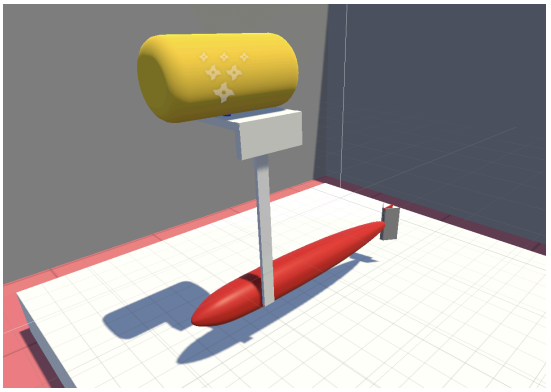
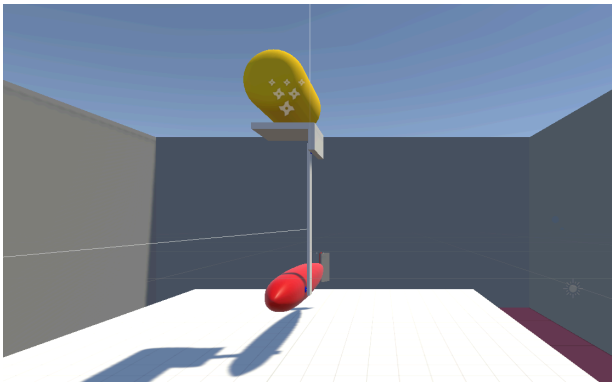
Introduction to Robotics: Project1 Report

Team 9
Name: 송우영, 양성지, 박정훈

1. Performance Record

Time [s]: 48.54
Number of Collisions: 0
Final Time [s]: 48.53896
Comments (optional):

- The servo motor does not start from 0 when the run begins.
- On Mac devices, there is an issue where the controller does not respond for about 10 seconds after execution, possibly due to the USB-C to USB-A adapter, after which it starts functioning normally.

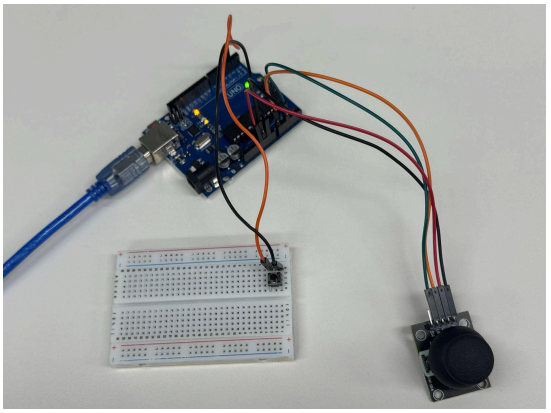
2. Participation Details

Describe which team member was responsible for each part. Note that multiple members may have contributed to the same part, but each member's individual contribution must be clearly specified. You may freely attach images below, if helpful.

2.1 Controller design

Explain how your team designed the controller.

Seongji Yang: Implemented exactly as described in the document.



2.2 Rocket design

Describe how your team designed the rocket, focusing on each component such as the body, engine, brake, and servo motor.

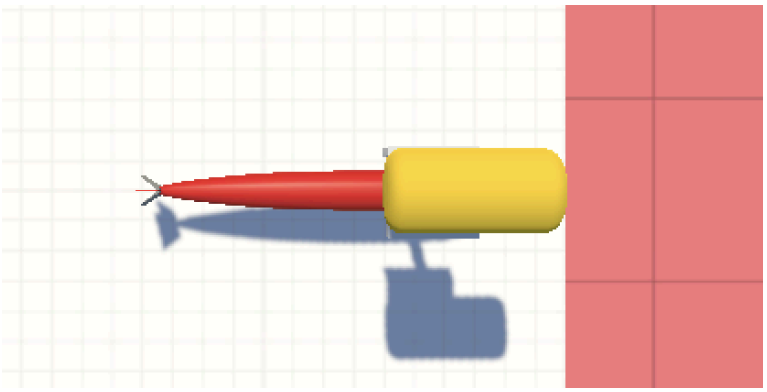
Wooyoung Song:

1. I first insisted of a rocket with fixed engine, whose direction was decided by additional two steering engine like real space craft. However, the performance of steering at sharp curve, for example linkage of T and A and top of A, had been lost ground. The reason of this was we granted only a small portion of thrust to steering engine, which causes pushing power weak and the body cannot make rotation as much as we wanted and also, recovery from rotation to straight was too slow, comparing with, dynamic engine.
2. From above speculation, I also made a claim to enlarge the number of rotation engine with small thrust, and we did lots of trials for placing the engines but, it cannot beat the one rotation engine

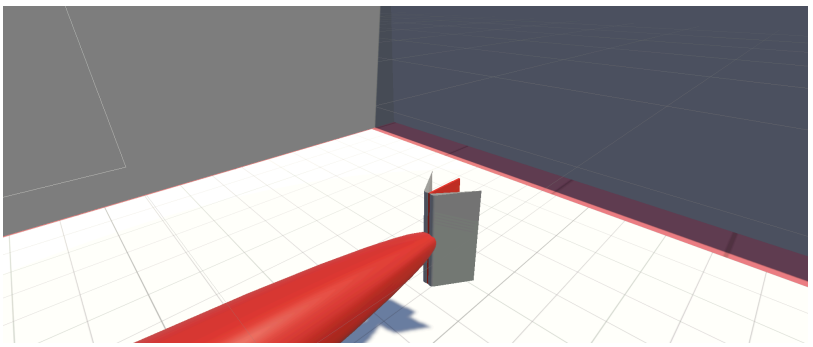
Seongji Yang:

1. When moving forward after rotation, the alignment between the long, straight body and the engine had a significant impact on performance. Therefore, we designed the rocket body to be long and positioned it as far back as possible so that it barely touched the goal line, allowing it to better utilize centrifugal force during turns. For the same reason, we placed the engine as far forward as possible to maximize the distance from the body's center of mass, improving rotational stability.

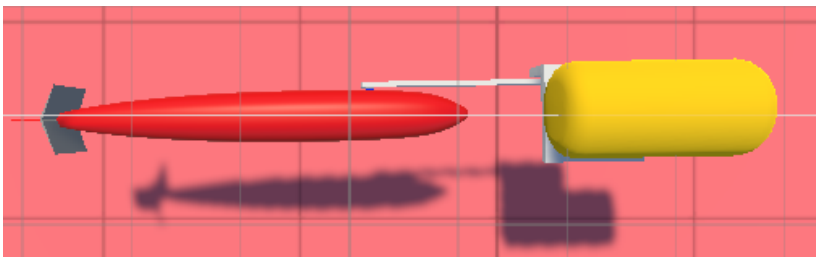
2. Since minimizing the travel path is advantageous, we avoided placing the engines on both sides of the body (which would increase distance from the wall) and instead installed a single engine at the center to shorten the path.



3. Similarly, we rotated the airbrake vertically rather than horizontally to prevent possible disadvantages in movement distance.



4. From 1 and 2, since more distance between body and engine means smoother rotation, we placed a servo motor at the side of the rocket, which was attached to a long frame towards the y-axis. Then attached the engine to that frame, and when we started, we lowered the frame to the front, making the engine stay in front of the rocket.



5. For 4, we needed the engine to always stay horizontal, thus adding a new servo motor and a frame to rotate the opposite of the servo motor attached to the rocket.

6. When the rocket reaches the end point, the timer stops only when it identifies the body. Therefore, we made the frame return to the original position when the joystick stayed neutral.

Jeonghoon Park:

1. Since the engine speed was not sufficient, I assisted in designing the optimal engine size by moving the rocket straight forward after launch and measuring the time until the first collision for ablation analysis. The results showed that weight had a significant effect on performance. We found that setting the maximum output of two engines to 60 (7.4 sec) each was less efficient than setting both to 30 (5.6 sec) , and later discovered that using only a single engine (5.6 sec) had the same effect.

2.3 Control system design

Detail how the control system was implemented in Unity/Arduino (e.g., mapping, noise management, input logics).

Wooyoung Song:

1. To increase the number of engines, I added lots of duplicated code for engines. Furthermore, as we can see that static engine weakened the performance of the space craft, adding more servors, it was made possible for engines to rotate the

same direction, but later, this implementation was discarded because, one big rotating engine shows great performance.

Seongji Yang:

1. Since using the brake naturally reduces speed, we decided not to use it. As a result, there was no need for a button input, so we implemented the rocket to move forward only.

Jeonghoon Park:

1. Since the engine experiences a significant speed reduction when its angle exceeds 90 degrees, a limit was imposed to restrict the maximum angle to 90 degrees.

```
float ang = Mathf.Atan2(y, x) * Mathf.Rad2Deg;  // -180..180
ang = Mathf.Clamp(ang, -90f, 90f);
servo[0].controlVal = ang;
```

2. On macOS, there was an issue where the controller input would not respond for about 10 seconds after startup. Also, At startup, noise caused the rocket's direction to shift slightly. By implementing a button-based engine On/Off toggle, I was able to start the engine precisely when desired while avoiding the inconvenience of having to hold down the button continuously.

```
if (D2 && !prevD2)
    engineOn = !engineOn;
prevD2 = D2;
if (engineOn){
    engine[0].controlVal = 1f;
    brake[0].controlVal = 0f;
}
else{
    engine[0].controlVal = 0f;
    brake[0].controlVal = 1f;
}
```

3. Discussion (optional)

3.1 What did your team learn from this project?

- We realized that even small designs, such as center of mass changes significantly affected the rocket's performance, as the results varied greatly with each modification.
- We also recognized the importance of influential factors such as weight. In real-world situations, additional external factors like air resistance and motion along the z-axis would come into play, making control much more challenging. This made us appreciate the importance of developing the ability to handle such complex conditions effectively.

3.2 What would you improve if you had more time?

1-1. from our current rocket, remove around 1 ~ 5 thrust from the (main) engine.

1-2. by making a new (mini) engine with that thrust, place it at the rear of the engine, under the airbrake.
1-3. This mini engine will be used for more efficient rotation. if the joystick points front, around -30 ~ +30 degrees of the head of the rocket, the mini engine will point the same direction as the rocket. Otherwise, the mini engine will point perpendicular to the rocket, since it will consider that the rocket will need to rotate.
1-4. This design also didn't consider retreat since like the airbreak, it will not be used due to time increase.

2-1. while saving the design of the servo motor and the frame it is directly linked, temporarily detach it from the servo motor that creates balance when the engine moves towards the head of the rocket..
2-2. add a new frame to the servo motor for balance. This frame will point to the end of the rocket the same way as the frame that points to the y-axis.
2-3. now add a new servo motor at the end of this new frame, and add the detached frame and engine to it.
2-4. Now, there will be 3 servo motor to make the engine head front and to keep balance.
2-5. if the first servo motor turns -90 degrees, the second servo motor will also turn -90 degrees, and the last servo motor will turn +180 degrees, creating balance.

3-1. Specifically, when the second button is pressed, the default forward angle could automatically adjust upward by a certain amount, allowing the engine thrust direction to better compensate for gravity. This would help the vehicle maintain optimal speed and responsiveness even in regions where gravity significantly affects motion.

Submition checklist:

- Performance report (.pdf)
- Performance video (.mp4, etc.)
- Resource file Unity package (.unitypackage)