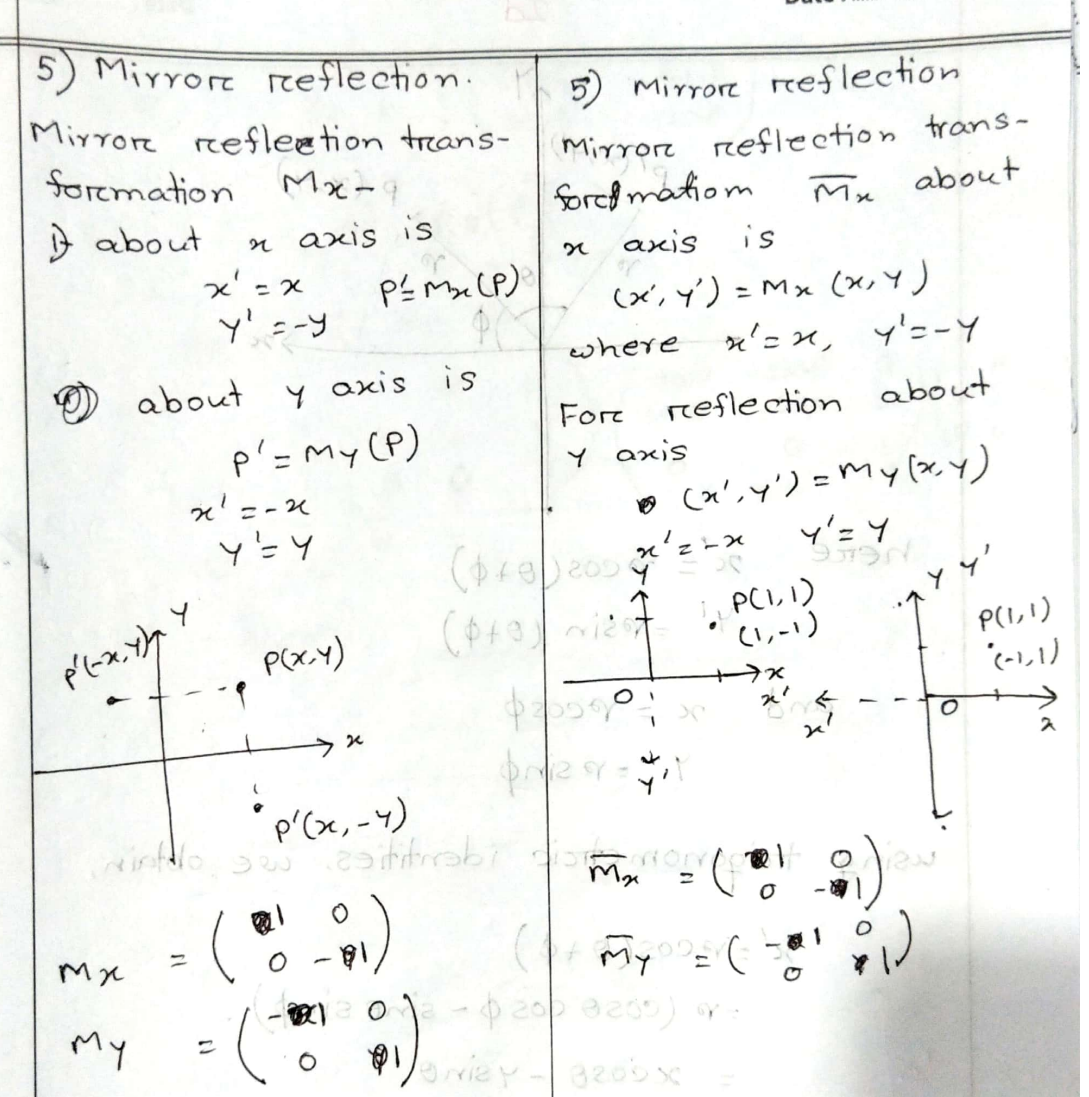
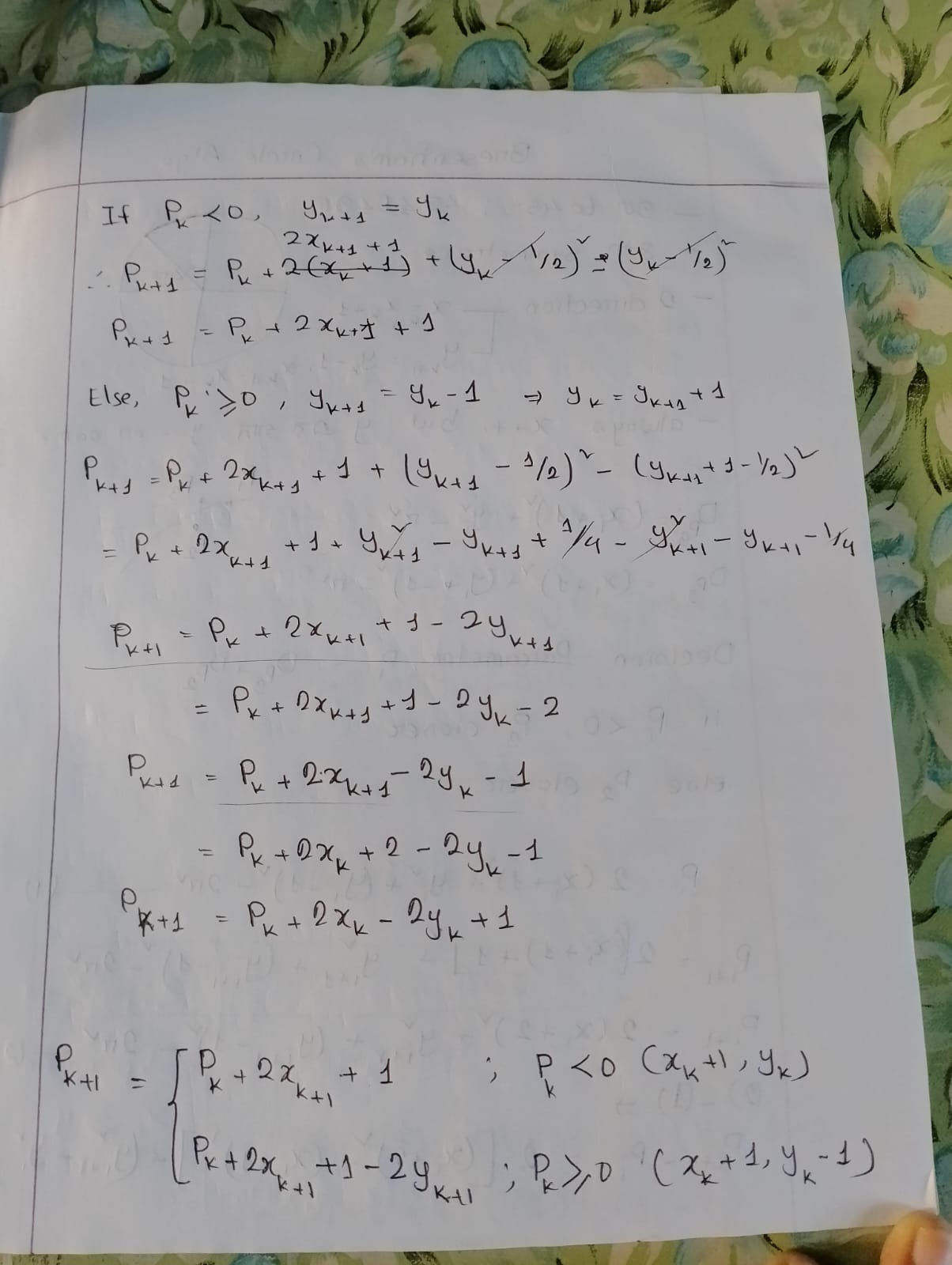
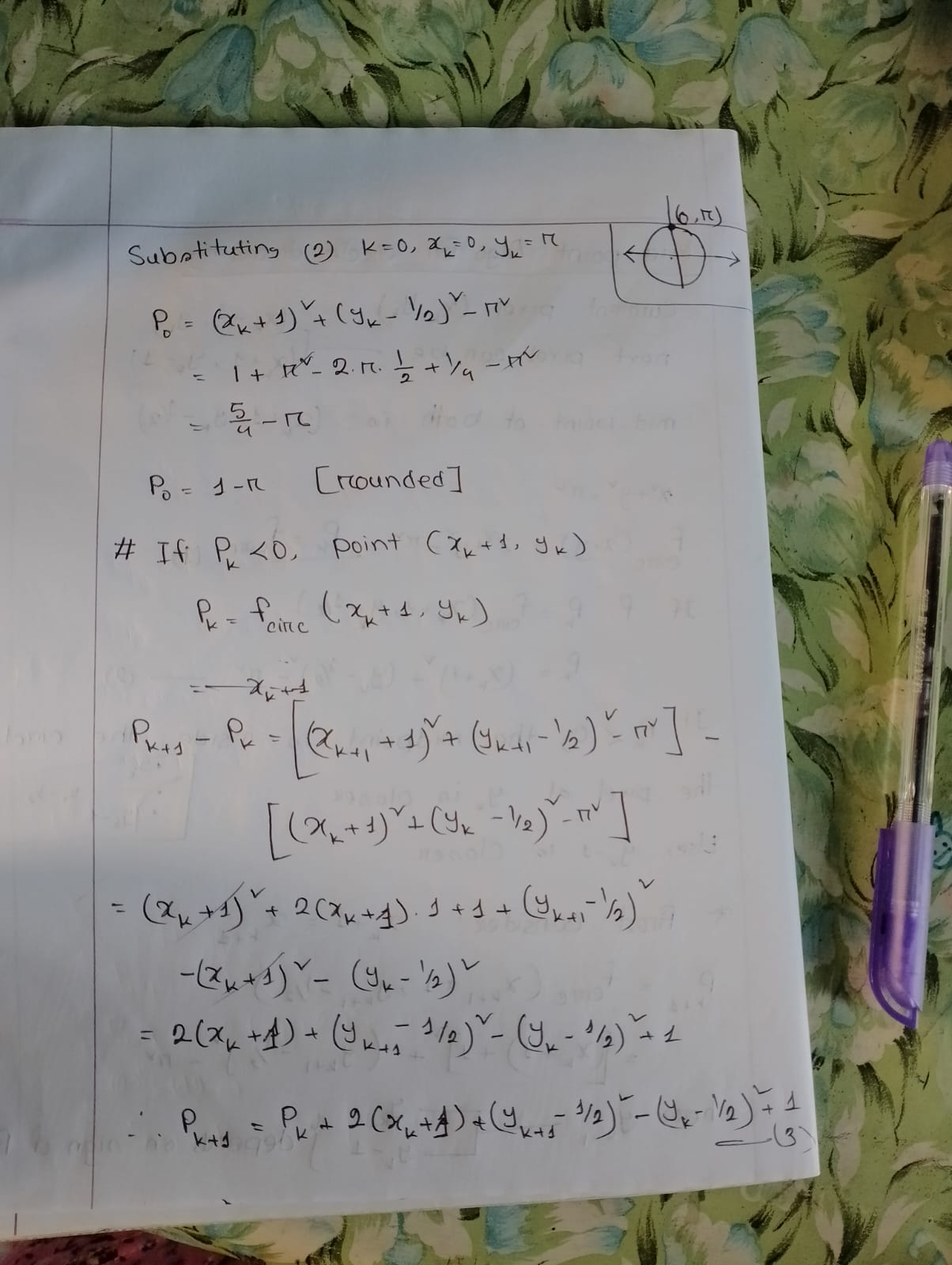
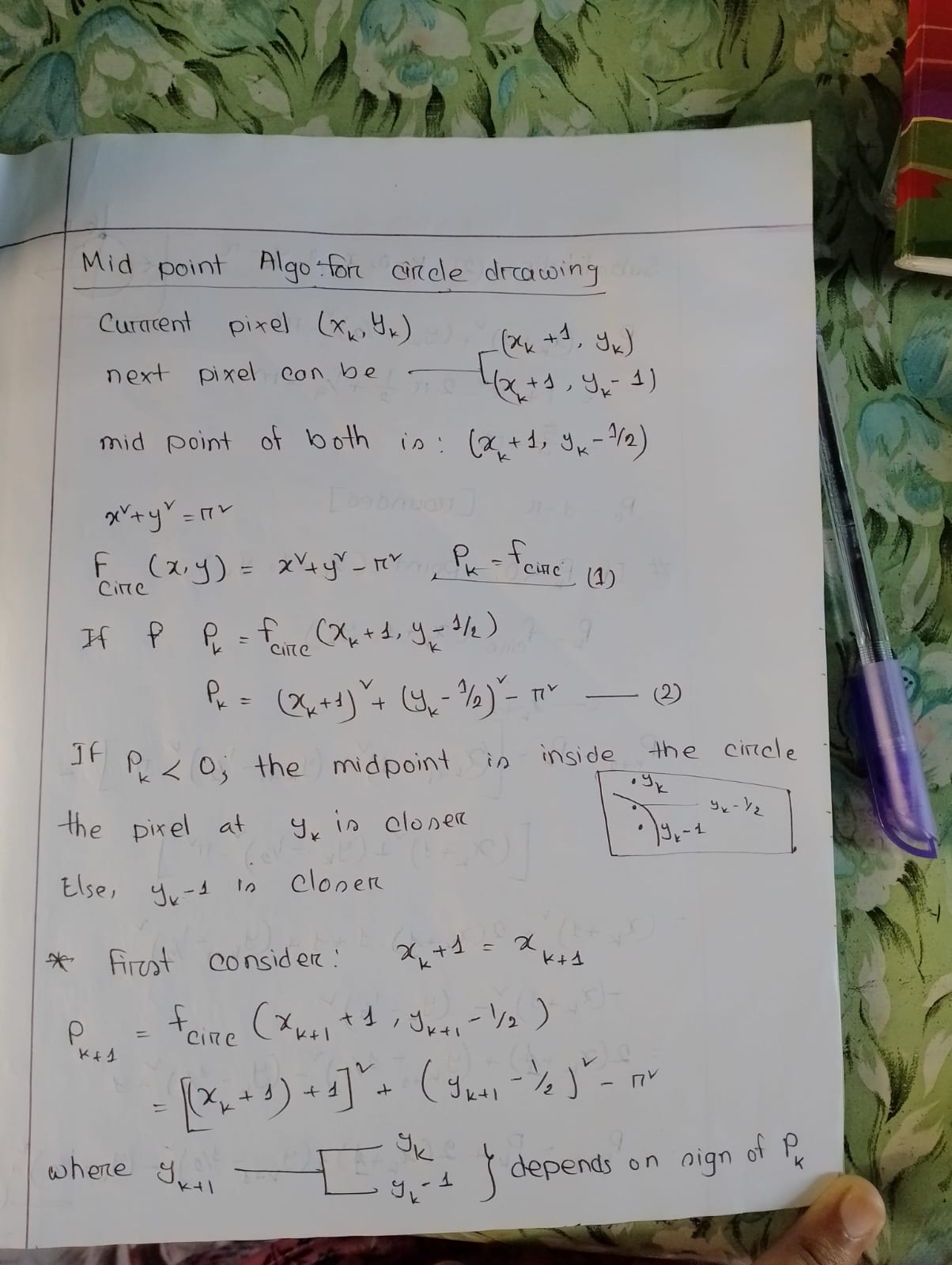
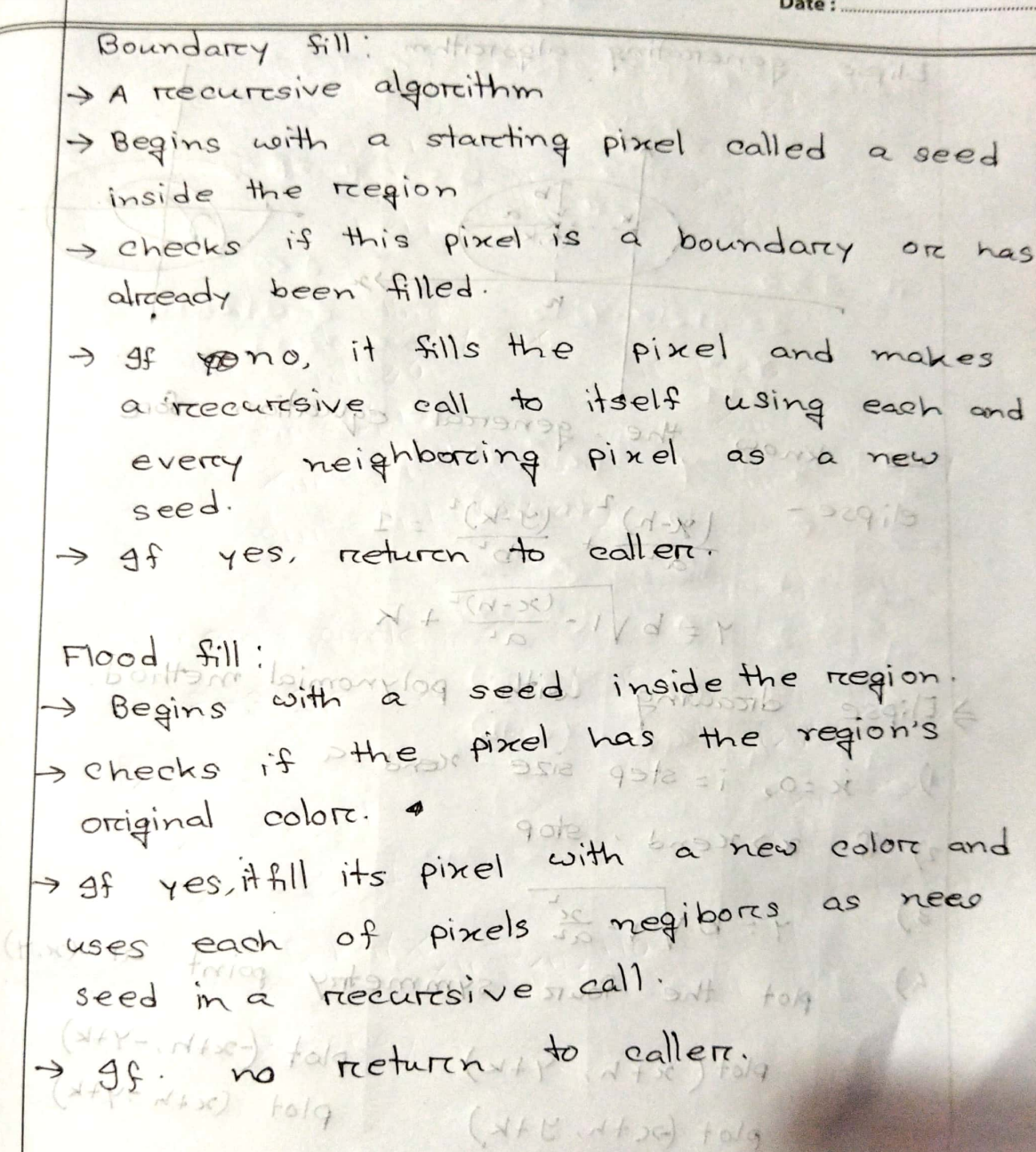
|  |
| --- |
| Question: explain geometric and co-ordinate transformation |
|  |



|  |
| --- |
| Question: difference between geometric transformation and coordinate transformation |
| While geometric transformation and coordinate transformation share similarities in their aim to manipulate objects, there are fundamental differences between the two techniques.   * Geometric transformation directly alters the objects’ properties, such as shape and orientation, while coordinate transformation manipulates the mathematical coordinates that define those objects. * Geometric transformation operates on the objects as a whole, whereas coordinate transformation deals with individual points and their transformations. * In geometric transformation, the properties of the objects are preserved, while in coordinate transformation, the properties may change depending on the transformation applied. |

how does the midpoint circle drawing algorithm work. show that the decision parameter for the midpoint circle drawing algorithm can be expressed by when P <0

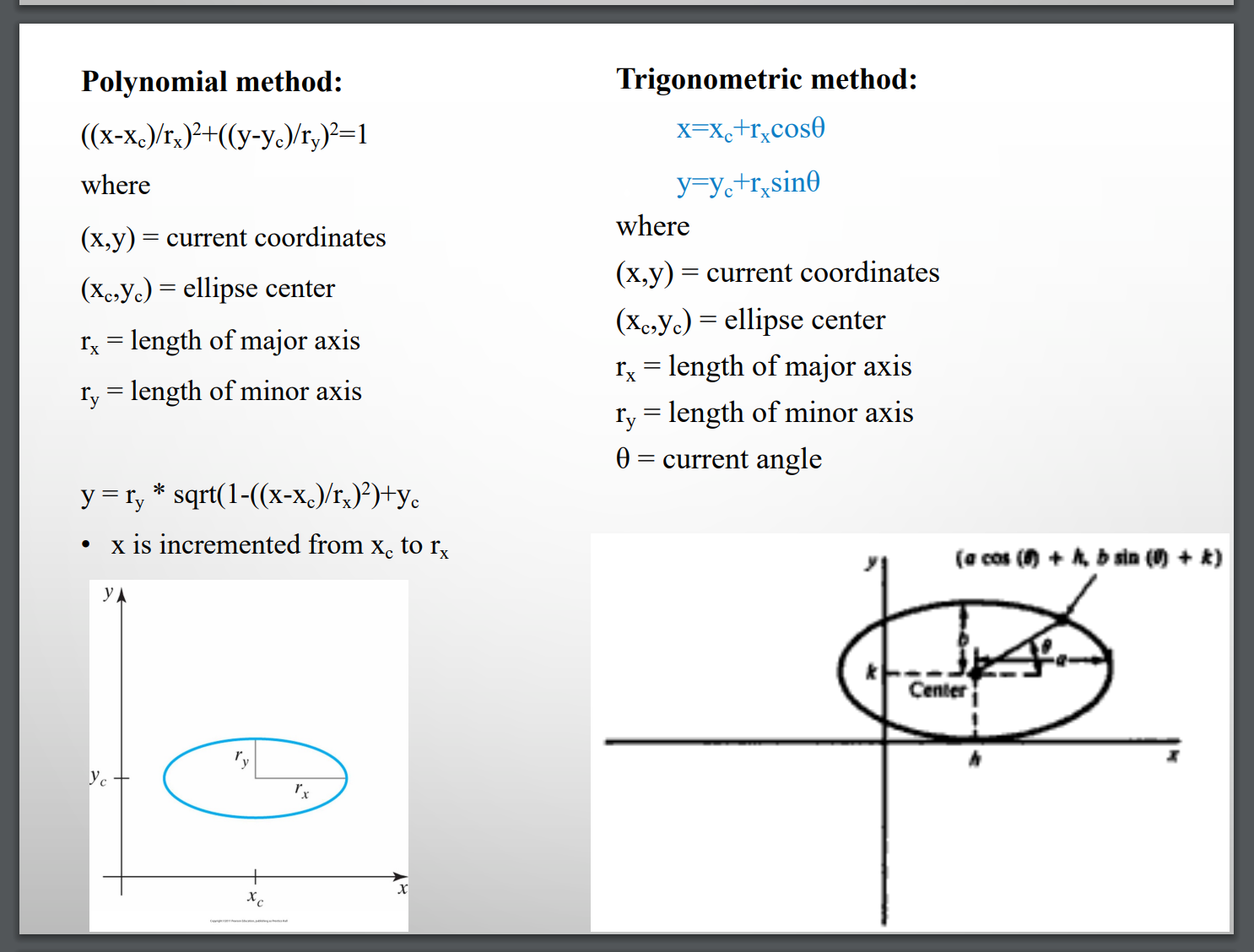
The Midpoint Circle Drawing Algorithm uses integer arithmetic to draw a circle efficiently. It calculates points for one-eighth of the circle using a decision parameter to determine the closest pixel, then mirrors these points using symmetry to draw the full circle. This avoids expensive floating-point calculations.



find the matrix that represent rotation of an object by 30° about the origin and find the new coordinates of a point P(2,-4)

write the steps required to scan convert an ellipse using trigonometric and polynomial method

|  |
| --- |
| **1. Trigonometric Method**  The trigonometric method is based on the parametric equations of an ellipse. An ellipse centered at (x0,y0)(x\_0, y\_0)(x0​,y0​) with semi-major axis aaa and semi-minor axis bbb can be represented parametrically as:  x=x0+a⋅cos⁡(θ)x = x\_0 + a \cdot \cos(\theta)x=x0​+a⋅cos(θ) y=y0+b⋅sin⁡(θ)y = y\_0 + b \cdot \sin(\theta)y=y0​+b⋅sin(θ)  where θ\thetaθ ranges from 0 to 2π2\pi2π.  **Steps:**   1. **Define Parameters:**    * Set the center of the ellipse (x0,y0)(x\_0, y\_0)(x0​,y0​).    * Define the lengths of the semi-major axis aaa and semi-minor axis bbb. 2. **Initialize Angle:**    * Start with an initial angle θ=0\theta = 0θ=0. 3. **Calculate Points:**    * For each angle θ\thetaθ from 0 to 2π2\pi2π, incrementally (e.g., by small steps like 0.010.010.01 radians), compute the ellipse points using the parametric equations: x=x0+a⋅cos⁡(θ)x = x\_0 + a \cdot \cos(\theta)x=x0​+a⋅cos(θ) y=y0+b⋅sin⁡(θ)y = y\_0 + b \cdot \sin(\theta)y=y0​+b⋅sin(θ) 4. **Round and Plot:**    * Round the computed (x,y)(x, y)(x,y) coordinates to the nearest integer values if working in a discrete space (like pixels in raster graphics).    * Plot or draw the point on the display. 5. **Repeat Until Full Circle:**    * Continue calculating and plotting points until θ\thetaθ completes a full circle (i.e., θ=2π\theta = 2\piθ=2π).   **2. Polynomial Method**  The polynomial method uses the implicit equation of an ellipse and focuses on evaluating points within the range to determine which points lie on the ellipse boundary.  The standard form of an ellipse centered at (x0,y0)(x\_0, y\_0)(x0​,y0​) is given by:  (x−x0)2a2+(y−y0)2b2=1\frac{(x - x\_0)^2}{a^2} + \frac{(y - y\_0)^2}{b^2} = 1a2(x−x0​)2​+b2(y−y0​)2​=1  **Steps:**   1. **Define Parameters:**    * Set the center of the ellipse (x0,y0)(x\_0, y\_0)(x0​,y0​).    * Define the lengths of the semi-major axis aaa and semi-minor axis bbb. 2. **Initialize Decision Parameters:**    * Initialize the starting point (x,y)=(0,b)(x, y) = (0, b)(x,y)=(0,b).    * Set the initial decision parameters for the region based on the ellipse equation: d1=b2−a2b+14a2d\_1 = b^2 - a^2b + \frac{1}{4}a^2d1​=b2−a2b+41​a2 3. **Region 1 (slope > -1):**    * While 2b2x<2a2y2b^2x < 2a^2y2b2x<2a2y:      + Plot the points using symmetry (due to the ellipse's symmetrical nature, points can be plotted in all four quadrants).      + Check if d1<0d\_1 < 0d1​<0:        - If yes, increment xxx and update the decision parameter: d1=d1+2b2x+b2d\_1 = d\_1 + 2b^2x + b^2d1​=d1​+2b2x+b2        - If no, increment xxx, decrement yyy, and update the decision parameter: d1=d1+2b2x−2a2y+b2d\_1 = d\_1 + 2b^2x - 2a^2y + b^2d1​=d1​+2b2x−2a2y+b2 4. **Region 2 (slope < -1):**    * Initialize decision parameter for Region 2: d2=b2(x+0.5)2+a2(y−1)2−a2b2d\_2 = b^2(x + 0.5)^2 + a^2(y - 1)^2 - a^2b^2d2​=b2(x+0.5)2+a2(y−1)2−a2b2    * While y>0y > 0y>0:      + Plot the points using symmetry.      + Check if d2>0d\_2 > 0d2​>0:        - If yes, decrement yyy and update the decision parameter: d2=d2−2a2y+a2d\_2 = d\_2 - 2a^2y + a^2d2​=d2​−2a2y+a2        - If no, increment xxx, decrement yyy, and update the decision parameter: d2=d2+2b2x−2a2y+a2d\_2 = d\_2 + 2b^2x - 2a^2y + a^2d2​=d2​+2b2x−2a2y+a2 5. **End Algorithm:**    * Stop when all points of the ellipse have been plotted.   **Summary**   * **Trigonometric Method:** Easy to implement but involves trigonometric computations, which can be computationally expensive. * **Polynomial Method:** Efficient and widely used in computer graphics for drawing ellipses as it relies on integer calculations.   Each method has its use cases depending on the requirements of the implementation, such as computational efficiency, simplicity, or accuracy. |



**2022 Question Set**

**Q1: Section-I (CO1): Answer all the questions**

**a) Point out the differences between raster and vector graphics.**

**Raster Graphics:**

* Composed of pixels arranged in a grid.
* Resolution-dependent; scaling leads to loss of quality.
* Used for photographs and detailed images.

**Vector Graphics:**

* Composed of paths defined by mathematical equations.
* Resolution-independent; scaling retains quality.
* Used for logos, illustrations, and graphic design.

**b) List the adverse effects of scan conversion. Mention the ways to eliminate these adverse effects.**

**Adverse Effects:**

1. **Aliasing:** Jagged edges due to pixel approximation.
2. **Staircase Effect:** Diagonal lines appear as stair-steps.
3. **Moire Patterns:** Interference patterns in textured areas.

**Elimination Methods:**

* Use anti-aliasing techniques like supersampling.
* Increase resolution for smoother transitions.
* Apply smoothing filters.

**c) Define the following:**

**Window Port:** A rectangular region in the world coordinate system selected for viewing.

**Viewport:** A rectangular region on the display device where the selected part of the window port is projected.

**Cel Animation:** Traditional hand-drawn animation on transparent sheets (cels).

**Path Animation:** Movement of objects along a defined path over time.

**d) State Shear and Affine transformations.**

**Shear Transformation:** Distorts the shape of an object by shifting its points along one axis in proportion to their positions on the other axis.

* **Matrix for X-shear:**
* **Matrix for Y-shear:**

**Affine Transformation:** A linear mapping that preserves points, straight lines, and planes, often including translation, rotation, scaling, and shear.

**e) Differentiate between local and global illumination models.**

**Local Illumination Model:**

* Considers only the light directly from a source to an object.
* Examples: Phong and Lambertian models.

**Global Illumination Model:**

* Considers light interactions between multiple surfaces.
* Examples: Ray tracing, radiosity.

**f) Outline the notion of rendering and morphing.**

**Rendering:** Process of generating a 2D image from a 3D model using lighting, shading, and texturing techniques.

**Morphing:** Gradual transformation of one image or shape into another over time.

**Q2: Section-II (CO2): Answer any three of the following**

**a) Discuss perspective projection along with its anomalies. Also explain parallel projection.**

**Perspective Projection:**

* Converts 3D objects into 2D views with depth perception by converging lines at a vanishing point.
* **Anomalies:**
  + Foreshortening of objects.
  + Objects far from the viewer appear disproportionately small.
  + Distortion near the vanishing point.

**Parallel Projection:**

* Projects 3D objects without convergence (lines remain parallel).
* Types: Orthographic and oblique projections.

**b) Describe the Cohen-Sutherland algorithm for line clipping.**

* **Steps:**
  1. Assign region codes to endpoints of the line (based on window boundaries).
  2. Perform logical AND of region codes to check trivial acceptance/rejection.
  3. If partially visible, clip line iteratively until fully inside the window.

**c) Using Bresenham’s circle drawing algorithm, draw the circle with center at (0, 3) and radius 10. Identify all the pixel positions.**

**Algorithm Steps:**

1. Start at (0, R).
2. Use decision parameter .
3. Reflect points in all eight octants using symmetry.
4. Update based on midpoint analysis.

**Pixel Positions:**

1. Initial point: (0, 10).
2. Symmetry points: (10, 3), (-10, 3), (0, 13), (0, -7), (7, 13), (-7, 13), etc.
3. Compute until completing the circle.

**d) Generalize the scaling matrix with respect to a fixed point .**

Where and are scaling factors in the x and y directions respectively.

**Q3: Section-III (CO3): Answer any three of the following**

**a) Write the steps required to scan-convert an ellipse using the trigonometric and polynomial method.**

**Steps:**

1. Use the equation of the ellipse: .
2. Calculate pixel points along the boundary using trigonometric properties.
3. Use polynomial interpolation to fill in pixels for smoothness.

**b) Explain the hidden surface problem. Interpret the steps involved in the Z-buffer algorithm.**

**Hidden Surface Problem:**

* Occurs when objects obscure parts of other objects in a 3D scene.

**Z-buffer Algorithm:**

1. Initialize a depth buffer with maximum depth values.
2. For each pixel, calculate depth (z-value) based on the object.
3. Compare depth values; keep the nearest one.
4. Render the visible object based on depth.

**c) Explain the curve and surface design requirements and discuss the basis function for B-Spline.**

**B-Spline Basis Function:**

* Piecewise-defined polynomial basis functions.
* Provide local control for curve manipulation.
* Preserve continuity across knots.

**d) Demonstrate the steps of constructing a 3D view without hidden surface removal.**

1. Define the 3D scene in world coordinates.
2. Apply viewing transformation to project 3D to 2D.
3. Display all objects without visibility checks.

**Q4: Section-IV (CO4): Answer any three of the following**

**a) Find the normalization transformation that maps a window whose lower left corner is at and upper right-hand corner is at onto a viewport that has lower left corner at and upper right corner at .**

**Normalization Transformation Matrix:**

Simplify to get numerical values.

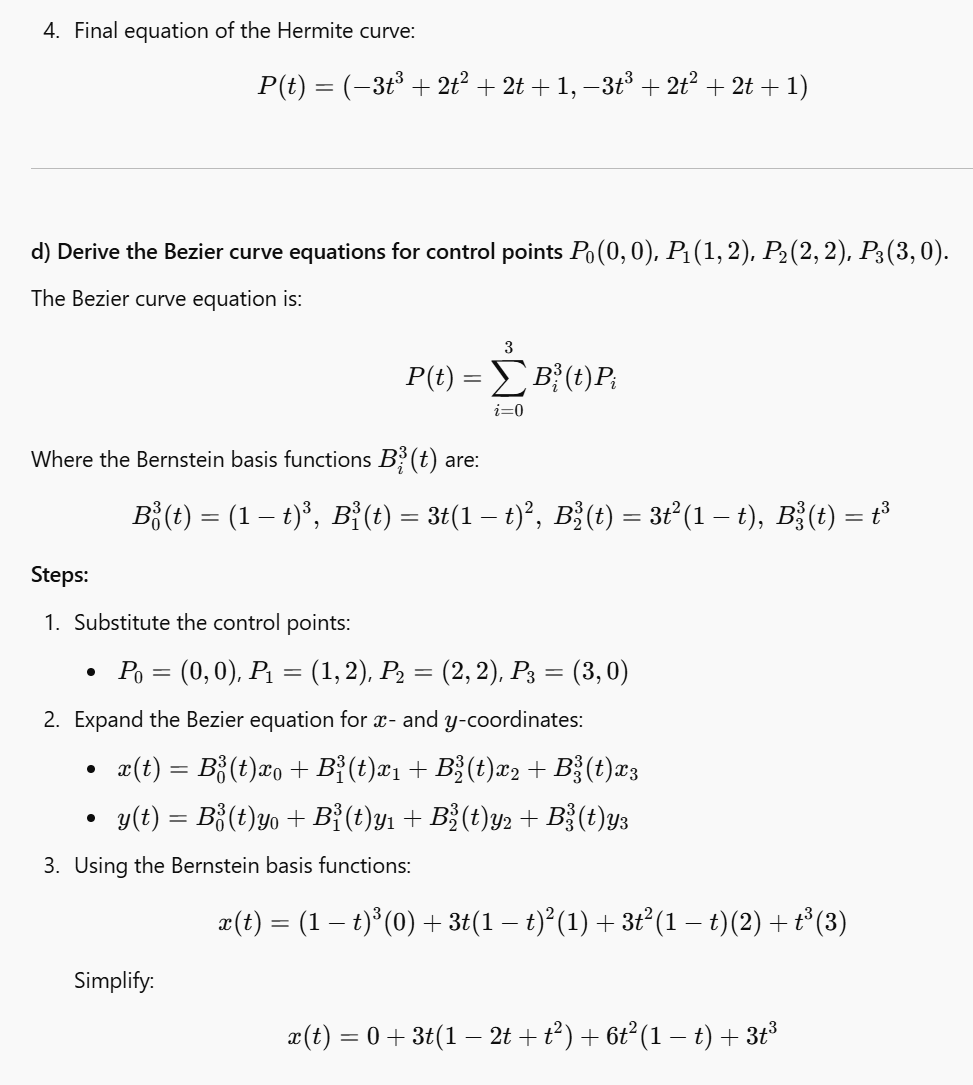
**b) Illustrate the steps of developing animation and figure out a basic rule of animation to avoid distortions.**

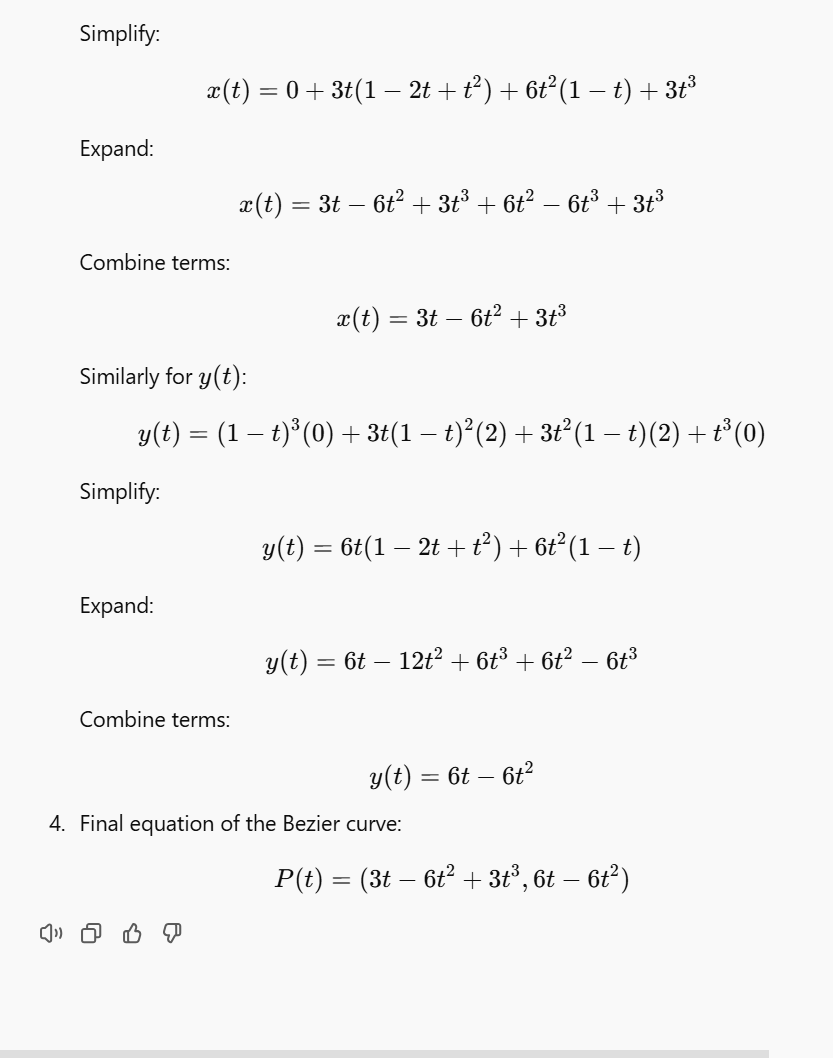
**Steps:**

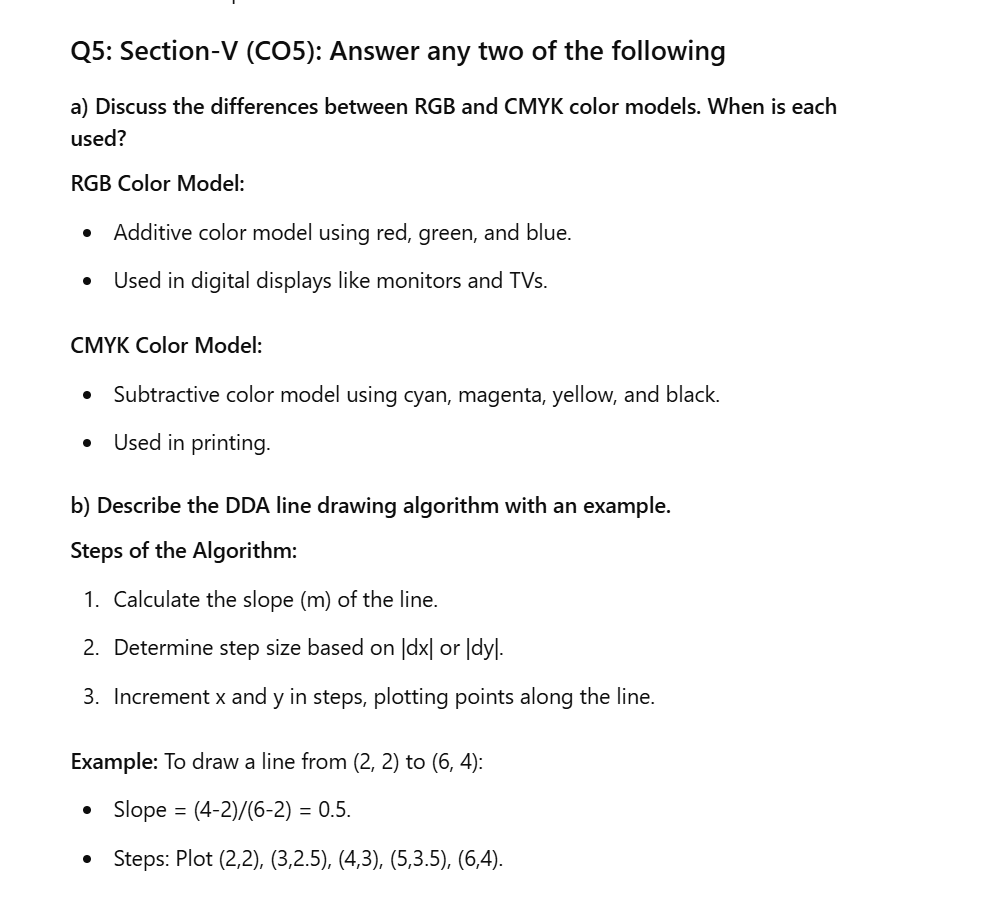
1. Define keyframes for initial and final states.
2. Use interpolation to generate intermediate frames.
3. Ensure continuity and smooth transitions.

**Rule to Avoid Distortion:** Maintain consistent scaling and avoid extreme transformations between frames.









**Q5: Section-V (CO5): Answer any two of the following**

**c) Explain Gouraud and Phong shading models. Compare their advantages.**

**Gouraud Shading:**

* **Definition:** A shading technique that calculates color at the vertices of a polygon and then interpolates these colors across the surface.
* **Process:**
  1. Compute the lighting at each vertex using the illumination model.
  2. Linearly interpolate the vertex colors along edges and scanlines.
  3. Assign the interpolated color to each pixel.
* **Advantages:**
  1. Computationally efficient.
  2. Smooth transitions for simple lighting.
* **Limitations:**
  1. Highlights or specular reflections may be missed if they do not align with a vertex.

**Phong Shading:**

* **Definition:** A shading technique that interpolates surface normals and calculates the illumination model for each pixel.
* **Process:**
  1. Compute vertex normals.
  2. Interpolate normals across edges and scanlines.
  3. Apply the illumination model to calculate pixel color.
* **Advantages:**
  1. Produces realistic highlights and smooth shading.
  2. Handles specular reflections better than Gouraud shading.
* **Limitations:**
  1. More computationally intensive than Gouraud shading.

**Comparison Table:**

| **Feature** | **Gouraud Shading** | **Phong Shading** |
| --- | --- | --- |
| **Efficiency** | Faster | Slower (computationally expensive) |
| **Highlight Accuracy** | May miss highlights | Captures highlights accurately |
| **Smoothness** | Smooth interpolation of vertex colors | Smooth interpolation of normals |

**d) Write the steps for implementing texture mapping in computer graphics.**

**Steps:**

1. **Define the Texture Image:**
   * Create or load a 2D image representing the texture.
   * Assign texture coordinates (u, v) to map the texture.
2. **Assign Texture Coordinates:**
   * Map texture coordinates (u, v) to object vertices in the 3D model.
   * Example: A cube might map the texture onto each face.
3. **Interpolate Texture Coordinates:**
   * For each pixel on the object surface, interpolate (u, v) coordinates based on the triangle or polygon.
4. **Map Texture to Pixels:**
   * Fetch the corresponding pixel from the texture image using (u, v) coordinates.
   * Adjust for tiling or wrapping if necessary.
5. **Apply Filtering:**
   * Use filtering techniques to improve visual quality:
     + **Nearest-neighbor:** Select the closest texture pixel.
     + **Bilinear interpolation:** Smoothly interpolate between nearby pixels.
6. **Render the Final Image:**
   * Blend the texture colors with lighting and shading for realistic output.