Assignment 5: An actual drone with enhanced camera

Out: 2nd March 2019 **Due: 13th March 2019 at 11:59pm**

In this assignment you will extend your program from Assignment 4 by replacing the three-axis rendering of the drone camera with an actual drone, and adding a second building to the scene. You will also add the capability to the drone camera to zoom in and out of the scene.

Part 1 : Second building (20 points)

In this part, you must add another building to the scene. This building has to have at least 3 floors and must have least 2 windows/doors per floor. The only constraints on the structure of this building is that it should look substantially different from the first building, and must have at least one facet of its structure that is non-rectilinear (i.e. not be box-like).

Part 2: Drone (40 points)

In this part you will replace the three-axis rendering of the drone with an actual model of a drone.

Part 2.1: Modeling

The drone model must have the following characteristics:

1. It must have a cylindrical portion for the camera. The camera must point in the forward direction. The cylinder does not have to move as the camera is rotated.
2. The drone must have at least 2 propellers. Each propeller must have at least 3 blades.
3. The drone must have a structure that holds a light (you will add a light here in the next assignment). The light must also point in the forward direction.

Part 2.2: Animation

In this part you will animate the drone in the scene. The main animating element must be that the propellers must move plausibly, to show that the drone is indeed flying. Here are some hints on how to create this animation:

1. In the Scenegraph class, there is an (empty) method called animate(float t). It can be called from the View with a representation of time.
2. Look at the scene graph design overall to see how you can support animation. You are not required to change the XML specification to add animation data: you are allowed to produce the animation programmatically.

You are allowed to change the design of the scene graphs as needed so that the scene graph can be animated when drawn. If you decide to add/edit any existing classes, be sure to consider whether the design is appropriate. You will be graded not only on what we see, but also how you implemented it in code!

Part 3: Zooming (20 points)

Give the capability to only the drone camera to zoom in and out with the + and – keys. The drone's camera has a zoom lens (i.e. it can zoom while remaining stationary). The camera should have limits of zooming in and out (i.e. you should not let your camera get into degenerate states if the user presses the above keys too many times). All other camera controls should work exactly as in the previous assignment.

Part 4: Design of animation (10 points)

Include a README file that explains your design to support animation in your program. Your design should answer the following questions:

1. Which parts of the design were changed to support animation, and why?
2. What are the limitations/compromises in your design?
3. If another developer uses your code to animate a different scene, how would they go about it? Be specific: which parts of the code would be changed for which purpose.

Extra credit (5 points)

Make an animation video of your final rendering (in mp4 or avi format). The video should showcase your achievement in this assignment. Look at the description on blackboard and the example code.

Program preparation

The program setup should be as before: load the scene using the config text file, and give the ability to the user to switch cameras in PIP mode and allow keyboard control of the camera.

What to submit

Submit the IntelliJ/Qt project folder set up correctly with your scene files you used as a zipped file (make sure you include the README file). Also include some screenshots that showcase your scene (even if you are not attempting the extra credit).