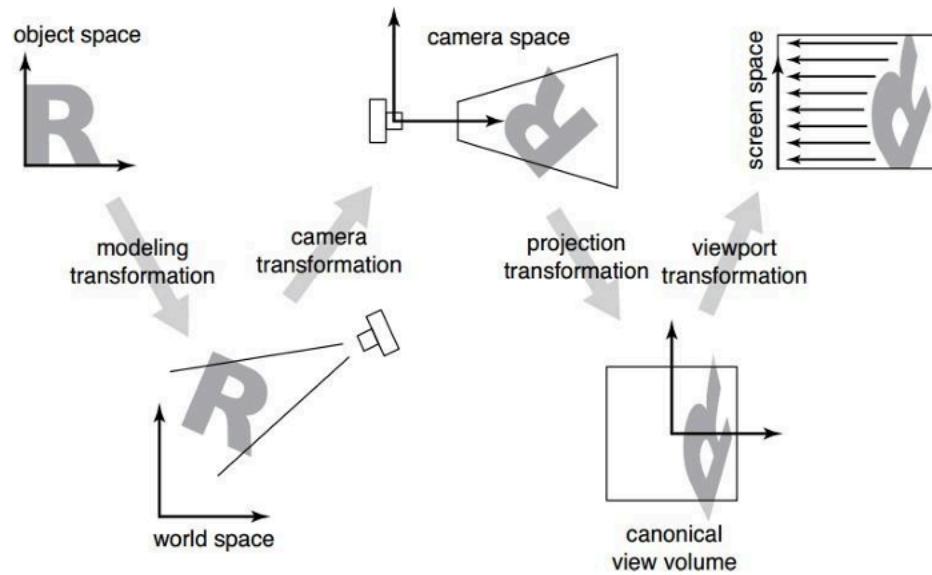
Rendering Techniques

Rendering Techniques (1/2)

- One of the basic tasks of computer graphics is rendering 3D objects:
 - taking a scene, or model, composed of many geometric objects arranged in 3D space
 - producing a 2D image that shows the objects as viewed
 - from a particular viewpoint.
- 1. Image-order rendering: iterate over the pixels in the image to be produced, rather than the elements in the scene to be rendered.
- 2. object-order rendering: that iterate over the elements in the scene to be rendered, rather than the pixels in the image to be produced.
- Image-order rendering:
 - **Ray-tracing**:
For each pixel is considered in turn,
 - All the objects that influence it are found
 - and the pixel value is computed.
- Object-order rendering:
 - **Viewing Transformation**:
For each object is considered in turn,
All the pixels that it influences are found and updated

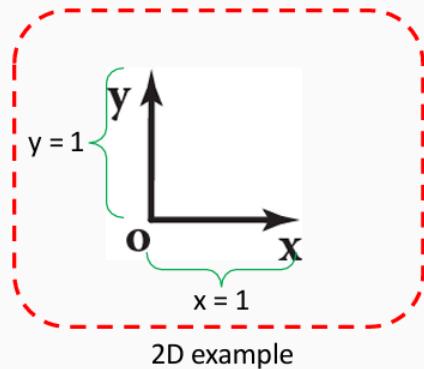
- **Viewing Transformation:**

- How to use ***matrix transformations*** to express any parallel or perspective view.
- These transformations:
 - Project 3D points in the scene (world space) to 2D points in the image (image space)



Coordinate System (1/1)

- A coordinate system, or coordinate frame, consists of **an origin** and **a basis**: *a set of three (two for 2D) orthonormal vectors.*



Canonical coordinate system:

- origin **o**
- orthonormal basis vectors **{x, y}**.
- Also called: *World coordinates*

Practice Problem - 2

- Explain with appropriate example that the frame-to-canonical transformation can be expressed as a rotation followed by a translation.
- Explain with appropriate example that canonical-to-frame transformation is a translation followed by a rotation; they are the inverses of the rotation and translation we used to build the frame-to-canonical matrix.
 - Hint: **Fundamentals of Computer Graphics, Section 6.5**

Solution:

Part (a): Frame-to-Canonical Transformation as Rotation Followed by Translation

When converting from a new frame (camera or observer's perspective) to the canonical (original) coordinate system, we first need to rotate the frame's coordinates to align with the canonical axes and then translate them to match the canonical origin.

Example:

Suppose you have a coordinate frame positioned at (x_e, y_e) with an orientation angle θ relative to the canonical frame.

Example:

Suppose you have a coordinate frame positioned at (x_e, y_e) with an orientation angle θ relative to the canonical frame.

1. **Rotation:** To align the frame's axes with the canonical axes, apply a rotation matrix that rotates the frame by $-\theta$ (to counter the frame's angle). This ensures that the frame's axes match the canonical ones.

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

2. **Translation:** After rotating, apply a translation to shift the origin from (x_e, y_e) to the canonical origin $(0, 0)$. The translation matrix would look like:

$$T = \begin{bmatrix} 1 & 0 & -x_e \\ 0 & 1 & -y_e \\ 0 & 0 & 1 \end{bmatrix}$$

3. **Combined Transformation:** The frame-to-canonical transformation is thus a combination of rotation followed by translation:

$$\text{Frame-to-Canonical} = T \times R$$

This transformation allows us to view points in the new frame from the perspective of the canonical coordinate system by first aligning the axes and then positioning them to match the canonical origin.

Part (b): Canonical-to-Frame Transformation as Translation Followed by Rotation

The canonical-to-frame transformation essentially reverses the operations from the previous part. It requires shifting the canonical coordinates to the frame's origin, followed by a rotation to align with the frame's orientation.

Example:

Given the canonical coordinates (x, y) , and a frame located at (x_e, y_e) with an angle θ :

1. **Translation:** First, translate the canonical coordinates to center around the frame's origin. This can be achieved using the following translation matrix:

$$T' = \begin{bmatrix} 1 & 0 & x_e \\ 0 & 1 & y_e \\ 0 & 0 & 1 \end{bmatrix}$$

2. **Rotation:** After translation, apply a rotation by θ to match the frame's orientation:

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

3. **Combined Transformation:** The canonical-to-frame transformation is the translation followed by rotation:

$$\text{Canonical-to-Frame} = R' \times T'$$

This approach effectively converts canonical coordinates into the frame's perspective by first shifting the origin and then aligning the orientation to match the frame's view.

Chapter 6

Rasterization

Rasterization হলো এমন একটি পদ্ধতি, যার মাধ্যমে কম্পিউটার গ্রাফিক্সে 3D অবজেক্টকে 2D স্ক্রিনে দেখানোর জন্য পিঙ্গেলে রূপান্তরিত করা হয়। এটি কম্পিউটার গ্রাফিক্সে একটি গুরুত্বপূর্ণ প্রক্রিয়া, বিশেষ করে রিয়েল-টাইম রেন্ডারিং (যেমন গেমসে) ব্যবহৃত হয়।

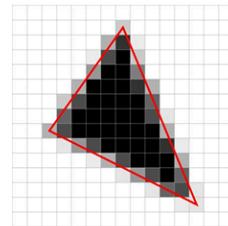
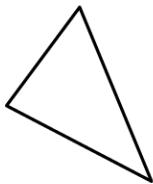
কিভাবে Rasterization কাজ করে:

1. প্রথমে 3D অবজেক্টের প্রতিটি পয়েন্ট বা গোলকে 2D স্ক্রিনে ম্যাপ করা হয়।
2. এরপর, পয়েন্টগুলোকে পিঙ্গেলে কনভার্ট করা হয়, যেখানে প্রতিটি পিঙ্গেল স্ক্রিনের একটি নির্দিষ্ট রঙ ধারণ করে।
3. এই প্রক্রিয়ায় পিঙ্গেলের মাল নির্ধারণ করে পুরো ছবি বা অবজেক্টটি স্ক্রিনে ফুটিয়ে তোলা হয়।

সংক্ষেপে বলা যায়: Rasterization হলো 3D অবজেক্টকে 2D পিঙ্গেলের গিডে রূপান্তর করার পদ্ধতি, যা পর্দায় অবজেক্ট দেখানোর জন্য অত্যন্ত গুরুত্বপূর্ণ।

Rasterization (1/2)

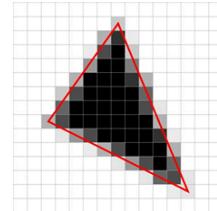
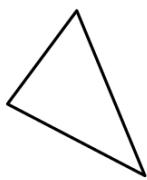
- The previous several chapters have established the mathematical skeleton for object-order rendering.
 - drawing objects one by one onto the screen
- Each geometric object is considered in turn and find the pixels that it could have an effect on.
- The process of finding all the pixels in an image that are occupied by a geometric primitive is called **rasterization**.



Graphics Pipeline

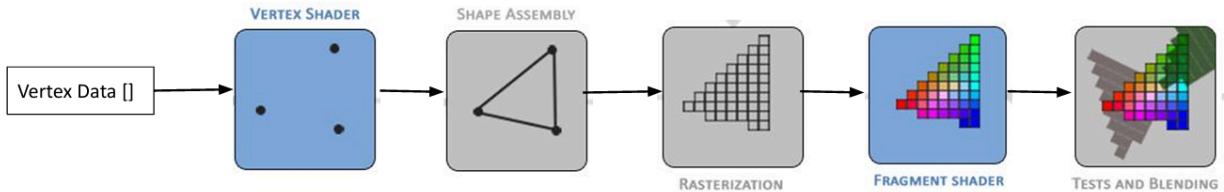
Graphics Pipeline (1/5)

- The sequence of operations that is required, starting with objects and ending by updating pixels in the image, is known as the **graphics pipeline**.



- Two quite different examples of graphics pipelines with very different goals are the
 - hardware pipelines used to support interactive rendering via APIs like OpenGL and Direct3D
 - the software pipelines used in film production, supporting APIs like *RenderMan* (by Pixar).

- Hardware pipelines:
 - run fast enough to react in real time for games, visualizations, and user interfaces.
- Software pipelines:
 - render the highest quality animation and visual effects possible and scale to enormous scenes
 - but take much more time to do so
- Remarkable amount is shared among most pipelines
- This lecture attempts to focus on these common fundamentals



What is Barycentric Coordinate?

Or

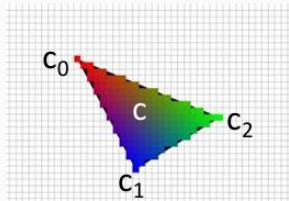
What is “Gouraud Interpolation”?

Soln:

Gouraud Interpolation বা Gouraud Shading হলো এমন একটি পদ্ধতি, যেখানে 3D অবজেক্টের পৃষ্ঠে আলো এবং রঙের ধীরে ধীরে পরিবর্তন দেখানো হয়। এর মাধ্যমে অবজেক্টের ওপর আলো যেন মসৃণভাবে ছড়িয়ে পড়েছে, সেটি বোঝানো যায়। এখানে প্রতিটি কোণায় (vertex) আলোর প্রভাব অনুযায়ী রঙ নির্ধারণ করা হয় এবং এরপর কোণ থেকে কোণ পর্যন্ত মসৃণভাবে রঙ মিশিয়ে (interpolate) পুরো পৃষ্ঠে ছড়িয়ে দেওয়া হয়।

- If the vertices have colors c_0 , c_1 , and c_2 , the color at a point in the triangle with *Barycentric coordinates* (α , β , γ) is:

$$\mathbf{c} = \alpha\mathbf{c}_0 + \beta\mathbf{c}_1 + \gamma\mathbf{c}_2$$



- This type of interpolation of color is known in graphics as **Gouraud interpolation**

Bresenham's Circle Drawing Algorithm :

- b) Write down the algorithm to create a half circle given the radius and the center using Bresenham's Circle drawing algorithm. [6]

Algorithm

```
void MidpointCircle(int radius)
{
    int x = 0;
    int y = radius ;
    int d = 1 - radius ;
    CirclePoints(x,y);
    while (y > x)
    {
        if (d < 0) /* SelectE*/
            d = d + 2 * x + 3;
        else
        { /* SelectSE*/
            d = d + 2 * (x - y) + 5;
            y = y - 1;
        }
        x = x + 1;
        CirclePoints(x,y);
    }
}
```

```
CirclePoints (x,y)
    Plotpoint(x,y) ;
    Plotpoint (x,-y);
    Plotpoint(-x,y) ;
    Plotpoint(-x, -y);
    Plotpoint(y,x) ;
    Plotpoint(y, -x);
    Plotpoint(-y, x);
    Plotpoint( -y, -x);
end
```

Chapter 8

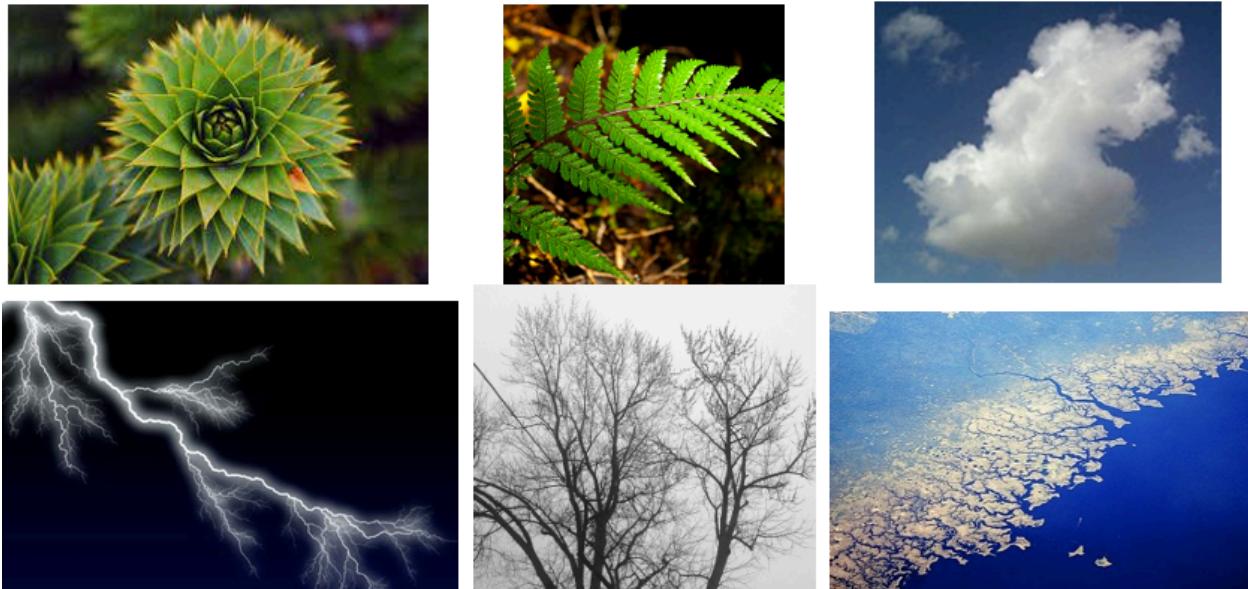
Fractal

Fractal হল একটি জ্যামিতিক আকৃতি যা একই প্যাটার্ন বারবার পুনরাবৃত্তি করে তৈরি হয়। এর প্রধান বৈশিষ্ট্য হলো **Self-similarity**, অর্থাৎ ছোট বা বড় যে কোনো অংশ দেখলে মূল আকৃতির সাথে মিল পাওয়া যায়। Fractal আকৃতি প্রাকৃতিক দৃশ্যে অনেক সময় দেখা যায়, যেমন গাছের ডালপালা, তুষারকণা, পাহাড়ের পাথর, মেঘ, এবং সমুদ্রের তেওঁ।

Fractal

- A **Fractal** is a geometric shape generated using set of recursive rules
- Fractals share a **self-similarity** property where part of fractal resembles the whole fractal
- The property is often seen in nature, e.g. clouds, plants and landscapes etc.
- We will look at methods used to generate some popular fractals in this lecture

Example of Fractals



L-Systems

L-System (Lindenmayer System) হলো একটি নিয়মভিত্তিক পদ্ধতি, যা পুনরাবৃত্তির মাধ্যমে জ্যামিতিক আকার তৈরি করে। এটি প্রাথমিকভাবে গাছের শাখা-বিস্তারের মতো প্রাকৃতিক গঠন মডেল করতে ব্যবহৃত হয়।
মূল উপাদান:

Alphabet: প্রতীকগুলোর সেট (যেমন F, +, -)

Axiom: শুরুর স্ট্রিং বা বিন্দু।

Production Rules: প্রতিটি প্রতীক কীভাবে রূপান্তরিত হবে, সেই নিয়ম।

L-systems

- Lindenmayer-systems or L-system is developed by 1968 by biologist Aristid Lindenmayer
 - It's a grammar based technique
 - Represent shape as string of symbol
 - Each symbol has meaning in drawing shape
 - Two parts:
 - Grammar for generating strings
 - Rendering algorithm for interpreting strings as shapes
-

L-system structure

L-system structure

- L-systems are commonly known as parametric L systems, defined as a tuple

$$\mathbf{G} = (V, \omega, P)$$

- **V (the alphabet)** is a set of symbols containing both elements that can be replaced (variables) and those which cannot be replaced ("constants" or "terminals")
 - **ω (start, axiom or initiator)** is a string of symbols from V defining the initial state of the system
 - **P is a set of production rules** or productions defining the way variables can be replaced with combinations of constants and other variables.
-

Semester Final Question Solve

State the differences between hardware and software pipelines.

Feature	Hardware Pipeline	Software Pipeline
Type	Physical components (CPU, GPU, etc.)	Program-based (code and algorithms)
Speed	Faster processing	Slower processing
Task Flexibility	Fixed tasks, limited flexibility	Flexible, can be easily modified
Cost	Expensive and complex	Less costly, does not require specialized hardware
Modification	Difficult to modify once set	Easy to update or adjust as needed
Usage	Suitable for specific, high-performance tasks	Suitable for general tasks and prototyping

Explain why triangles are commonly used as the primary primitive in computer graphics.

Here's a shorter explanation:

Triangles are the primary shape in computer graphics because:

1. **Simplicity**: Triangles have only three points and are always flat, making them stable for complex 3D models.
2. **Efficient Rendering**: GPUs process triangles quickly, allowing for smooth animations and detailed surfaces.
3. **Flexibility**: By combining many triangles, almost any shape or surface can be created, even curves.
4. **Math-Friendly**: Calculations for lighting, shading, and textures are easier with triangles, leading to accurate rendering.

In short, triangles are stable, efficient, flexible, and easy to work with mathematically, making them perfect for 3D graphics.

Explain the level-of-detail rendering.

Solve daya ase

1.

- b) What is a vanishing point? Give an example scenario of multiple vanishing points. [3]
-

1. b. Solution: by 45

In computer graphics and perspective drawing, a vanishing point is a point in the distance where parallel lines appear to converge or "vanish" when extended. A single vanishing point is often used in one-point perspective, where all parallel lines converge to a single point on the horizon line. Multiple vanishing points are used in two-point and three-point perspective to create a more realistic depiction of three-dimensional space.

Two-Point Perspective: For example, drawing a cityscape with tall buildings. The vertical edges of the buildings converge to one vanishing point on the left and another on the right.

Decipher44



c) State the differences between lossless and lossy compression.

2

[3]



d) Describe how emissive display device produces colored images.

2

[3]

Solve daya ase

- b) Explain why the degree of a B-spline curve remains unaffected by the number of control points used in the curve. **3** [3]

The **degree of a B-spline curve** is determined by its formula, not by the number of control points. This means that no matter how many control points are added, the **degree (or smoothness) of the curve** stays the same.

1. **Fixed Degree:** The degree is set when you start defining the B-spline. For example, if it's set as degree 3 (a cubic B-spline), it will stay cubic no matter how many control points you add.
2. **Control Points Affect Shape, Not Degree:** Adding more control points **changes the curve's shape** by pulling or pushing sections of the curve closer to the points, but it does not change the degree.
3. **Example:**
 - Imagine drawing a curve with three control points and making it cubic (degree 3). If you add more points to adjust the shape, it remains a cubic curve but becomes more detailed.

Summary

In B-splines, the degree is fixed and set initially, while the control points only adjust the curve's shape. So, adding points changes the curve's form but not its degree.

 State the differences between raster and vector images.

[4]

2. a. **Solution:** solved by- 146 (from chatgpt)

Raster image: convert (an image stored as an outline) into pixels that can be displayed on a screen or printed.

Vector image: store instructions for displaying the image rather than the pixels needed to display it.

Aspect	Raster Image	Vector Image
Basic Representation	Grid of pixels	Geometric shapes and paths
Resolution Dependency	Resolution-dependent (DPI)	Resolution-independent
Scaling	May result in quality degradation when scaled up or down : Resizing can result quality degradation	No loss of quality when scaled
File Size	Larger file sizes	Smaller file size
Editing Flexibility	Limited flexibility; editing may degrade quality	Highly editable without quality loss
Image Quality	Suitable for photorealistic images	Limited photorealism; best for line art, logos, and illustrations
Storage Format Examples	JPEG, PNG, BMP	SVG, AI, PDF
Printing Quality	Quality may vary based on resolution	Consistently high-quality printing
Ideal Use Cases	Photographs, detailed images	Logos, icons, illustrations

1. Lecture -03

- b) State the differences between raster and vector images.

[4]

1. b. Solution: 024

Raster Graphics	Vector Graphics
They are composed of pixels.	They are composed of paths.
In Raster Graphics, refresh process is independent of the complexity of the image.	Vector displays flicker when the number of primitives in the image become too large.
Graphic primitives are specified in terms of end points and must be scan converted into corresponding pixels.	Scan conversion is not required.
Raster graphics can draw mathematical curves, polygons and boundaries of curved primitives only by pixel approximation.	Vector graphics draw continuous and smooth lines.
Raster graphics cost less.	Vector graphics cost more as compared to raster graphics.
They occupy more space which depends on image quality.	They occupy less space.
File extensions: .BMP, .TIF, .GIF, .JPG	File Extensions: .SVG, .EPS, .PDF, .AI, .DXF

2. Lecture - 03

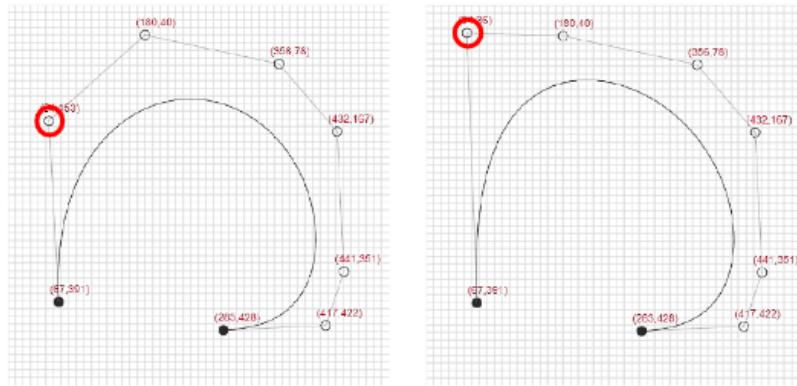
b) Discuss the limitations of Bezier curve.

[2]

2. b. Solution: Rabab 039

Disadvantages

- A change to any of the control point alters the entire curve.
- Having a large number of control points requires high polynomials to be evaluated. This is expensive to compute.



Credit: CPSC 589/689 Course Notes, University of Calgary, Faramarz Samavati

2. Lecture - 05

b) Explain the problems associated with it if homogeneous coordinates were not used in matrix transformation.

[2]

2. b. Solution: 024

In general, Translation process is a summation process. But other transformation processes are matrix multiplication. So, if homogeneous coordinates were not used, the translation process could not be combined with other transformations.

45

Homogeneous coordinates are especially important in 3D graphics, where transformations involve both rotations and translations in three dimensions. Without homogeneous coordinates, the representation and manipulation of 3D transformations would be significantly more complex and error-prone.

perspective projection, are essential for creating realistic 3D scenes. Homogeneous coordinates are crucial for representing and efficiently applying projective transformations. Without them, handling perspective projection would be extremely complex and inefficient