lab2 Part II Report

黄玉安 11610303

Before

In this lab, run those command in a terminal to start the gazebo simulator.

```
roslaunch turtlebot3_gazebo turtlebot3_empty_world.launch
```

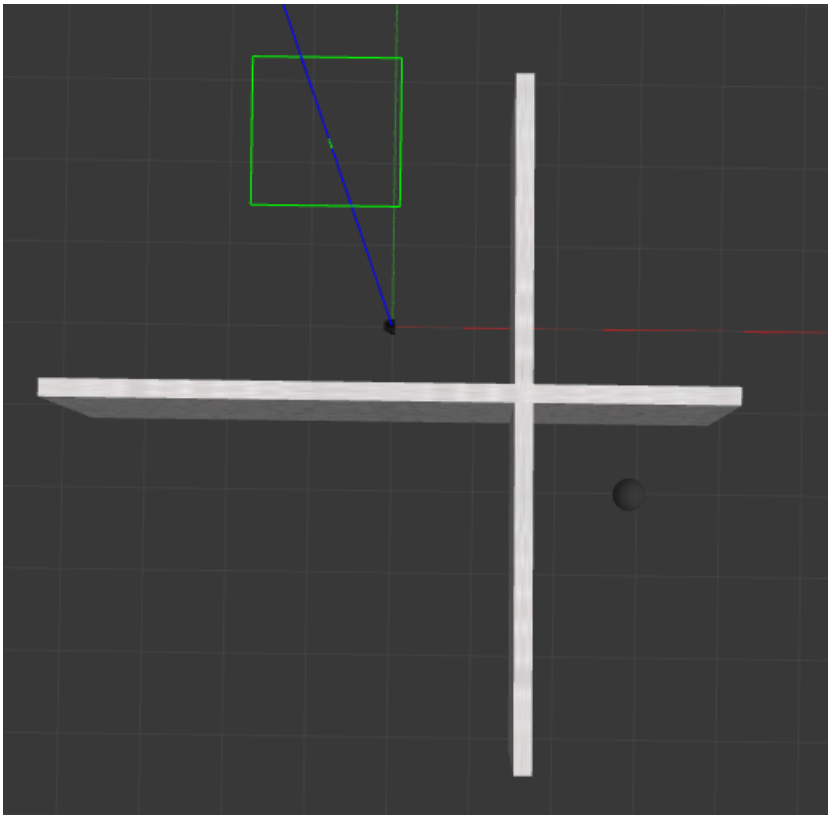
And before I did this, I have set some environmental variables in ~/.zshrc file. (I use terminal `on-my-zsh`).

```
source /opt/ros/melodic/setup.zsh
export ROS_IP=10.21.75.207
export ROS_MASTER_URI=http://10.21.75.207:11311
export TURTLEBOT3_MODEL=burger
export ROS_HOSTNAME=10.21.75.207
source ~/catkin_ws/devel/setup.zsh
```

By following the step provided in [Supplementary Document II], I can use MRST to communicate with tutlebot3 in gazebo.

Scene

This is scene in my gazebo.



There are wall to stop robot pass. The robot is initial in position (0, 0). Its goal is to reach the ball (in position (3, -2)).

Code:

This is my code. I also provided the whole .m script.

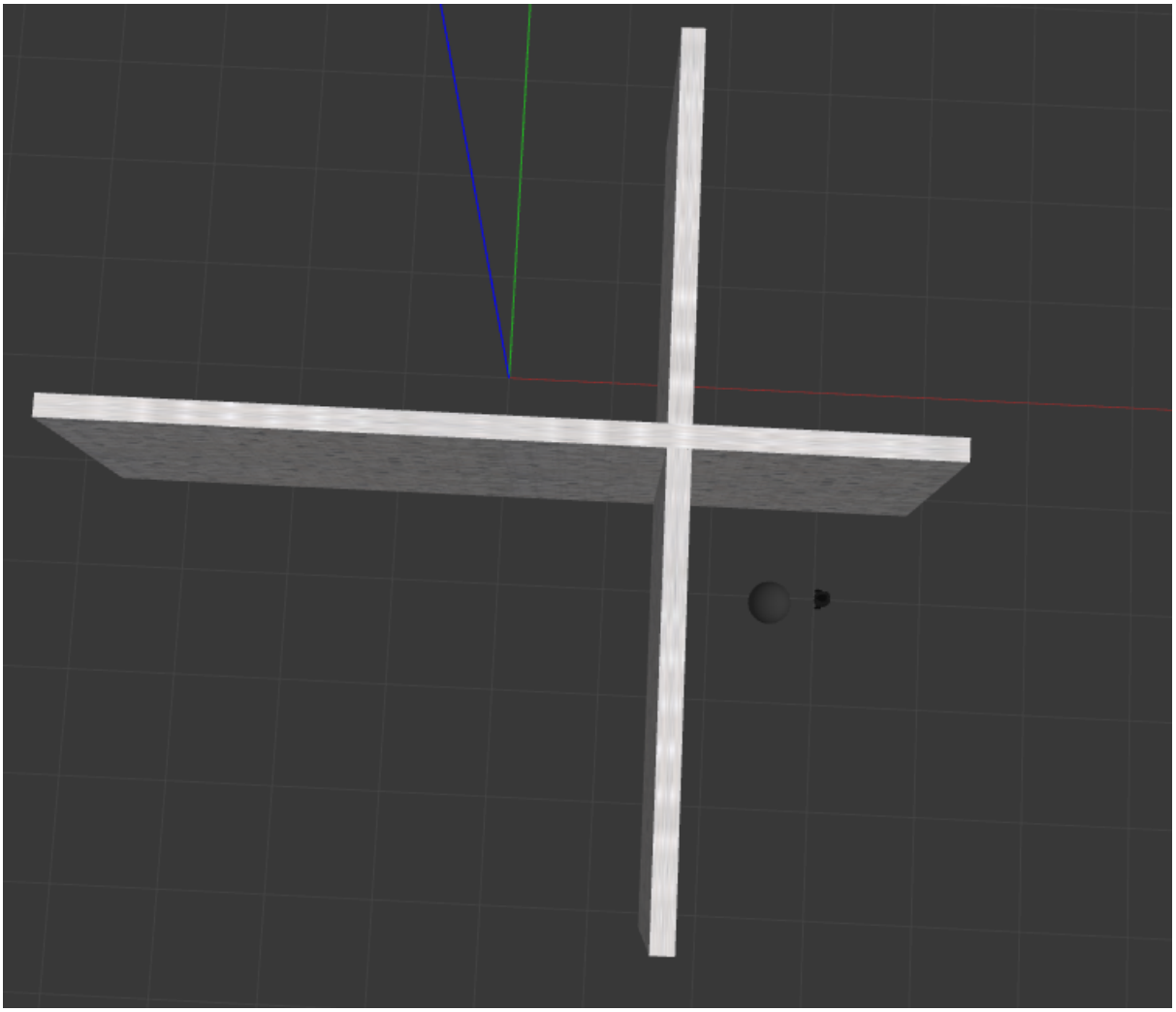
```
function test()
% ==========drawing map ====

    clear;
    clc;
    function gazebo_init()
       rosshutdown;
       rosinit('10.21.75.207', 'NodeHost', '10.21.75.207');

        setenv('ROS_MASTER_URI','http://10.21.75.207:11311');
        setenv('ROS_IP','10.21.75.207');

        linkStates = rossubscriber('/gazebo/link_states');
        stateData = receive(linkStates);
    end

% send velocity and angular speed to robot
    function control_gazebo(V, A)
        velmsg = rosmessage(velpub);
        velmsg.Linear.X = V;
        velmsg.Angular.Z = A;
        send(velpub,velmsg);
    end

% ================ using system control===========
path = [0 0;
        0 4;
        1 5;
        2 5;
        2 0;
        5 0
        5 -2;
        3 -2;
        ]

controller = robotics.PurePursuit;
controller.Waypoints = path;
controller.DesiredLinearVelocity = 0.5;
controller.MaxAngularVelocity = 0.5;
controller.LookaheadDistance = 0.5;

gazebo_init()


velpub = rospublisher('/cmd_vel');
odom = rossubscriber('/odom');

    function res = getPose()
        odomdata = receive(odom,3);
        pose = odomdata.Pose.Pose;
        x = pose.Position.X;
        y = pose.Position.Y;
        z = pose.Position.Z;
        quat = pose.Orientation;
        angles = quat2eul([quat.W quat.X quat.Y quat.Z]);
        theta = rad2deg(angles(1));
        res = [x y angles(1)]
    end
```

```
goalRadius = 0.1;
robotCurrentPose = getPose();
robotGoal = path(end,:);
distanceToGoal = norm(robotCurrentPose(1:2) - robotGoal);
controlRate = robotics.Rate(5);
axis([-1 6 -3 6]);
grid on;
scatter(robotCurrentPose(1), robotCurrentPose(2), 'bh', 'filled');
scatter(robotGoal(1), robotGoal(2), 'gp', 'filled');
hold on;

while(distanceToGoal > goalRadius)

    [v omega] = controller(robotCurrentPose);
    control_gazebo(v, omega);
    robotCurrentPose = getPose();
    scatter(robotCurrentPose(1), robotCurrentPose(2), 'r.');
    distanceToGoal = norm(robotCurrentPose(1:2)-robotGoal);
    waitfor(controlRate);
end
control_gazebo(0, 0);

end
```

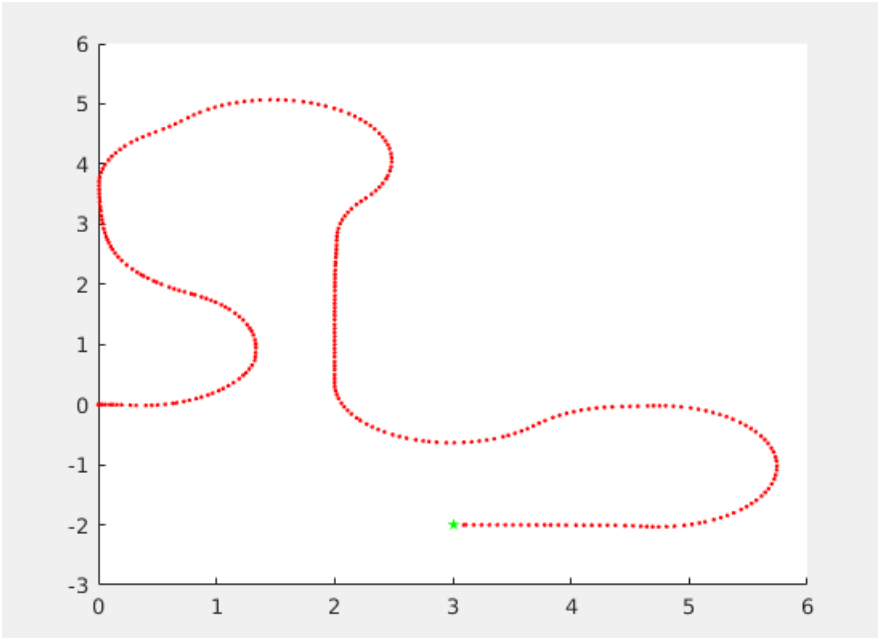
Using function `gazebo_init()` to setup connection with gazebo. Using `getPose()` to know the position and orientation of robot. Using `control_gazebo` to send velocity and angular speed information to robot. Using `robotics.PurePursuit;` to calculate.

Result:

This is my result



Robot finally reach the ball. I receive the position information from gazebo robot and draw the path on Matlab. It is shown below.



Green star is the goal point. Red point is path. It avoided obstacles.

Special case

Sometimes robot will slip (wheel rotate but robot do not move). In this case, try more times, and it would work.