Moving to 3D

Shiffman Chpt. 14

The Mysterious Third Dimension…

2D to 3D

- Add a third dimension, Z, corresponding to depth
- Must change the rendering mode to a 3D mode
  - JAVA2D is the default
  - Two 3D modes can be set:
    size(<x>, <y>, P3D);
    size(<x>, <y>, OPENGL);
  - OpenGL is hardware accelerated. (May need to import the OpenGL library)

2D to 3D

- Some commands naturally extend. e.g.
  - can use translate(x, y, z);
- Rotation becomes more complicated
  - Rotation in 2D is one DOF
    - Rotate around axis perpendicular to the screen
  - Rotation in 3D is two DOF
    - Can rotate around each of the x, y and z axes

Transformations

- rotate(<radian>);
  - rotates around the origin, about the Z axis
  - May need to translate first to rotate an object around the desired pivot (e.g. to spin it in space).
- e.g. rotate rect(100,20, 20, 300); 45 degrees:

translate(-100, -20); *//assuming rectMode(CENTER)*
rotate(radians(45));
translate(100, 20);

Transformations

- rotate(<radian>);
  - rotates around the origin, about the Z axis
  - May need to translate first to rotate an object around the desired pivot (e.g. to spin it in space).
- Axis specific rotations:
  rotateX(<radian>);
  rotateY(<radian>);
  rotateZ(<radian>);

Transformations

- Rotation order matters
  - For example:

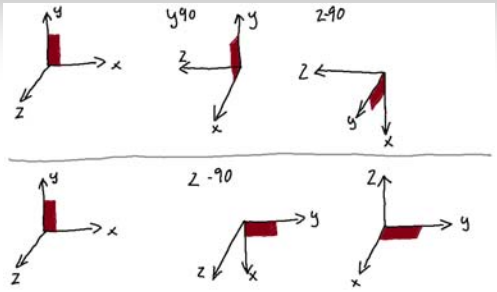
    rotateY(radians(90));

    rotateZ(radians(-90));
  - Is not the same as

    rotateZ(radians(-90));

    rotateY(radians(90));

Difference Between Two Rotation Orders



Class Exercise

- A laser gun is oriented to fire along the X-axis
- It needs to hit a target at (10, 10, 10)
- What rotation(s) must be applied to the gun in order for it to hit its target?

Transformations

- scale(<float>);
  - Uniform scale in all dimensions
- scale(<float>, <float>, <float>);
  - Vary the scale in each dimension

Transformations

- Transformations are stored in a matrix
- Can save and restore state
- pushMatrix();//saves the current state
- popMatrix();//retrieves the previous state
- (same as we did in 2D)

Simple Explosion

- Much research has been done on physically simulating explosions for special effects
- This just uses randomness to create a basic approximation, as follows
  - Represent object by a grid of small objects
  - Translate and rotate to add movement
  - Apply some randomness to create variation

```
translate(0, 0, count); //translates in z
 for(int i =0; i< numRows; i++)  {
  for(int j = 0; j<numCols; j++)    {
    pushMatrix();
    //for each of x and y, specify the object offset, the offset of the
    //small object within the block and a random offset that grows
    //over time
    translate(offsetX + (1-transFactor1[i][j])*count + i*sWidth,
        offsetY + j*sWidth + (1-transFactor2[i][j])*count);
    rotateY(radians(speedFactor[i][j]*count));
    rotateX(radians(speedFactor[i][j]*count));
    rotateZ(radians(speedFactor[i][j]*count));
```

3D Drawing Primitives

- Processing offers two
  - box()
  - sphere()
- Draw wireframe
  - stroke(), noStroke()
- Object color
  - fill()
  - Basic version. More advanced options later.

3D Computer Graphics…

- … is like a movie set
- Have
  - Models/objects (a set)
    - Models are covered with different materials
  - Lights
  - Cameras
- Computer computes a 2D image as would be seen from the camera, given the models, their materials and the lights

Lights

- Create shading on the sufrace
  - Default lights
    - lights()
  - Ambient
  - Directional
  - Point Light
  - Spot Light
  - Specular color

Actual 3D Objects

- Previous examples involved 2D (planar) objects with 3D transformations
- Can also create 3D objects

Basic Drawing

```
//single polygon
beginShape();
        vertex(50, 50);
        vertex(150, 50);
        vertex(150, 150);
endShape(CLOSE); //shape should be closed by
connecting the last and first vertex
```

Drawing Options

- beginShape(<type>)
  <type>: POINTS, LINES, TRIANGLES, TRIANGLE_FAN, TRIANGLE_STRIP, QUADS, QUAD_STRIP

  curveVertex(); //connected with curved lines.
-

Creation Process

- Specifying geometry in code is too time consuming for most models
- A common workflow is to design a model in other software (Maya, Studio Max, Blender) and import it into Proccessing
  - There are common image formats like .obj

Examples

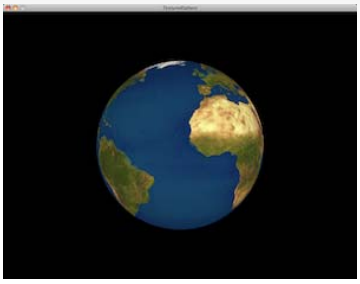
- Basic drawing
- Icosahedron

More Advanced Surface Control

- Material Properties
  - ambient()
  - emissive()
  - shininess()
  - specular()
- Shaders
  - Custom code that will control the appearance of the surface
  - Emulate various materials

Texture

- Can wrap an image around 3D geometry



Camera

- Can position, aim, tilt
- Change projection
  - Perspective
  - Orthographic