3D Graphics Software Options

- From scratch (on top of)
  - 2D (or no) software / hardware
- Low-level package
  - cross-platform (e.g., OpenGL)
  - vendor-specific (e.g., Direct3D)
- High-level package
  - Typically built on top of low-level package
- Examples
  - OpenInventor
  - Java3D
  - Vtk
- Interactive development environment
3D Graphics Software Concepts

- Objects
  - Geometry and other properties (e.g., “material”)
- Lights
- Cameras
- Geometric transformations
- Renderer: Draws scene
- Interaction
- Interface to window system

Scene Storage

- Immediate mode (no)
  - App sends info directly to renderer
  - Graphics package does not retain scene structure
- Retained mode (yes)
  - Low-level (e.g., OpenGL)
    - Display list (stored list of graphics commands)
  - Scene graph (e.g., OpenInventor, OpenSceneGraph, OpenSG, Unity)
    - Hierarchical Directed Acyclic Graph (DAG)
    - Graph nodes represent geometry, properties, transformations, lights, cameras, ...(typically data)
    - Renderer traverses scene graph, app modifies scene graph as needed
      - App is composed of attached scripts in Unity
Game Engines

- Special purpose (for specific games or specific kinds of games)
- General purpose (e.g., Unity)
- Include a mix of
  - Graphics package functionality
  - Interaction
  - Physics (including collision detection)
  - “AI”
  - Visual effects
    - Environmental (fog, rain, sky, etc)
    - Particle systems (fire, smoke, …)
  - Audio
  - Networked collaboration
  - Animation
  - Model loading from “standard” formats
  - GUI components
  - Support tools (coding/modeling/…)
  - IDE

Unity