Space Simulator - Project 02

Evan Cummings and Joshua Bartz
CSCI 576 - Human Computer Interaction

March 10, 2014

Description

For our project idea, we would like to develop a 3D rendered engine via Python's interface to OpenGL. As it stands, we have a 3D particle simulation and an efficient way to calculate forces between particles and such. From this simulation, we would first like to create a space simulator - complete with gravity between planets and other space bodies and thrusters on the spacecraft which create rotational and directional momentum. With the addition of a rear engine and front braking thrusters, this would effectively simulate a real space craft.

Once the thruster part of the engine has been created, it would be interesting to add a missile with the same thruster system which could seek out and destroy asteroids or other space objects which may be in the spacecraft's way. Next, multiplayer could be developed where two opponents might chase each other through an asteroid field or planetary system by following some kind of particle trail.

Long-term development outside of the scope of this class includes random generation of planetary systems from a series of differential equation solvers which the user will run in the background. These simulations may take several hours to complete, but will allow the user to explore these new planetary systems with their spacecraft. From here, it would be interesting to add the option for planetary evolution, i.e., randomly generated atmosphere, oceans, and mountain ranges. The user may then enter the atmosphere with their spacecraft, complete with physics calculations and potential failure. This is a major highlight of the game as it would allow the user to generate and explore an entire solar system, all with the power of their home PC. In a sense this would be similar to entering into the computer's imagination, a completely unique experience.

Intended users

The intended users of the system are:

1. Anyone who enjoys flying is space.
2. People who enjoy the experience of speed.
3. People who enjoy competition.
4. Anyone interested in the physics of space.
5. People looking for a casual low-stress game.
6. People looking for a high-stress challenge.
7. People looking for a quick game.
8. People who enjoy wasting an entire day playing one game because it is so good.
9. People looking for a game they never tire of.

10. People who enjoy watching things explode.
11. People who wonder what it would be like to fire a rocket in space.
12. People who are technically-minded.
13. People who just want to have a good time.
14. Anyone who enjoys wireframe rendered 3D games.
15. People looking to escape from reality.
16. People who always wanted to be a space-person when they grew up but never could.
17. Younger people.
18. People who do not enjoy excessive violence in a video game.
19. Parents.
20. People who like flashing lights and colors that dazzle the senses.
21. People with a computer.

Usability requirements

The non-functional requirements for this game will be:

The game will be

1. fun,
2. relatively easy to learn,
3. realistic,
4. rewarding,
5. challenging,
6. artistic,
7. innovative,
8. visually pleasing,
9. surprising,
10. informative,
11. educational,
12. awe inspiring,
13. worth paying money for,
14. able to make the *user* feel artistic,
15. full of infinite possibilities,
16. replayable,
17. affordable,
18. easily executable from an PC created within the last decade,

Functional requirements

High priority:

The users of the game should be able to

1. navigate easily between menus and the game screen,
2. configure the controls,
3. pause the game and return to the main menu,
4. fly the spaceship,
5. rotate the spacecraft 360 degrees,
6. move the spacecraft side to side,
7. change the speed of the spacecraft in all directions,
8. control the thrust of the main engine,

9. visually see their current speed and acceleration,
10. control the craft with a mouse and keyboard,
11. choose from a menu to start or stop a game or configure settings,

Medium priority:
The users of the game should be able to

1. avoid the gravitational pull of planets and asteroids,
2. have a map and a way to navigate,
3. aim by moving a targeting reticule,
4. know their distance from their selected destination,
5. control the craft with a joystick,
6. adjust the sensitivities of the mouse or joystick,
7. invert either axis of input for the mouse or joystick,
8. fly through a nebula and see color changes related to velocity,

Low priority:
The users of the game should be able to

1. dodge space debris,
2. see on-screen vectors for moving objects,
3. fly through rings that guide the ship into orbit of a planet,
4. scale the game's complexity to increase the framerate of the game,
5. display the framerate of the game,
6. choose from several different spaceships with different attributes,
7. customize their own spacecraft from attributes, i.e., speed, maneuverability, or color,
8. select from a list of rendering effects to increase framerate,

Design critique 1 feedback

Evan's feedback:

- *Provide a goal other than simply flying around in space.*

  Decision : We plan on implementing an obstacle course and entering into orbit around a planet. We will also be able to launch missiles at space debris and learn about space and physics. Also, the challenge with the simulator lies in simply avoiding collisions with planets and other space debris.

- *Include a clear navigation system.*

  Decision : We will indicate the current speed, direction, and acceleration of the spacecraft in a corner of the screen, as well as a compass. Also, tiny particles will be included to give the player a sense of speed as they move through empty space.

- *Show the player statistics of the space vehicle such as speed.*

  Decision : New statistics and feedback options need to be explored.

- *Provide player customizations for the spacecraft.*

  Decision : We would like to allow the user to apply attributes to the craft such as maximum speed and maneuverability.

- *Make it possible to scale the difficulty and realism.*

  Decision : We would like to scale the size of the space to render and quality of visual effects, as well as a 'simulator' mode providing the most realistic space movement and 'arcade' mode with simplified controls.

- *Explicitly educate the players on the physics of space.*

  Decision : Ideas for this include Nebula color changes to particles as they gain kinetic and potential energy with a colorbar which indicates current energy in Joules. Also, we would like to show the player a path which places the spacecraft in orbit around a planet or star. This will include a visual indicator of direction and speed.

- *Challenge the player to put their spacecraft into orbit around a planet.*

  Decision : Rings may be included which the player could use to guide the spacecraft through while increasing or decreasing speed in order to obtain a stable orbit.

- *Model the known solar system.*

  Decision : In the time scales relevant to our game, this may be a challenge. We have discussed increasing time and providing near- or at-light speed travel. The physics of the planets may also be included in this simulation with gravitational forces acting between each planet and the spacecraft.

- *Allow the user to launch probes into space to investigate other systems.*

  Decision : While an interesting idea, the expanse of space would be difficult to model within the course of the class, but if there was an efficient way to randomly generate these other systems, this is an interesting option.

- *Provide a "missile cam" to guide missiles through obstacles.*

  Decision : Totally cool – If we have time, this will be included.

Josh's feedback:

Menus

The menu system has been adapted from Josh's design crit one and uses a common video game configuration with buttons accessed with the mouse or joystick. An explanation of the individual screens are presented below.

Menu storyboards

Figure 1: Start screen upon initialization. From here the player may either start a new game, load an existing game, edit the configuration of the game, or exit.

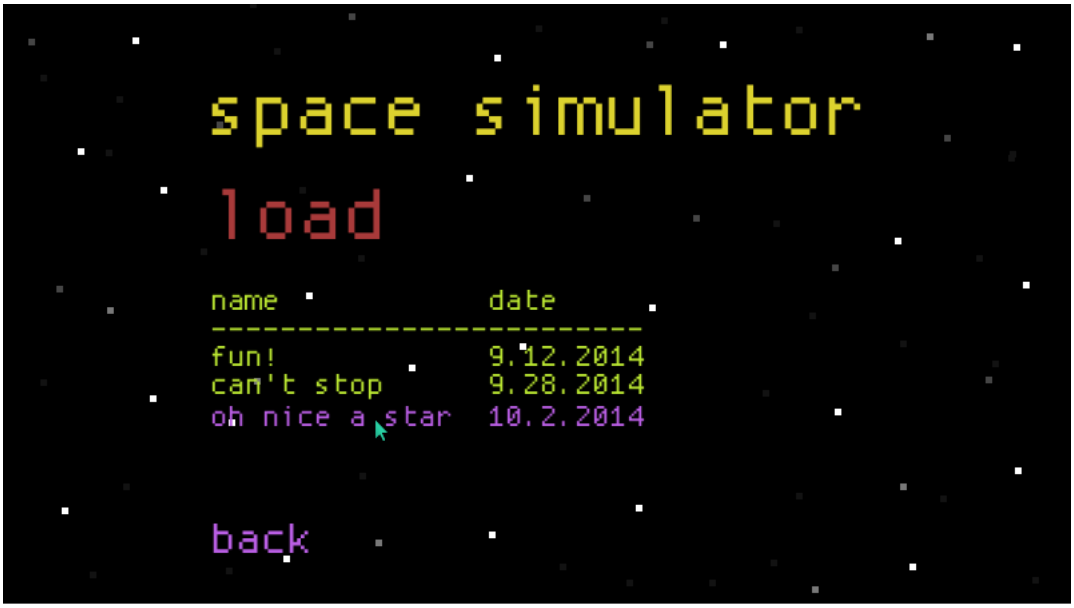

Figure 2: Load screen enabling a player to load a previously-saved game.

Figure 3: Pause screen accessed at any time by pressing the “esc” key.

Figure 4: Save screen. The player can name the save game and press “save” to save.

Figure 5: Config for global settings with access to control config screens. The player clicks on the parameter which increments the setting. Once any setting has been changed, nothing more is required of the player.

Figure 6: Mouse config, the user may select from a list of logical actions for both left and right buttons.

Figure 7: Joystick config. User may set the button by pressing the desired joystick button or moving the joystick in a particular direction.

Figure 8: Keyboard config. User may set the button by pressing the desired key.

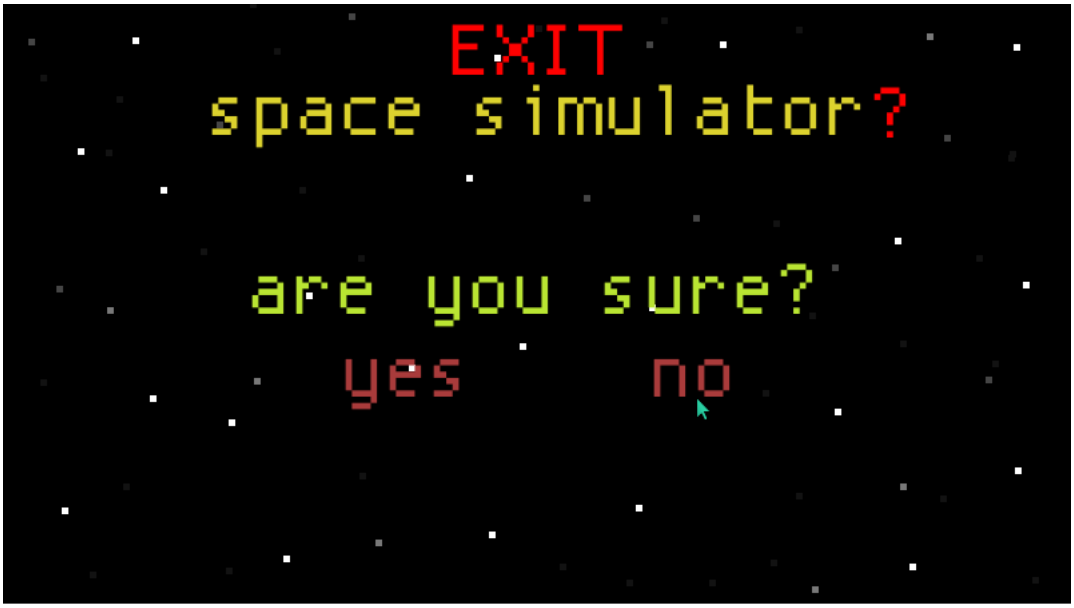

Figure 9: Exit screen confirms the player's desire to exit the program.

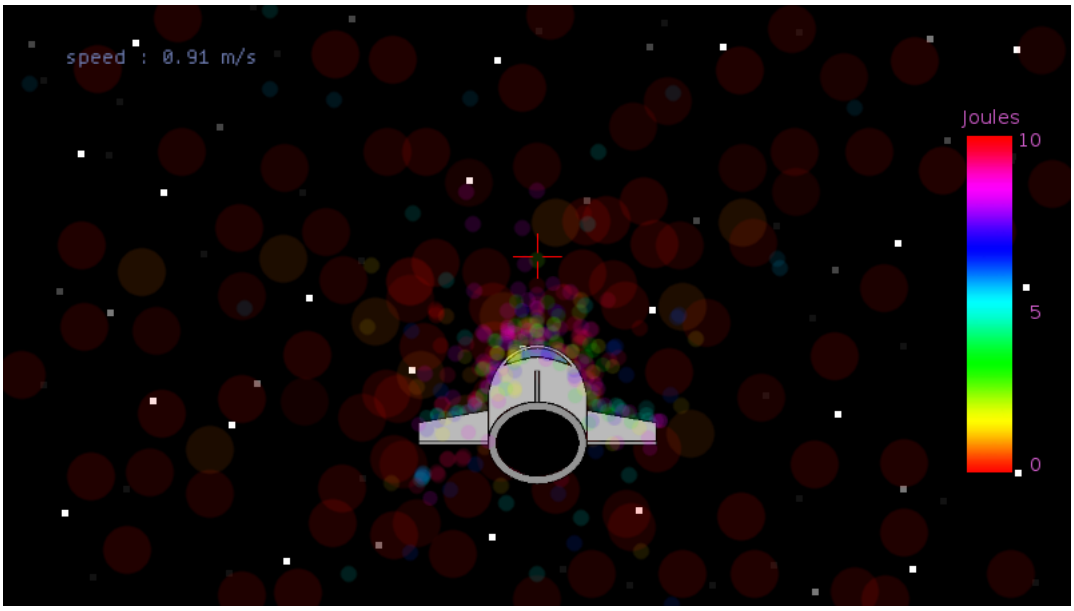

Figure 10: Here we have the spacecraft flying into the nebula, with a colorbar to show the energy level of the nebula particles. The red 'X' in the middle is a crosshair used for guiding the ship and firing rockets when implemented. The speed of the craft is shown in the top left corner. Notice that the color of the nebula changes as the craft pushes nebula gas particles out of its path.

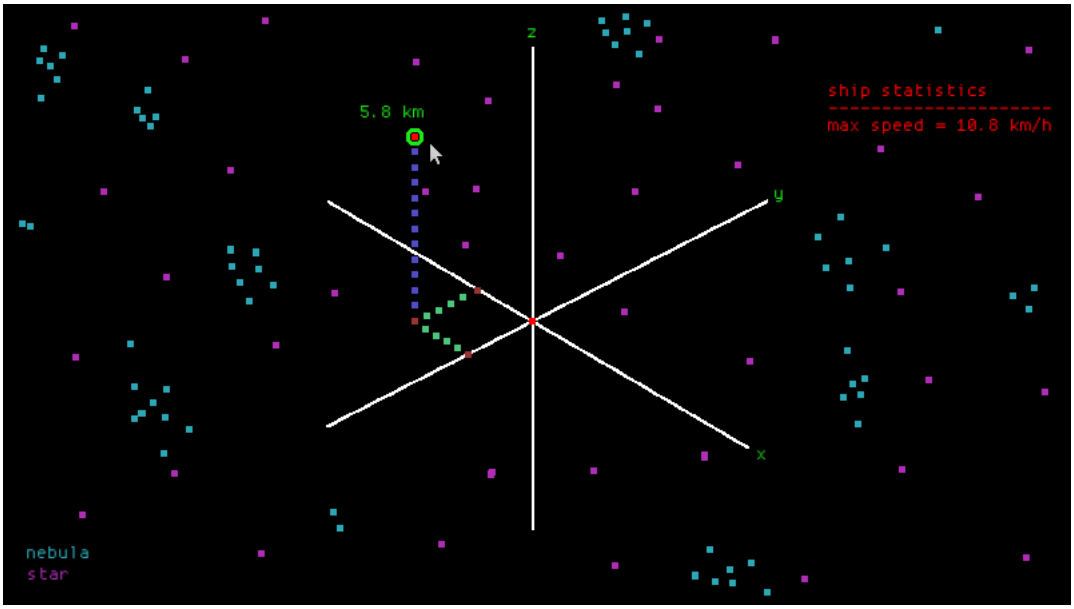

Figure 11: Map screen - the player selects a planet (purple) or a nebula (blue) to see the distance to it. The ship is located at the red dot in the intersection of the axes. Once a destination is clicked, the green circle appears around the planet. This green circle will remain in the flight screen as well to enable the player to navigate to their chosen destination. The view may be rotated by depressing the right mouse button and dragging the mouse up and down, left and right. Extra ship statistics are also shown to the left of the screen, here with only the max attainable speed of the craft.

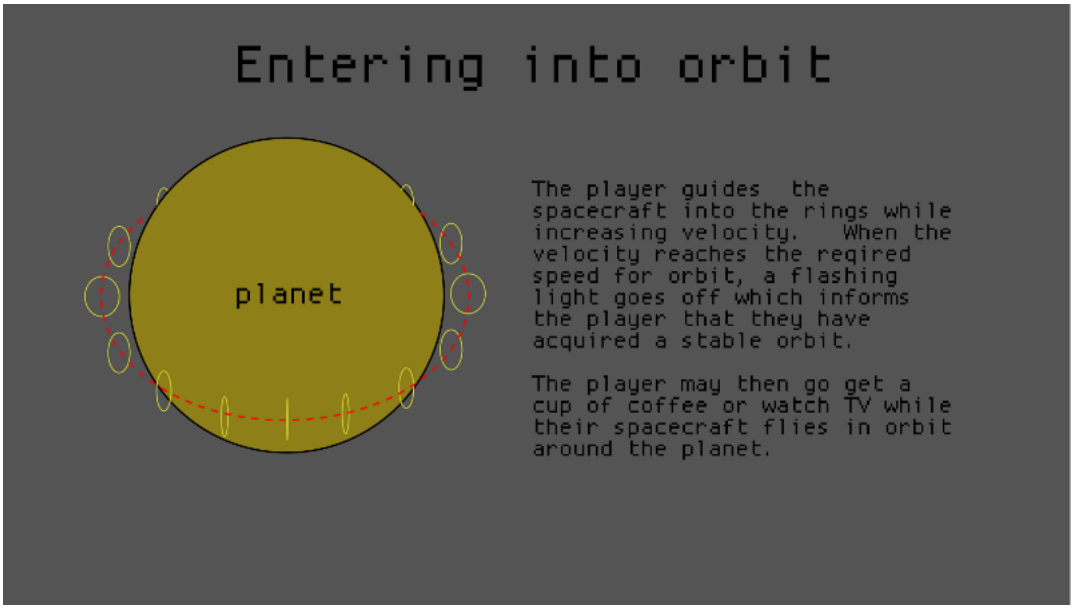

Figure 12: Concept for entering the spacecraft into orbit.