CS 378: Computer Game Technology

3D Engines and Scene Graphs
Spring 2012
What’s a 3d engine

- OGRE Core
  Class Structure
What are the objects?

- Geometry - polygon (triangle, quad) meshes
Hierarchical Modeling

- How can you make articulated characters move in the world?
  - Move the whole character wrt the world
  - Move legs, arms, head wrt body
  - Move hands wrt arms
  - Move upper vs. lower arm
  - Same for legs
Symbols and instances

- Most graphics APIs support a few geometric primitives:
  - spheres
  - cubes
  - triangles
- These symbols are **instanced using an instance transformation.**
Use a series of transformations

- Ultimately, a particular geometric instance is transformed by one combined transformation matrix:

- But it’s convenient to build this single matrix from a series of simpler transformations:

- We have to be careful about how we think about composing these transformations.

  (Mathematical reason: Transformation matrices don’t commute under matrix multiplication)
Connecting primitives
3D Example: A robot arm

Consider this robot arm with 3 degrees of freedom:
- Base rotates about its vertical axis by $\theta$
- Upper arm rotates in its $xy$-plane by $\phi$
- Lower arm rotates in its $xy$-plane by $\psi$

Q: What matrix do we use to transform the base?
Q: What matrix for the upper arm?
Q: What matrix for the lower arm?
Hierarchical modeling

- Hierarchical models can be composed of instances using trees or DAGs:
  - edges contain geometric transformations
  - nodes contain geometry (and possibly drawing attributes)

How might we draw the tree for the robot arm?
A complex example: human figure

Q: What’s the most sensible way to traverse this tree?
```cpp
figure()
{
    torso();
glPushMatrix();
    glutTranslate( ... );
    glutRotate( ... );
    head();
glPopMatrix();
    glutPushMatrix();
    glutTranslate( ... );
    glutRotate( ... );
    left_upper_arm();
glPushMatrix();
    glutTranslate( ... );
    glutRotate( ... );
    left_lower_arm();
glutPopMatrix();
    glutPopMatrix();
    ...
}
```
Animation

- The above examples are called **articulated models**:  
  - rigid parts  
  - connected by joints  
- They can be animated by specifying the joint angles (or other display parameters) as functions of time.
The most common method for character animation in production is **key-frame animation**.
- Each joint specified at various **key frames** (not necessarily the same as other joints)
- System does interpolation or **in-betweening**

Doing this well requires:
- A way of smoothly interpolating key frames: **splines**
- A good interactive system
- A lot of skill on the part of the animator
Scene graphs

- The idea of hierarchical modeling can be extended to an entire scene, encompassing:
  - many different objects
  - lights
  - camera position
- This is called a **scene tree** or **scene graph**.

![Scene Graph Diagram]