

# Lecture Notes for Thursday September 2, 2004

## GL Drawing in OpenGL:

- Setting up the view
  - Put in reshape callback!
  - Not in an initialization function before `glutMainLoop()`! **Q: Why?**
- Drawing
  - Put stuff in display callback
- Updating
  - **Either** Post redisplay in idle callback
  - **Or** Post redisplay after changes

## Review of Setting up the View (i.e. Your Reshape Callback!):

- Set the portion of the window to draw into (the *viewport*).
  - `glViewport( 0, 0, width, height );` to get the whole window.
- Set up the *projection* — how coordinates you specify map to the screen.
  - Tell OpenGL you're changing the projection using `glMatrixMode(GL_PROJECTION);`
  - Clear existing information about the projection occurs using `glLoadIdentity();`
  - Set coordinate system for the screen using `gluOrtho2D( 0, w, 0, h );` to set lower left to (0,0) and the upper right to (w,h).
    - \* Note `gluOrtho2D()` performs a matrix multiply with the *projection matrix stack*
    - \* Important to clear stack (with `glLoadIdentity()`) first!
  - Tell OpenGL you're done changing the projection (move onto drawing!) using `glMatrixMode(GL_MODELVIEW);`

## Drawing Lines (Basic):

- `void glBegin( GLenum mode );` — Line modes include:
  - `GL_LINES` ( $v_0$  and  $v_1$  make a line,  $v_2$  and  $v_3$  make a line, ...)
  - `GL_LINE_STRIP` ( $v_0$  and  $v_1$  make a line,  $v_1$  and  $v_2$  make a line, ...)
  - `GL_LINE_LOOP` ( $v_0$  and  $v_1$  make a line,  $v_1$  and  $v_2$  make a line, ... ,  $v_n$  and  $v_0$  make a line)
- `void glColor3f( GLfloat red, GLfloat green, GLfloat blue );` — Only between `glBegin()` and `glEnd()`!
- `void glVertex2i( GLint x, GLint y );` — Only between `glBegin()` and `glEnd()`!
- `void glEnd( void );`

**Q: What would happen if between glBegin/glEnd a call to gluOrtho2D() or glFlush() occurred???**

Between glBegin() and glEnd(), many OpenGL function calls generate errors or have undefined results! Check Table 2-3 (in the Redbook, p. 47-48) for info on legal GL calls.

**Q: What about other C/C++ code?**

Other code (for loops, if/then, math functions, etc.) is valid, if you want to compute the locations of your geometry on the fly, for instance.

## Drawing Lines (Advanced):

- Varying line width:
  - `void glLineWidth( GLfloat width );`
    - \* Sets GL state specifying line width
    - \* Must be called *before* glBegin(GL\_LINES)... **Q: Why???**
    - \* Generally should be integer width... **Q: When would non-integer width be useful???**
- Patterned lines (often called “stippled lines”):
  - `void glLineStipple( GLint factor, GLushort pattern );`
    - \* *pattern* is a 16-bit bitmask (usually specified as hex, e.g. 0x0F0F) for the pattern
    - \* The pattern length (number of pixels per bit of the pattern) is multiplied by *factor*
  - `void glEnable( GLenum cap );`
    - \* The capability to enable for lines is GL\_LINE\_STIPPLE
  - `void glDisable( GLenum cap );` **Q: Why does glDisable() need a parameter???** **glEnd() doesn't!**
  - **Q: Where do these functions need to be called???**
- Antialiased lines: **Q: What is “aliasing”?**
  - **Brainstorm: How would you reduce aliasing?**
  - Thought: give line an area (Fig 6-3, Redbook p.242) **Q: How does that help???**
  - Don't “sample” at just the center of the line. Multisample. (Can do this in OGL, but won't discuss now)
  - Compute how much of each pixel is covered by the line. How to do in OpenGL:
    - \* Enable anti-aliased lines: `glEnable(GL_LINE_SMOOTH);`
    - \* Enable blending: `glEnable(GL_BLEND);` **Q: What is blending???** **Q: Why enable???**
    - \* Setup blending: `glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);` What does this mean?
      - First entry: The source factor (i.e., what to multiply the source/fragment by)
      - Second entry: The destination factor (i.e., what to multiply the destination/framebuffer by)
      - The constants are pretty self explanatory. This call means: multiply the fragment's color by its alpha value, multiply the color in the framebuffer by one minus the fragment's alpha, then **add** the result. **Q: What is an alpha value???**
    - \* Draw your lines.
    - \* Disable GL\_LINE\_SMOOTH and GL\_BLEND
    - \* **Note:** Enable/Disable do not need to go in display function (as long as all lines are antialiased)