Design Documentation: Helicopter Project Milestone 1

Group:

Nam Thach

Dylan Rush

Yachan Wang

Milestone 2 update March 27th, 2014

Introduction

For the purpose of this milestone, the problem we are trying to solve is a helicopter movement problem. Mainly the power of the throttle of the helicopter. Right now there are only controls that allow the helicopter to increase and decrease rotor speed. We would like the helicopter controls to do more than just that. We would like to have the helicopter to be able to go to 0 throttle immediately and have a neutral control of the throttle where the force of the propeller will be set equal to the force of gravity upon the helicopter. Where if the tilt is 0, then the helicopter would essentially be floating in the air. More Problems will be implementing viscous air friction towards the movement of the helicopter.

Current Status

The current state of the helicopter program is that it will allow users to increase and decrease the rotor. Need to implement controls to have a "Neutral mode" and "set rotor to 0". As well as add extra obstacles to maneuver the helicopter around. The tilt on the helicopter is already set to have a maximum of 15 Degrees; but air viscous is yet to be implemented.

Team Roles

Dylan Rush: Implementing extra controls of the helicopter

- Change Rotor Increase and Decrease controls to 1 and 2
- Implement '0' key to reduce the rotor speed to zero
- Implement '3' key to set the rotor to a "neutral mode" where force of rotor will equal, in magnitude, to the force of gravity.
- Design software to make it easy to change specific keys used.

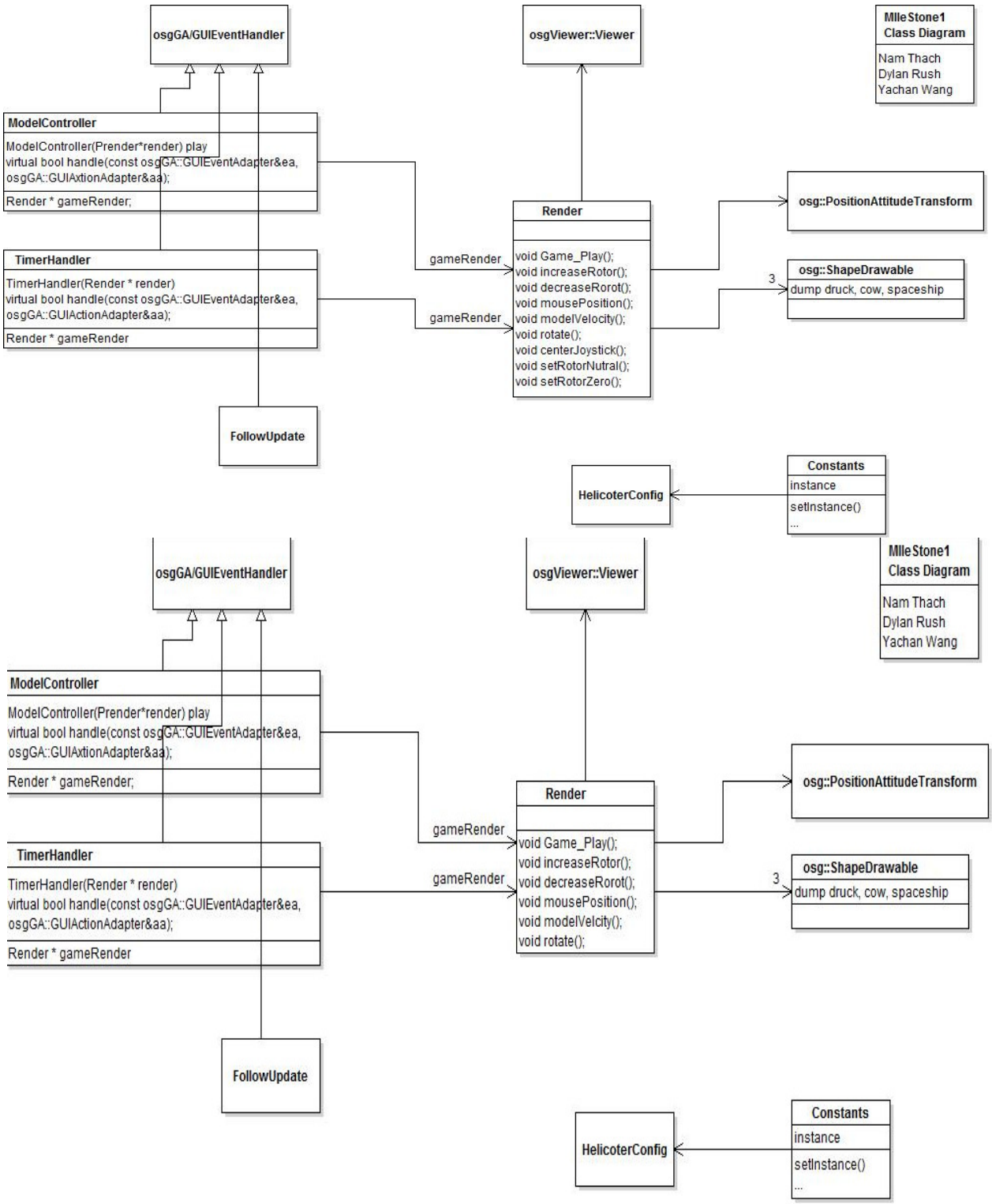
Nam Thach: Change terrain

- Add obstacles to the course
    - Reused balls from milestone warm-up
    - Added dump truck, cow, and spaceship to obstacle course
- Design Document

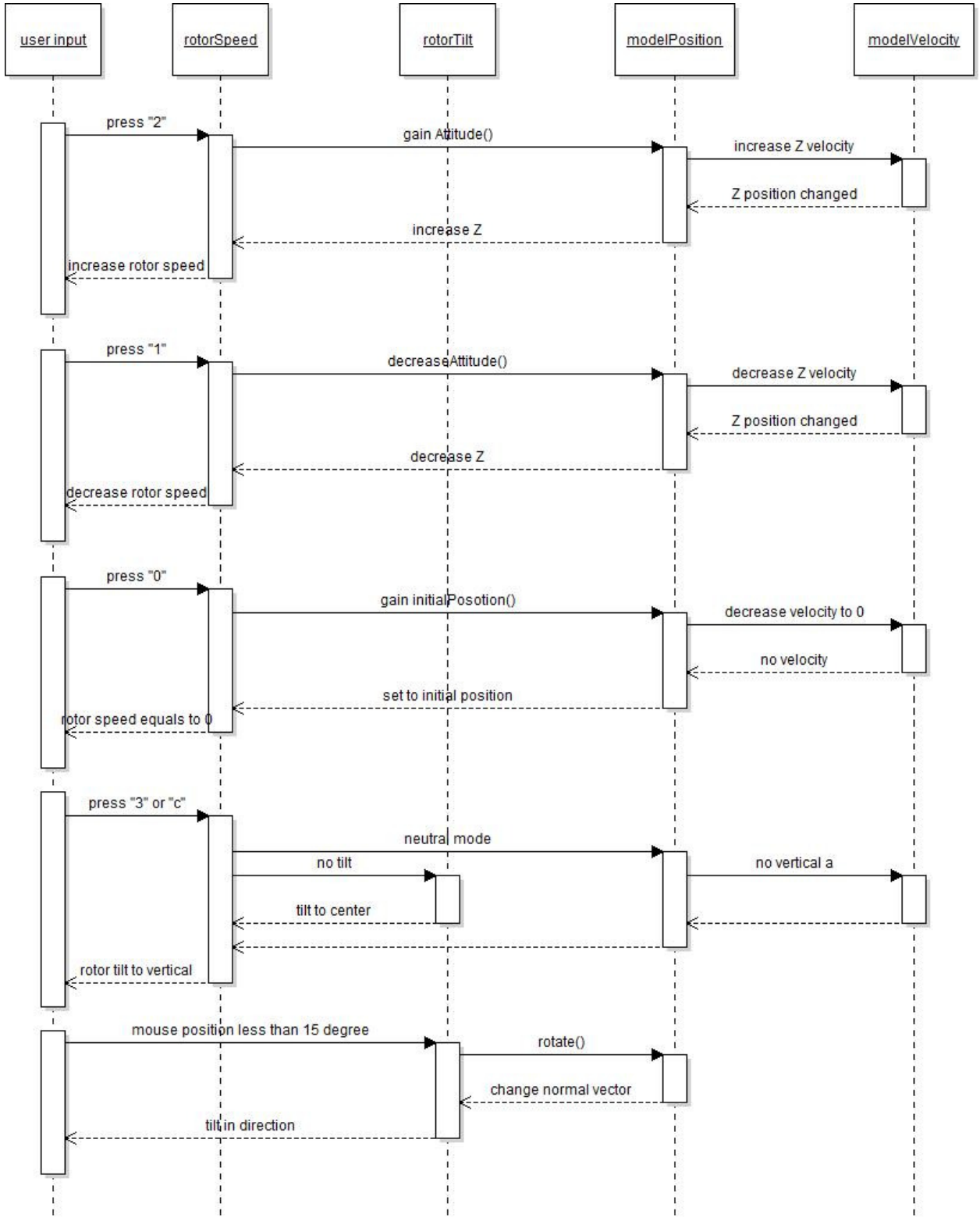
Yachan:

- Implement Viscous air friction coefficient
- Create new UML models and diagrams

UML Diagram


osgGA/GUIEventHandler
osgViewer::Viewer
MileStone1
Class Diagram
Nam Thach
Dylan Rush
Yachan Wang
ModelController
ModelController(Prender*render) play
virtual bool handle(const osgGA::GUIEventAdapter&ea,
osgGA::GUIAxtionAdapter&aa);
Render * gameRender;
TimerHandler
TimerHandler(Render * render)
virtual bool handle(const osgGA::GUIEventAdapter&ea,
osgGA::GUIActionAdapter&aa);
Render * gameRender
gameRender
gameRender
Render
void Game_Play();
void increaseRotor();
void decreaseRorot();
void mousePosition();
void modelVelocity();
void rotate();
void centerJoystick();
void setRotorNutral();
void setRotorZero();
osg::PositionAttitudeTransform
osg::ShapeDrawable
dump druck, cow, spaceship
3
FollowUpdate
HelicoterConfig
Constants
instance
setInstance()
...
osgGA/GUIEventHandler
osgViewer::Viewer
MileStone1
Class Diagram
Nam Thach
Dylan Rush
Yachan Wang
ModelController
ModelController(Prender*render) play
virtual bool handle(const osgGA::GUIEventAdapter&ea,
osgGA::GUIAxtionAdapter&aa);
Render * gameRender;
TimerHandler
TimerHandler(Render * render)
virtual bool handle(const osgGA::GUIEventAdapter&ea,
osgGA::GUIActionAdapter&aa);
Render * gameRender
gameRender
gameRender
Render
void Game_Play();
void increaseRotor();
void decreaseRorot();
void mousePosition();
void modelVelcity();
void rotate();
osg::PositionAttitudeTransform
osg::ShapeDrawable
dump druck, cow, spaceship
3
FollowUpdate
HelicoterConfig
Constants
instance
setInstance()
...

UML Sequence Diagram



Summary

During this milestone we felt good as a group going forward as everyone does there part in a efficient manner. We feel confident going forward, but getting to know OSG and being able to use all of its functionality could prove a bit troublesome. As far as the milestone 1 goes, everyone accomplished what was assigned but bonus functionality was missed to be implemented, for example; collision with the new objects was not implemented as we ran out of time and we were not able to experiment with OSG as much as we would have like.

Milestone 2

The purpose of this milestone was to allow scripting to be handled by the user. As well as run cxxtests to make sure the simulation worked as planned and all of the position and velocity calculations were in order. As well we updated the code of the project to make everything look neater and more readable.

Team Roles

Nam Thach

- Clean up code and add code so testing would be easier
- Added Scripting code so when the user pressed ‘p’ the script would run
- Attempted to get testing to work

Yachan Wang

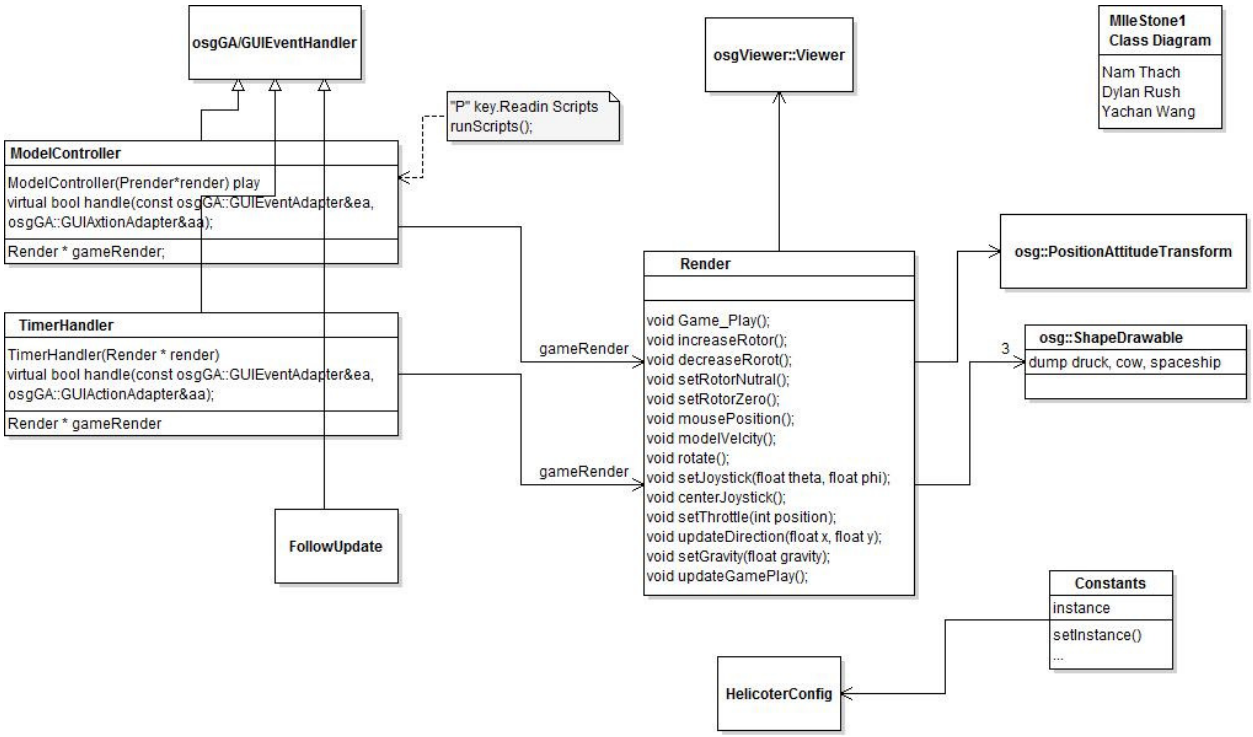
- Attempted to get cxxtest to work with visual studios,
- Created test code for the project
- Updated class UML diagram

Dylan Rush

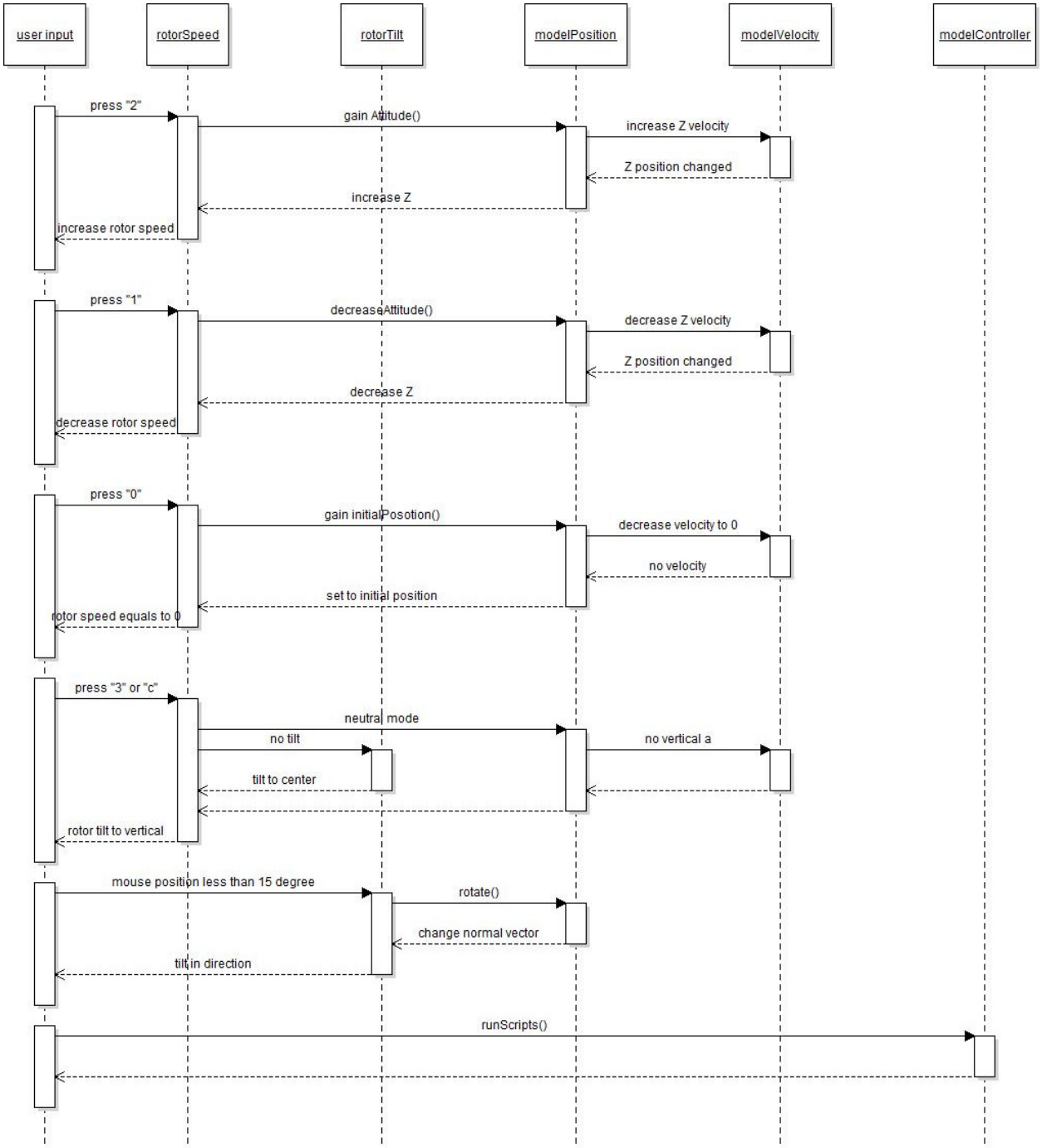
- Worked on Scripting code
- Attempted to get cxxtest to work

Milestone 2 Diagrams

UML Diagram



Sequence Diagram



Summary on Milestone 2

Milestone 2 was relatively a difficult task at the beginning when we were not as comfortable with code. But as the code started getting cleaned up, it became easier but not as easy as one would hope for. The scripting code was added to the model controller as when the simulation get an input of ‘p’ or ‘P’ the scriptrunner will run and the script that has the following commands:

throttle 15

delay 4

throttle 9

delay 2

throttle 10

#should be close to hovering

set_joystick 10 45

delay 10

centre_joystick

delay 30

end

After this was completed we disable the mouse as it was interfering with the input commands of the script. After the mouse was disabled every9one attempted to get cxxtest to function with the OSG library and visual studios. This was the hardest part of the milestone as it didn’t have anything to do with the code but rather the platform the simulation runs on would not cooperate with cxxtesting. Our testing code was created by Yachan:

```
void testLift()
{
     disable friction
     set_position (0,0,0);
     set_velocity (0,0,0);
     set_rotor_force (mass*g*1.1);
     delay (10.0);
     x = get_position();
     v = get_velocity();
     a = get_acceleration();
     TS_ASSERT_EQUAL();
     TS_ASSERT_DELTA(x.Z(),5.0*g, 1e-4);
     TS_ASSERT_DELTA(v.z(),g,1e-4);
     TS_ASSERT_DELTA(v.z(),0.1g,1e-4);
     restore friction
}
```

But we didn’t have a chance to even test it.

In conclusion we feel confident when it comes to the actual code to move onto the next and final milestone but if it has to deal with testing, that is another challenge in itself.

GitHub Repository Link:

https://github.com/Nammy1101/Helicopter_Project.git