Today’s class

- Graphics programming
- Color
testline.c

- Example program testline.c shows the general structure of an OpenGL program
- Note the use of a callback function for the display
Overview of image generation

- Scene – collection of objects in space
- Viewer – forms image from objects
- Light – illuminates the objects and helps visualize their textures
- All of these need to be modeled to generate an appropriate image
Coordinate systems

- World coordinates - the application’s coordinate system
- Device coordinates - the graphics device coordinate system, usually first quadrant
- Normalized device coordinates - \((0, 0) \leq (x, y) \leq (1, 1)\)
Output primitives

- The basic building blocks of graphics images:
  - points
  - lines
  - polygons
  - circles
  - character strings
Attributes

- Properties of output primitives:
  - intensity
  - color
  - line style
  - text style
  - area-filling patterns
OpenGL geometry

- Specified by edges and vertices
- Vertices
  - a sequence or array of coordinates
  - specified with 2, 3, or 4 (homogeneous) coordinates in short, int, float, or double types
  - routines are `glVertex{234}{sifd}[v]()`
- Edges are determined by the order the vertices are entered
OpenGL geometry (cont.)

- Vertices are specified between `glBegin()` and `glEnd()`.
- Parameter to `glBegin()` describes type of geometry:
  - `GL_POINTS` - a disconnected set of points
  - `GL_LINES` - n/2 line segments
  - `GL_LINE_STRIP` - n-1 line segments connected end to end
  - `GL_LINE_LOOP` - n line segments connected end to end and last to first
OpenGL geometry (cont.)

- More parameters to `glBegin()`:
  - `GL_TRIANGLES` - \( \frac{n}{3} \) triangles
  - `GL_TRIANGLE_STRIP` - \( n-2 \) triangles
    - for even \( i \) (0, 2, 4, \ldots) vertices \( i, i+1, i+2 \) define the triangle
    - for odd \( i \) (1, 3, 5, \ldots) vertices \( i+1, i, i+2 \) define the triangle
  - `GL_TRIANGLE_FAN` - \( n-2 \) triangles
    - vertices 0, \( i+1, i+2 \) define the triangles
OpenGL geometry (cont.)

- More parameters to `glBegin()`:
  - `GL_QUADS` - n/4 quadrilaterals
  - `GL_QUAD_STRIP` - n/2-1 quadrilaterals
    - vertices 2i, 2i+1, 2i+3, 2i+2 define the quadrilateral
  - `GL_POLYGON` - a single convex polygon
Example program

- Program `brownian.c` is an example of OpenGL and GLUT
In class exercise

- The picture below shows the first step in generating the Koch curve, a famous fractal.
- Write an OpenGL code fragment to draw the image on the right.
Color
Color characterization

- Colors are combinations of wavelengths of light
- Characterized by a function, \( C(\lambda) \), which represents the strength of each wavelength of visible light present in the color
- To accurately produce a color on the CRT would require representation of each wavelength present in the color
- Impossible to do, and really does not represent how our eyes perceive color
Cones

- Color receptors in our eyes are called **cones**
- Come in three types
- Each type has its own sensitivity curve, which represents how sensitive it is to the different wavelengths of light
Tristimulus values

- When cone $i$ is exposed to a color distribution $C(\lambda)$ it sends the brain a “number” which represents the total response of that cone to the color:

$$A_i = \int S_i(\lambda)C(\lambda)d\lambda$$

- Thus, there are 3 numbers (one for each cone) that are sent to the brain to represent a color

- These are known as the **tristimulus values**

- The brain sums these numbers
CRT color model

- Applying the tristimulus approach to a CRT we have 3 color guns (R, G, B)
- Each gun’s intensity is weighted by the appropriate tristimulus value to produce an overall color
- Note that this is an **additive** system
RGB color model

- Most common
- Defined on a unit cube
- Black at (0, 0, 0), white at (1, 1, 1)
- Diagonally opposite colors are complementary (two colors added together give white; e.g. yellow (1,1,0) + blue (0,0,1))
- \((R,G,B) = (1,1,1) - (C,M,Y)\)
Color in OpenGL

- Two models
  - RGBA
    - A indicates opacity
      - a value of 1.0 indicates an opaque color
      - a value of 0.0 indicates a transparent color
    - RGB is the same as RGBA with A set to 1.0
    - This is the internal form used in OpenGL’s state
  - Color index
    - An index into a table of RGB values
    - Requires interaction with the windowing system
Color on paper

- For printing on paper, a complementary approach is needed since you start with white paper (the presence of all color).
- A *subtractive* system is needed, such that when you place all 3 primary colors down you get black.
- The standard subtractive system is CMY (cyan, magenta, yellow).
HLS

- A distortion of the RGB cube
- A double cone
- Hue is location of the dominant wavelength in the color (measured as angle from 0° (red))
- Lightness is measured from 0 to 1 vertically
- Saturation is the radial distance from the lightness axis, also 0 to 1
Color angles

- Red = 0°
- Yellow = 60°
- Green = 120°
- Cyan = 180°
- Blue = 240°
- Magenta = 300°
HSV

- A single cone
- Hue and saturation the same as HLS
- Value is captured in the vertical axis, also on a 0 to 1 scale
HSV to RGB conversion

- If $H = 360$ then $H = 0$
- $H = H / 60$
- $I = \text{floor}(H)$
- $F = H - I$
- $P1 = V \times (1 - S)$
- $P2 = V \times (1 - S \times F)$
- $P3 = V \times (1 - (S \times (1 - F)))$
- Case I of
  - 0: $R = V; G = P3; B = P1$
  - 1: $R = P2; G = V; B = P1$
  - 2: $R = P1; G = V; B = P3$
  - 3: $R = P1; G = P2; B = V$
  - 4: $R = P3; G = P1; B = V$
  - 5: $R = V; G = P1; B = P2$
Programming the color lookup table

- In OpenGL a float is used for the red, green, and blue components.
- One approach is to equally space the colors at some fixed L and S on the HLS model.
- Another approach is to determine the range of red, green, and blue in the image and quantize on three axes.
Index color mode

- **GLUT_INDEX** is used for the display mode
- `glutSetColor(index, r, g, b)` enters a color map entry
- `glIndexi(index)` chooses a color map entry for the current color
- `glClearIndex(index)` sets the clear color for a window in index mode
Default colors

- 0 = black
- 1 = red
- 2 = green
- 3 = yellow
- 4 = blue
- 5 = magenta
- 6 = cyan
- 7 = white
Color index mode on PCs

- 8 bit planes
- 256 entries in color index
- Need to be in 256-color mode
- Avoid setting colors 0-7
Unix machines

- SGI machines
  - 12 bit planes
  - avoid setting colors 0-255
- Sun workstations
  - 8 bit planes
  - no default colors
Example program

- Program `colorMap.cpp` shows using the color map and indexed color.
- A large list of colors and their RGB values has been posted on the Example Programs page of the course website.