CS 543 - Computer Graphics: 3D Modeling

by

Robert W. Lindeman

gogo@wpi.edu

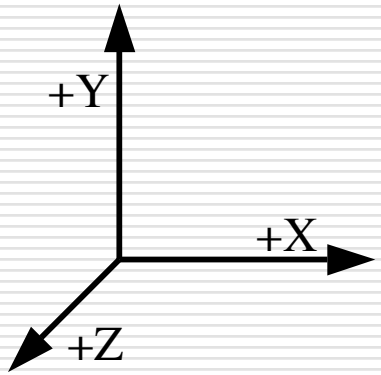
(with help from Emmanuel Agu ;-)

Overview of 3D Modeling

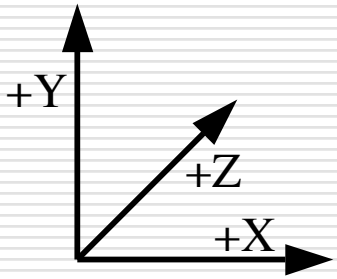
- Modeling
  - Create 3D model of scene/objects
- OpenGL commands
  - Coordinate systems (left hand, right hand)
  - Basic shapes (cone, cylinder, *etc*.)
  - Transformations/Matrices
  - Lighting/Materials
  - Synthetic camera basics
  - View volume
  - Projection
- GLUT models (wireframe/solid)
- Scene Description Language (SDL)

Coordinate Systems

- Right-handed and left-handed coordinate systems
  - Make an "L" with index finger and thumb
  - Right-handed is used in OpenGL
  - Converting from one to the other is a simple transformation



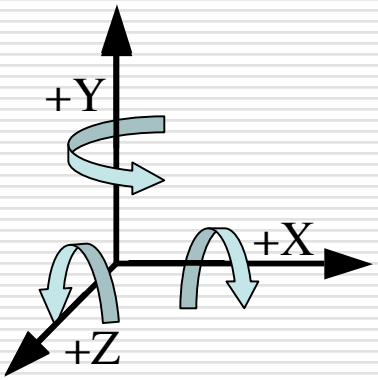
Right-Handed Coordinate System

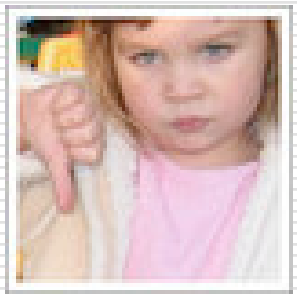


Left-Handed Coordinate System

Right-Handed Coordinates

- To determine positive rotations
  - Make a fist with your right hand, and stick thumb up in the air (CCW)





GLUT Models

- Two main categories
  - Wireframe Models
  - Solid Models
- Basic Shapes
  - Cylinder: **`glutWireCylinder( ), glutSolidCylinder( )`**
  - Cone: **`glutWireCone( ), glutSolidCone( )`**
  - Sphere: **`glutWireSphere( ), glutSolidSphere( )`**
  - Cube: **`glutWireCube( ), glutSolidCube( )`**
- More advanced shapes:
  - Newell Teapot: (symbolic)
  - Dodecahedron, Torus

GLUT Models: `glutSolidTeapot( )`

- The famous Utah Teapot (a.k.a. Newel Teapot) has become an unofficial computer graphics mascot

`glutSolidTeapot( 0.5 )`

More teapot info:
http://www.sjbaker.org/teapot/

- Create a teapot with size 0.5, and position its center at (0.0, 0.0, 0.0)

  (also **`glutWireTeapot( )`**)

- Again, you need to apply transformations to position it at the right spot

GLUT Models

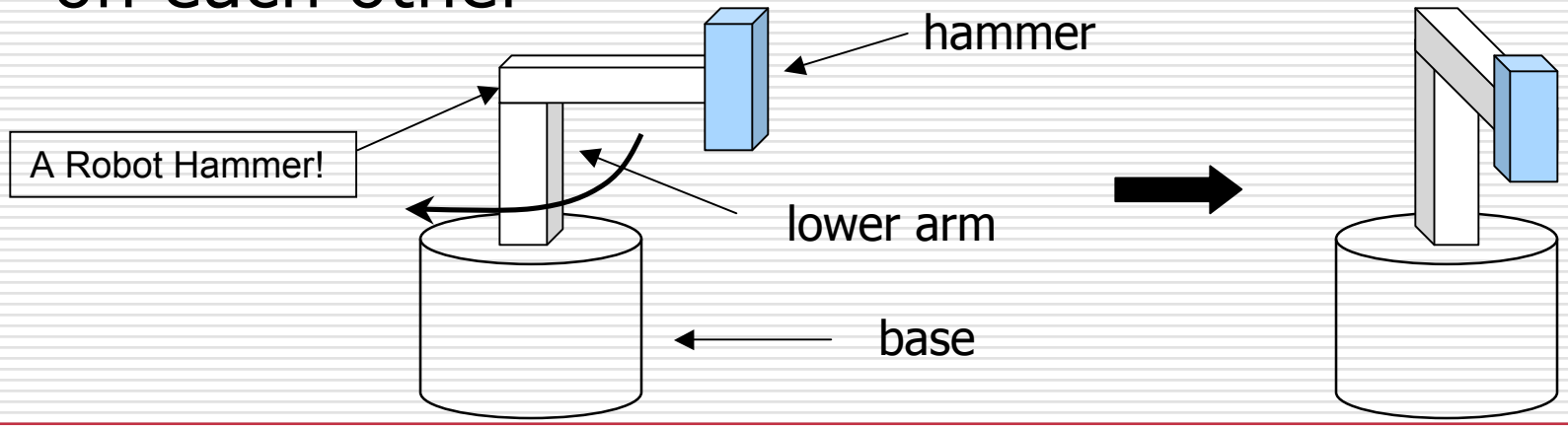
- GLUT functions
  - Actually generate sequence of points that define corresponding shape
  - Are centered at 0.0
- Without GLUT models
  - Use generating functions
  - More work!!
- What does it look like?
  - Generates a list of points and polygons for simple shapes
  - Spheres/Cubes/Cylinder/*etc.*

Example: Generating a Cylinder

```
glBegin( GL_QUADS )
  For each A = Angles {
    glVertex3f( R*cos( A ), R*sin( A ), 0 );
    glVertex3f( R*cos( A+DA ), R*sin( A+DA ), 0 );
    glVertex3f( R*cos( A+DA ), R*sin( A+DA ), H );
    glVertex3f( R*cos( A ), R*sin( A ), H );
  }
glEnd( )
// Make Polygons for Top/Bottom of cylinder
```

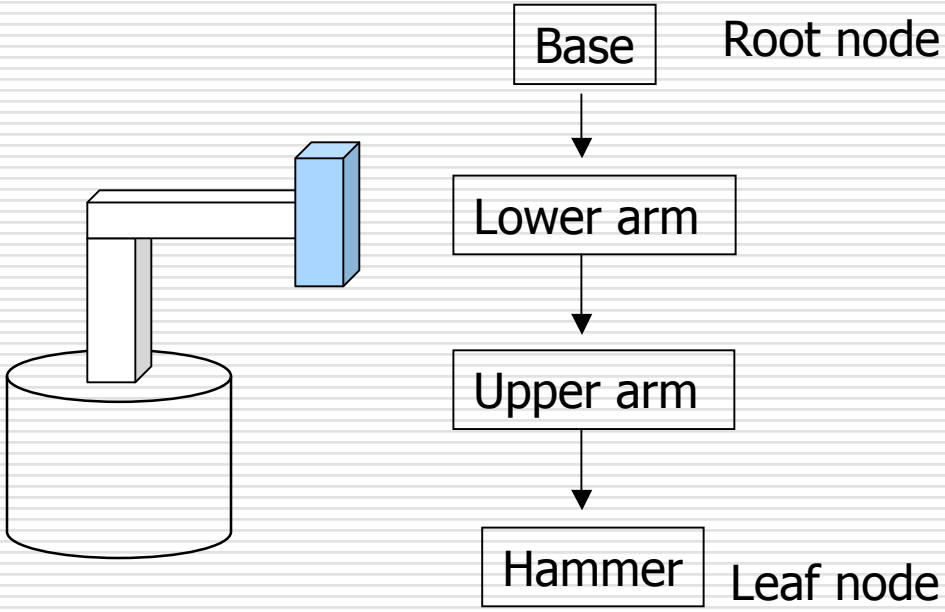
Hierarchical Transformations

- Two ways to model
  - Immediate mode (OpenGL)
  - Retained mode (SDL)
- Graphical scenes have object dependencies
- Many small objects
- Attributes (position, orientation, *etc.*) depend on each other



Hierarchical Transformations (cont.)

- Object dependency description using tree structure

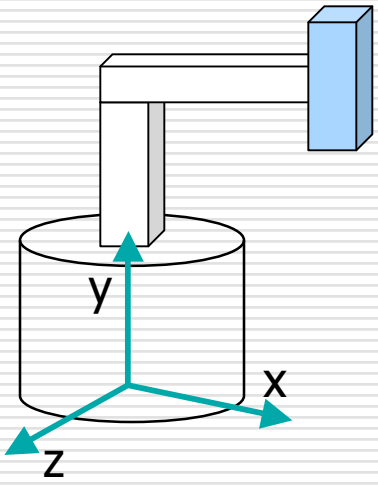


Object position and orientation can be affected by its parent, grand-parent, grand-grand-parent, … nodes

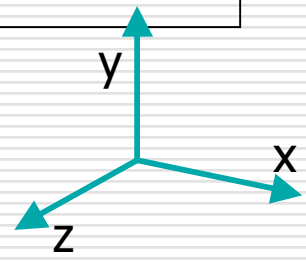
Hierarchical representation is known as **Scene Graph**

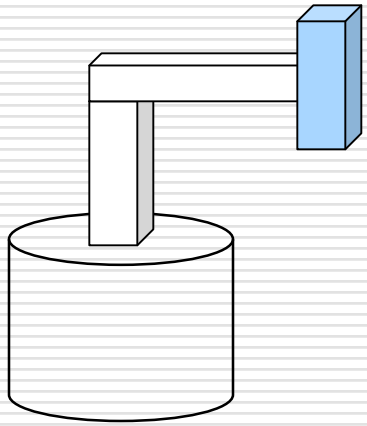
Transformations

- Two ways to specify transformations
  1. Absolute transformation: each part of the object is transformed independently relative to the origin



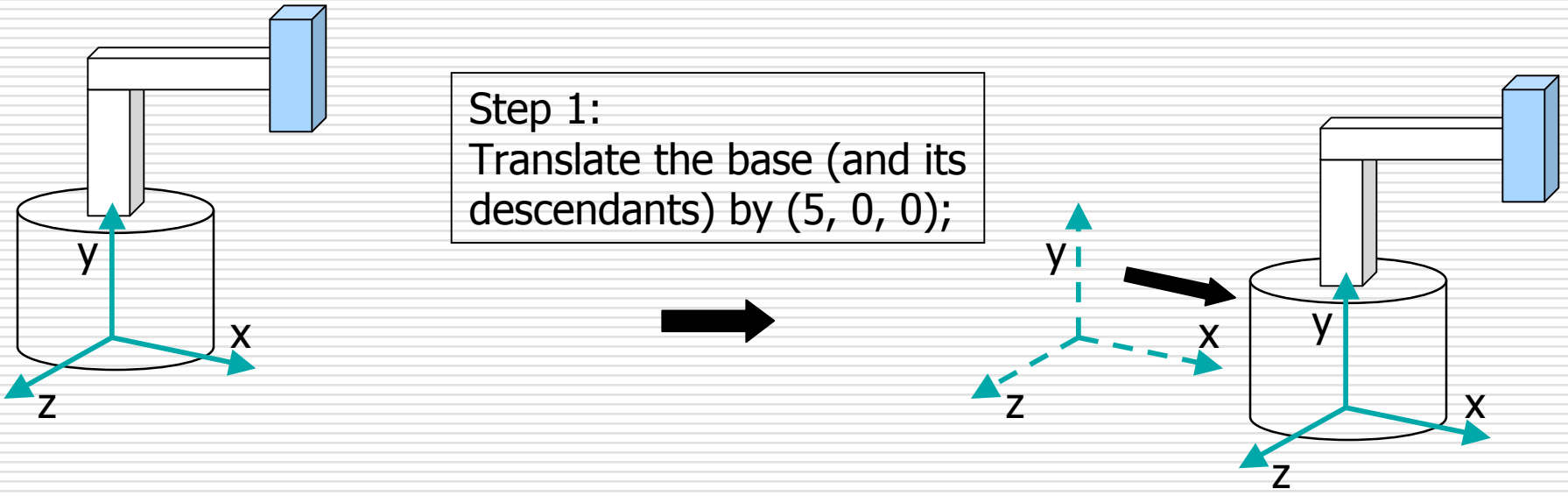

```
Translate the base by (5, 0, 0);
Translate the lower arm by (5, 2, 0);
Translate the upper arm by (5, 4, 0);
Translate the hammer head by (5, 4, 4)
...
```





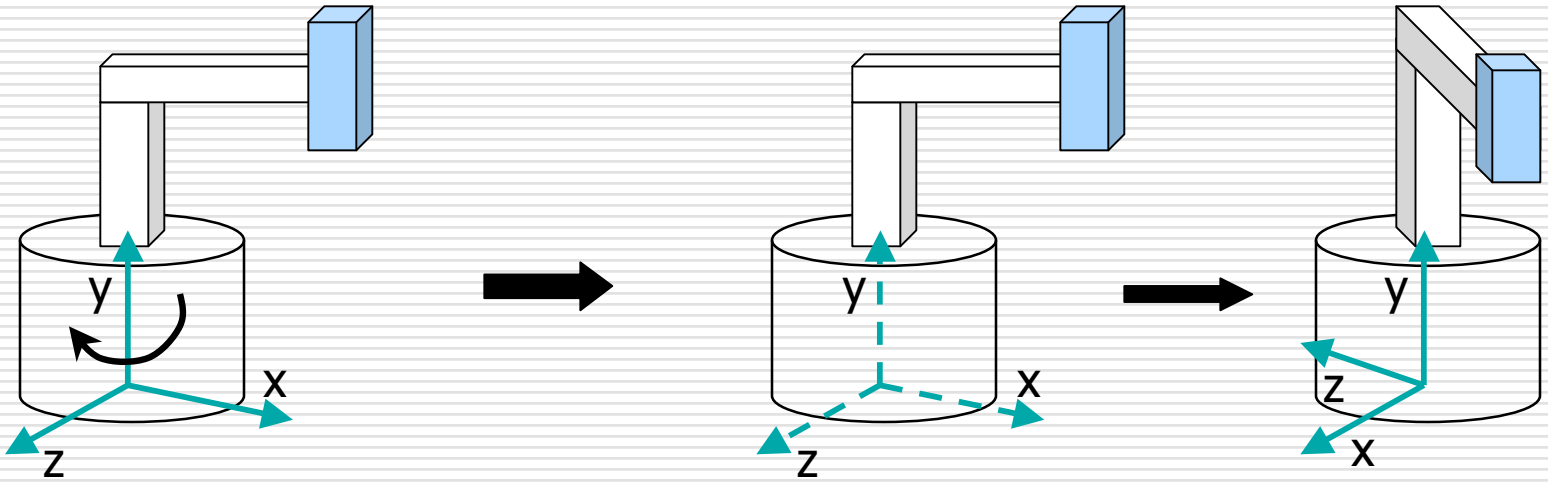
Relative Transformations

- A better (and easier) way
  1. Relative transformation: Specify the transformation for each object relative to its parent

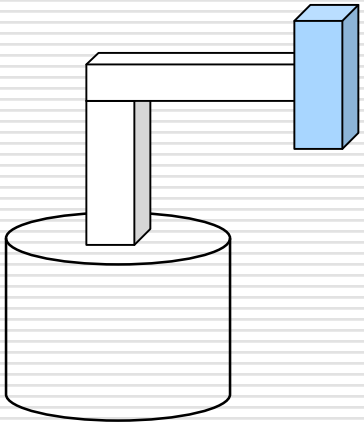


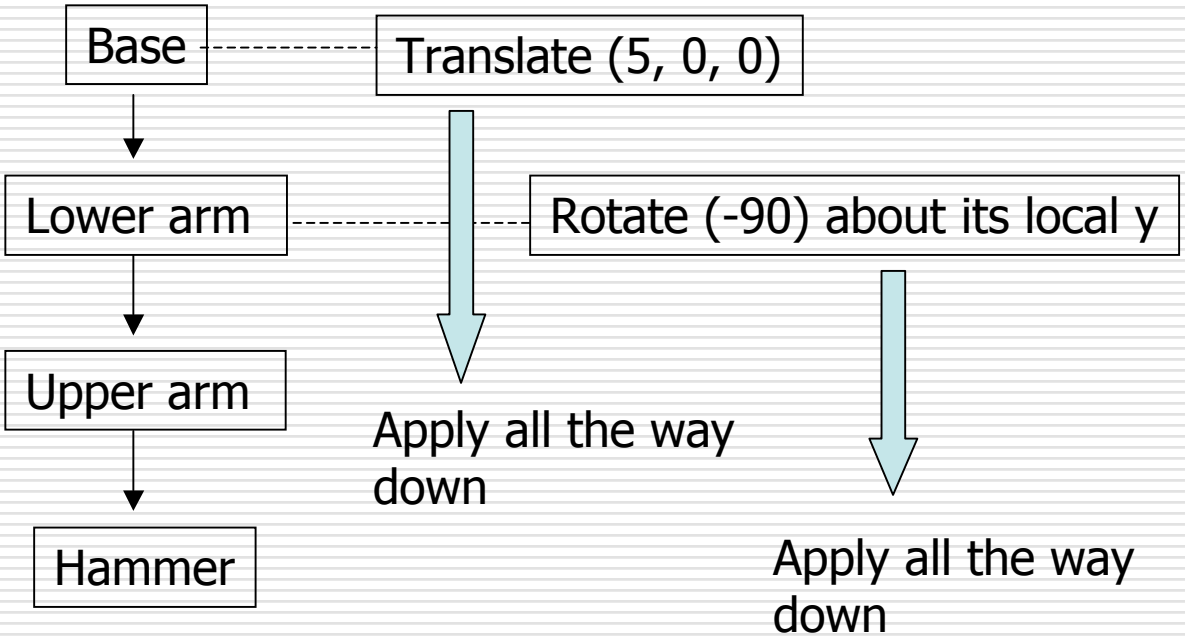
Relative Transformations (cont.)

Step 2:
Rotate the lower arm and (its descendants) relative to the base's local y axis by -90 degrees



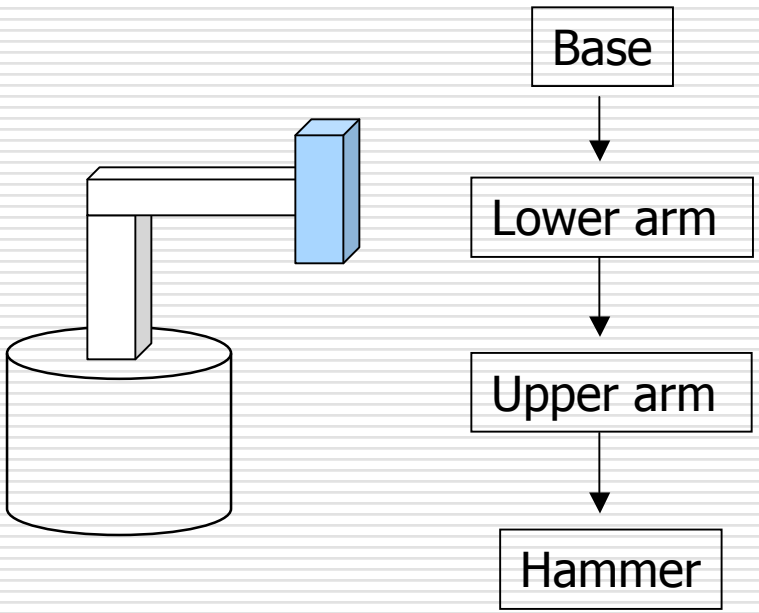
Relative Transformations Using a Scene Graph





Relative Transformations Using OpenGL

- Translate base and all its descendants by (5, 0, 0)
- Rotate the lower arm and its descendants by -90 degree about the local y



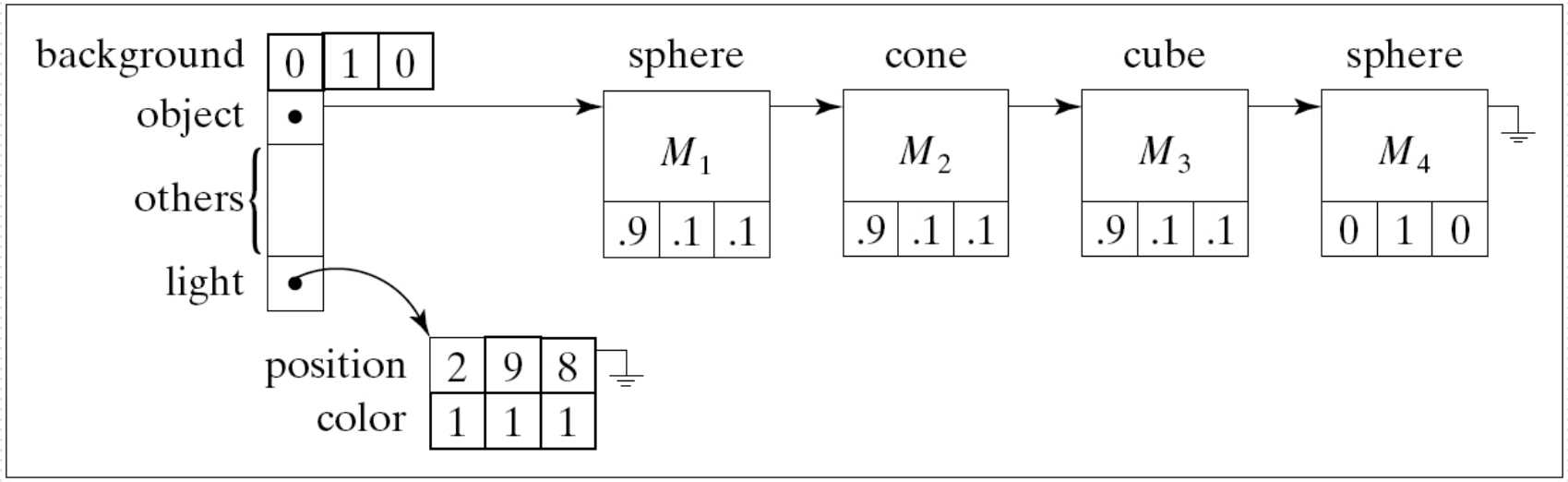

```
glMatrixMode( GL_MODELVIEW );
glLoadIdentity( );
...
// setup your camera
...
glTranslated( 5, 0, 0 );
DrawBase( );
glTranslated( 0, 2, 0 );
glRotated( -90, 0, 1, 0 );
DrawLowerArm( );
glTranslated( 0, 2, 0 );
DrawUpperArm( );
glTranslated( 0, 0, 4 );
DrawHammer( );
```

Hierarchical Models

- Two important calls:
  - **`glPushMatrix( ):`** Save current transform matrix
  - **`glPopMatrix( ):`** Restore transform matrix to last pushed one
- If matrix stack has **M1** at the top, after **`glPushMatrix( )`**, **M2** is on the top, and **M1** is in the second position
- If **M2** is at the top and **M1** is in second position, **`glPopMatrix( )`** removes **M2**, makes it the current transform, and moves **M1** to the top
- To **pop** a matrix without error, there must have been a corresponding **push**

Scene Description language (SDL)

- Immediate mode graphics with OpenGL: A little tougher
- SDL: Example language for **retained mode** graphics
- SDL makes hierarchical modeling easier
- SDL data structure format



SDL (cont.)

- Easy interface to use
- 3 steps:
  1. `#include "sdl.h"`
     Add sdl.cpp to your Makefile
  2. Instantiate a Scene Object
     Example: `Scene scn;`
  3.
```
// read your scene
scn.read( "your scene file.dat" );
// build lighting data structure
scn.makeLightsOpenGL( );
// draw scene using OpenGL
scn.drawSceneOpenGL( );
```

Example: Table Without SDL

```
// define table leg
//-----------------
void tableLeg( double thick, double len )  {
  glPushMatrix( );
    glTranslated( 0, ( len * 0.5 ), 0);
    glScaled( thick, len, thick );
    glutSolidCube( 1.0 );
  glPopMatrix( );
}

// note how table uses tableLeg
void table( double topWid, double topThick,
            double legThick, double legLen )  {
  // draw the table - a top and four legs
  glPushMatrix( );
    glTranslated( 0, legLen, 0 );
```

Example: Table Without SDL (cont.)


```
    glScaled( topWid, topThick, topWid );
    glutSolidCube( 1.0 );
  glPopMatrix( );

  double dist = 0.95 * ( topWid * 0.5 ) - ( legThick * 0.5 );
  glPushMatrix( );
    glTranslated( dist, 0, dist );
    tableLeg( legThick, legLen );
    glTranslated( 0, 0, -2*dist );
    tableLeg( legThick, legLen );
    glTranslated( -2*dist, 0, 2*dist );
    tableLeg( legThick, legLen );
    glTranslated( 0, 0, -2*dist );
    tableLeg( legThick, legLen );
  glPopMatrix( );
}
```

Example: Table Without SDL (cont.)


```
// translate and then call
glTranslated( 0.4, 0.0, 0.4 );
// draw the table
table( 0.6, 0.02, 0.02, 0.3 );
```

Example: Table With SDL

```
def leg {
  push
    translate 0.0 0.15 0.0
    scale 0.01 0.15 0.01
    cube
  pop
}

def table {
  push
    translate 0.0 0.3 0.0
    scale 0.3 0.01 0.3
    cube
  pop
```

Example: Table With SDL (cont.)

```
  push
    translate 0.275 0.0 0.275 use leg
    translate 0.0 0.0 -0.55 use leg
    translate -0.55 0.0 0.55 use leg
    translate 0.0 0.0 -0.55 use leg
  pop
}

push
  translate 0.4 0.0 0.4 use table
pop
```

Examples

- Hill contains useful examples on
  - Drawing wireframe models (example 5.6.2)
  - Drawing solid models and shading (example 5.6.3)
  - Using SDL in a program (example 5.6.4)