Additional OpenGL Tidbits

• Function naming conventions
  – Prefixes: gl*, glu*, glut*, GL*, GLU*, GLUT* — you know about those already
  – Postfixes
    • OpenGL functions have explicit variants for different number types, such as integers, floats, and doubles
    • Many functions also take arrays (a.k.a. vectors)
    • A final variant: some functions have versions with different numbers of arguments (2, 3, 4)
      – So:
        • function_name ← base_name [arg_count] [arg_type] [vector]
          – glColor()
            - glColor3s(GLshort red, GLshort green, GLshort blue)
            - glColor3ub(GLubyte red, GLubyte green, GLubyte blue)
            - glColor4b(GLbyte red, GLbyte green, GLbyte blue, GLbyte alpha)
            - glColor4dv(const GLdouble *v) — where v is expected to point to a 4-element array representing RGBA
          – glVertex()  
            - glVertex2d(GLdouble x, GLdouble y) — 2D is a special case of 3D where z = 0
            - glVertex2iv(const GLint *v) — where v[0] is x and v[1] is y
            - glVertex3v(const GLfloat *v)
            - glVertex4iv(const GLint x, GLint y, GLint z, GLint w)
            - glVertex4dv(const GLfloat *v)
        • glTranslatef(GLfloat x, GLfloat y, GLfloat z);
        • glTranslated(GLdouble x, GLdouble y, GLdouble z)
        • See Table 1-1 in the red book for the full acronym–type table

GLUT Prefab Shapes

• Useful for experimentation and prototyping; you will eventually grow out of them; they all paint at the origin
  – glutWireSphere(GLdouble radius, GLint slices, GLint stacks)
  – glutSolidSphere(GLdouble radius, GLint slices, GLint stacks)
  – glutWireCone(GLdouble base, GLdouble height, GLint slices, GLint stacks)
  – glutSolidCone(GLdouble base, GLdouble height, GLint slices, GLint stacks)
  – glutWireCube(GLdouble size)
  – glutSolidCube(GLdouble size)
  – glutWireTorus(GLdouble innerRadius, GLdouble outerRadius, GLint sides, GLint rings)
  – glutSolidTorus(GLdouble innerRadius, GLdouble outerRadius, GLint sides, GLint rings)
  – glutWireDodecahedron(void)
  – glutSolidDodecahedron(void)
  – glutWireTeapot(GLdouble size)
  – glutSolidTeapot(GLdouble size)
  – glutWireOctahedron(void)
  – glutSolidOctahedron(void)
  – glutWireTetrahedron(void)
  – glutSolidTetrahedron(void)
  – glutWireIcosahedron(void)
  – glutSolidIcosahedron(void)
Settings and Optimization

- OpenGL has tons of switches and settings. Many have their own functions because they specify values of different types:
  - `glColor`, `glTexImage2D`
  - `glMatrixMode`, `glViewport`
  - `glNormal`

- Many settings, however, are just on/off toggles. For those, OpenGL supplies a pair of functions, which take a constant representing the “switch”
  - `glEnable()` turns something on; `glDisable()` turns it off
  - sample enable/disable constants include `GL_CULL_FACE`, `GL_LINE_SMOOTH`, `GL_LIGHT0`, and many many more

- If you’re good with your data structures and arrays, you can optimize things beyond individual `glVertex*()` calls
  - vertex arrays (red book chapter 2)
  - display lists (red book chapter 7 — the whole chapter!)