Assignment 4

CS 381: The Game Development Pipeline
Spring 2015
Max Score: 100

Assignment

You will work on handling multiple entities and use oriented 2D physics to control how entities move in your developing game engine. Please use the posted solution to assignment three as the basis for this assignment.

0.1 Multiple entities and selection

Select and load at least one example of each of the ten (10) different types of ship models (entity types) at `http://www.cse.unr.edu/~sushil/models/381/` into your evolving game engine. Bind the tab key to selection - that is - pressing the tab key will round-robin select the "next" entity. A selected entity, and only the selected entity, has a visible axis aligned bounding box.

You will notice that these entities are mostly ships and boats. Since boats and ships move on the surface of water, you will only need to deal with two-dimensional motion in our game engine's three dimensional world. Experiment with the different ways of texturing your ground plane with water.

0.2 Oriented Vector Physics

The arrow keys (or the numpad keys) on your keyboard control the selected entity. Up/Down will increase and decrease **speed** in the current direction of motion. Left/Right arrow keys turn the entity to the entity's left and right respectively. That is, arrow keys change the Entity's desired heading and desired speed. Ships and boats must move realistically and make smooth turns. Large ships turn more slowly than small boats. The aircraft carrier is the fastest ship in the ocean. Clearly, your physics aspect must now implement more realistic physics.

0.3 Camera control

You will continue to control the camera with the WASD and PgUp and PgDown keys. In addition, you will now also use, the Q and E keys to control camera yaw and the Z and X keys to control camera pitch.

0.4 Quitting

Hitting the escape key should gracefully shut down your running game engine.

0.5 Design contraints

Here are design constraints you MUST follow. I assume that your `_createScene` function is in `as4.py`

- You will create one instance of an `EntityMgr` class in `_createScene`. The `EntityMgr` manages entities. This means the `EntityMgr`
  1. maintains a list (or map or array) of all entities in your game engine
  2. creates entities and assigns them a unique identifier
  3. tracks which entity is currently selected
- The `Entity` class from the posted solution to assignment three will need to be modified to hold
  1. speed, heading
  2. desiredSpeed, desiredHeading
  3. acceleration, turningRate
- Each entity type will need a seperate class, each of which inherits from our `Entity`. All these classes can be in one file (`ent.py`) and do not need to be distributed one class per file.
- Each entity will now have two aspects
  1. Physics - you can modify the existing `Physics` class for this. For an entity, if the entity's desired speed is not the entity's actual speed, the physics aspect changes the entity's speed using the entity's acceleration as the rate of change of speed. The same process applies to a selected entity's desired heading, heading, and turning rate. Once speed and heading are updated, you can compute the vector velocity. Now you can apply our old familiar code to update the position (`pos = pos + vel * dt`).

2. Renderable - you will create a new class `Renderable`. Renderable manages the scene node (or nodes) associated with our `Entities` and on every tick, copies the position **and heading** from our `Entity` to the `Entity`'s scene node. Look up how to set the `yaw` for a scene node in the Ogre Manual.

- You must use Python Ogre's 3D vector class.
- You will create and use two FramelListeners.
  1. One frame listener will call our entity's tick method. This should inherit from `ogre.FrameListener`
  2. The other frame listener will handle keyboard input

These constraints will result in a cleaner, better, design for your emerging game engine. They will also increase your understanding of oriented physics, python-ogre, and python.

0.6 Extra Credit

This is cumulative!

- Add mouse selection (+5)
- Add islands in your waterworld (+3)
- Texture and light all meshes well and hand in a document that provides a step by step tutorial on how to texture a model in blender, and export it for ogre (+10)
- Add group mouse selection (+10)
- Add the ability to use standard RTS game mouse commands in order to have a selected (left-clicked) ship or boat to intercept another (right clicked) ship (+10)

1 Turning in your assignment

Assume that this format will be used for all your laboratory assignments throughout the semester unless otherwise specified.

1. Demonstrate your working program in the lab on the due date.
2. In lab, submit your code using a submission program (5 points).
   (a) Make a subdirectory in your home directory named `as4`.
   (b) Place all your project files in as4 (you will demo from this folder in lab).
   (c) Type `~sushil/bin/subAs4` while you are in this folder
   (d) Submission will open on the due date

Ask me (sushil@cse.unr.edu) if you have questions.

Objectives

1. Demonstrate an ability to apply knowledge of computing, mathematics, science, and engineering by learning and applying knowledge of Python to solve a problem (1)

2. Demonstrate an ability to analyze a problem, and identify, formulate and use the appropriate computing and engineering requirements for obtaining its solution (5)

3. Demonstrate an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice (11)

4. Demonstrate an ability to apply design and development principles in the construction of software systems or computer systems of varying complexity (13)