CPSC 484 – Graphical Programming
3d Game or Simulation – Final Project

The final project for this class is to produce a 3d simulation or game.

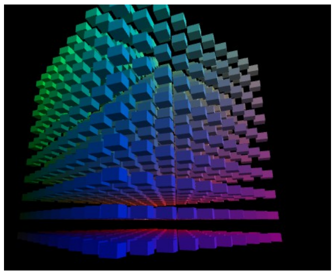
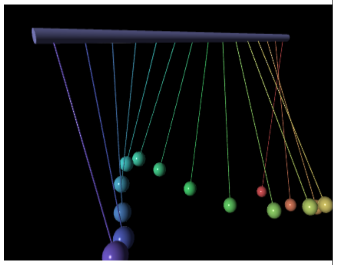
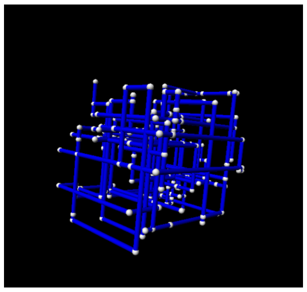
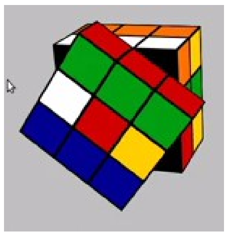
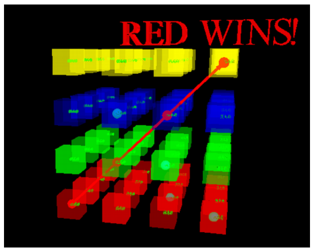
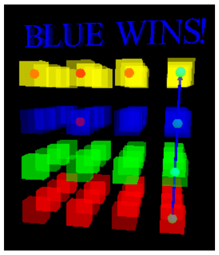
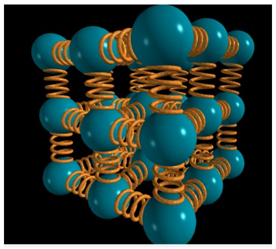
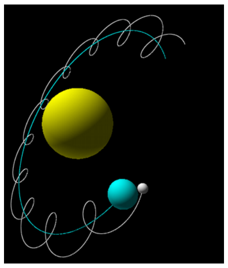
3D simulations must involve the 3D motion of multiple objects, rotation in one or more dimensions, and follow physical rules such as the flocking behavior of birds, the motion of the planets through the solar system, the motion of leaves on trees, and so forth. The simulation should provide the ability to rotate and zoom the scene, to see additional views and details. It should also provide lights and shadows. Although true ray-tracing is difficult to achieve, your game or simulation will look more realistic if the lighting and shadowing effects are present.

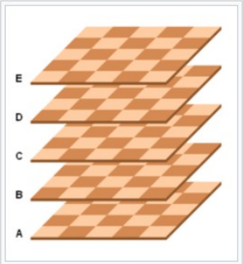
An example of a complex simulation would be that of the solar system. It should allow viewing the entire system (all eight planets), zooming in on single planets to see its distant moons, and zooming in further to see details of the closest moons, and the surface of the planet. It should also use astronomical data to accurately portray the eccentricity of the elliptical orbits, their inclination to the ecliptic, the periods of the planets and moons, and so forth.

2D games (such as 2d chessboards, othello, checkers, tic-tac-toe) are not acceptable. Semi-3D versions of games that simply add slight depth to a 2D game are also not acceptable. 3D versions of these games are acceptable, especially if they involve the ability to rotate and zoom on different aspects of the game (see examples below). For example, several 3D versions of Chess exist (usually called tridimensional chess) some with varying size boards, and some with boards that can be moved.

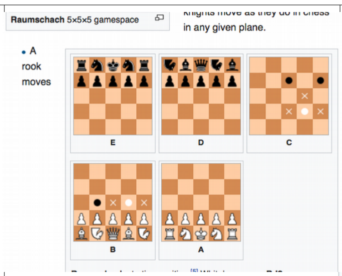
Some examples of possible games would be building a house of cards, 3D tic-tac-toe, Othello, Checkers or Chess, Rubik's cube, solving 3D puzzles, such as cubes that are put together with a variety of 3D pieces, having a game that is a simpler version of Crossy Road, platform games similar to Mario Brothers, Snake games in which a Snake moves through a 3D maze, and projectile games involving the motion and targeting of projectiles through space.

OpenGL, C++ toolkits that invoke OpenGL, or Python wrappers of OpenGL, such as Visual Python, are all acceptable environments for developing your game or simulation.



Raumschach 5×5×5 gamespace


Raumschach 5×5×5 gamespace

in any given plane.

- A rook moves

3D chess on *Star Trek* (from the episode "Court Martial")

The Tri-D chessboard

Note: Although each step of this project is fairly straightforward, thinking in three dimensions definitely involves more complex logic than thinking in two dimensions. There are also far more combinations to test. Also, there is a learning curve involved in using whatever three-dimensional development platform you choose. Do NOT wait until the last couple of days to begin this project.

As always, your code MUST compile and run correctly.
Put all of your source code, and the corresponding output file, into a SINGLE PDF file, and submit that to Titanium.
All source files must include the name of the student creating it.

This project is due by 2355 (not 2359) on Saturday, 20 May 2017 at the latest. There will be NO extensions given if you miss this date. Make sure you upload your code at least 2-3 hours before the deadline.