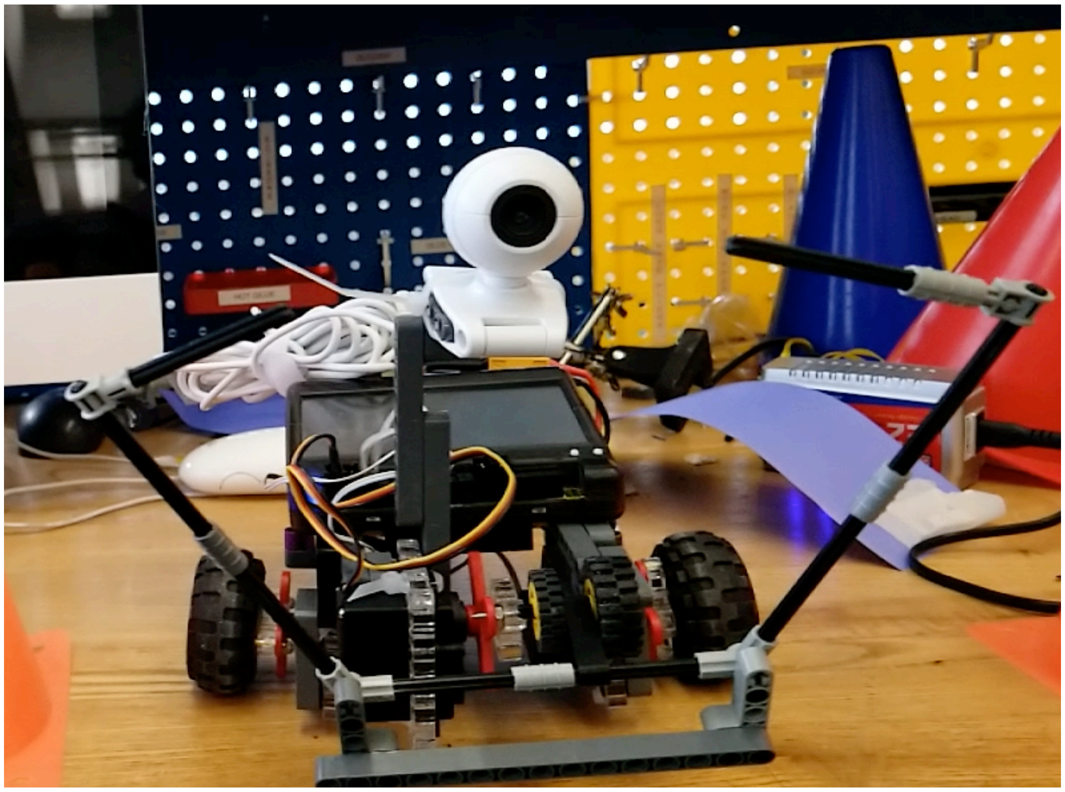
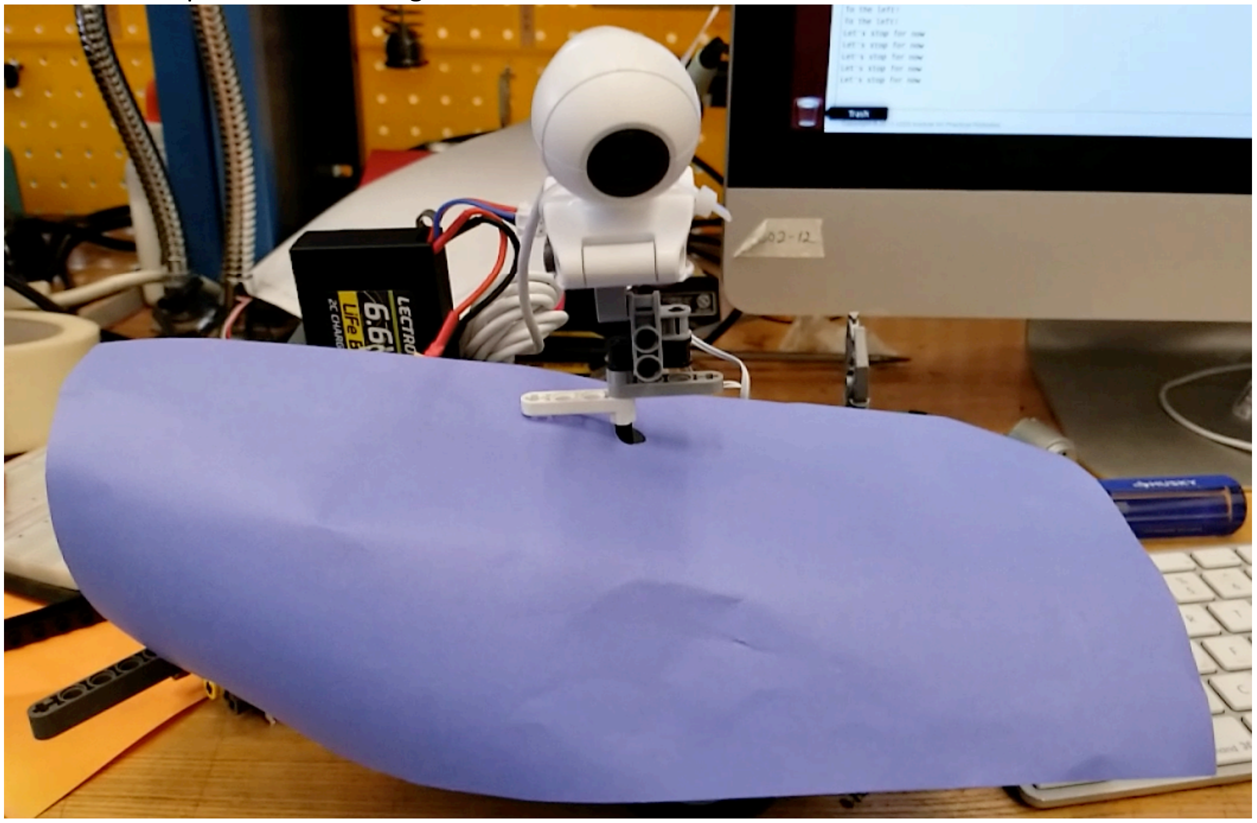
Tim Barber
Solomon Bitone
Mobile Robotics I Final Report
Video trailer of the bots in action: https://youtu.be/vDkqJ2yHYbs

1. The task for our final project was to create two robots that played soccer with each other. Our stretch goal was to have two robots that could interchange between goalie and kicker after a goal was successfully scored. We didn't reach our stretch goal, so instead we decided to permanently make one robot the goalie and the other the kicker. This was done because we needed more time to make sure the goalie and kicker functioned properly, so there was not enough time to try merging their roles. Since this was our stretch goal, we decided it was more important to have the kicker and goalie working separately. The kicker robot attempts to locate the ball and kick it into the goal, while the goalie tries to prevent the ball from entering the goal.

2. The kicker has two prongs in front of it that are attached to a servo. They face upwards, but once the robot is close enough to the ball they come down and attempt to hold it. The robot will "kick" the ball by quickly lifting its prongs, pushing the ball forward. We have a vision sensor in front of the robot as well, and this is used to locate the ball, goalie, and goals. Originally, the vision sensor was too short and it could not adequately see its surroundings. We adjusted the height and it fixed this problem. It had two motors with a wheel attached on either side to provide movement and steering. There is also a back wheel that provides stability and easier mobility for the robot as it moves around.



The goalie has two motors with a wheel attached to each one, allowing it to move left and right. It also has a vision sensor in front that is used to locate the ball. Underneath the vision sensor is a piece of colored paper. This is used by the kicker's vision sensor to identify the goalie. The goalie went through one complete revision in order to make it simpler. Originally, it's design was made to mirror the kicker because our plan was to have them swap roles. Unfortunately this was scrapped because we needed a working live demo, and the code for swapping the robot and kicker could not be made in time. Lastly, a servo motor was shoehorned in front of the goalie in hopes that the goalie would "kick" the ball back to the kicker once it saved a goal. Unfortunately, it does not work very well, so during our live demo I manually moved the ball back to the kicker so it could attempt to score more goals.



3. The robots used computer vision exclusively, so the code had to handle all aspects of playing soccer, from grabbing the ball to shooting the ball around the goalie. To do this, the code was split into three sections; detection, movement, and the overall logic. The detection portion of the code was dedicated to machine vision. Below is the pseudocode of the overall logic, the descriptions of each function can be found in a comment block above the functions in the actual code.

   The logic for the kicker is called on a loop after a new frame has been grabbed from the camera. The states were stored using global variables.

```
int resetBot = 0;
int ballDegrees = -1;
int foundGoal = 0;
int haveBall = 0;
int centeredGoal = 0;
int fooled = 0;
          void kicker(){
   if(resetBot == 1){
   if(in_goal_range() > 0)
   {
      TURN
   }
      else{
        BACKUP AND TURN
         resetBot = 0;
      }
         return;
   }

   if(haveBall == 0){
          track_color(0);
          //If the ball is close enough, we have the ball
         if(have_ball() == 1){
                   CENTER AND GRAB BALL
                   haveBall = 1;
         }
   }

          if(haveBall == 1){
            GO TO GOAL AND CENTER BETWEEN BOTH CONES
               }
               else if(fooled == 0){
                  MOVE TOWARDS BIGGEST GAP
               }
               else{
                 WE ARE FACING THE GAP, MOVE FORWARD
               }

               if(should_kick() == 1){
                  //kick ball
                    SHOOT BALL
                    BACKUP AND RESET
                    resetBot = 1;
               }
            }
}
```

The text overview is as follows: The robot searches for the ball, when it finds the ball it goes towards it, keeping it in the center of its vision. When the ball is determined to be close enough, it lowers the cage around the ball. It then begins to search for two orange cones. Once it finds both orange cones, it moves to the center of the two cones. It then looks for the biggest gap between the goalie and center cones. Once it lines up with the center of the biggest gap, it moves forward. When it determines that it is close enough, by either the cones being too big or the cones about to go out of frame (meaning the kicker will be lost), it moves the cage up, kicking the ball into the goal. The robot then turns around until it can't see any cones, then moves forward (assumed to be away from the goal), and goes back to looking for the ball.

For the goalie:

The robot is constantly trying to center on the ball, and it does this through a while loop. Here is the primary function used, center():

```
while(1)
{
if (cannot see ball)
        do nothing;
while(ball is centered)
        do nothing;
if(on left side of ball)
        move left;
if(on right side of ball)
        move right;
if(ball is within reach)
        kick ball away;
}
```

The code is simple for a reason. There have been 3 revisions of the goalie function, but they were all overly complicated. In the end, KISS was the focal point of the last revision (keep it simple, student). If the robot sees the ball is on the left side, move left. If the robot sees the ball is on the right, move right. If the robot is centered on the ball, stay put. This code is to ensure that the goalie can perform its job well. It was a single process.

4. See code attached

5. The robots behaved somewhat to our expectations. Our expectations were that the kicker would be able to determine where to shoot the ball, and when it was close enough to kick. This was not always the case, as the cones were sometimes partially detected, leading to the kicker thinking the cones were farther away than they actually

were. Another problem was losing sight of the cones or goalie. The logic was designed to retrace its steps when losing sight of either the goalie or the cones, but it could not account for every situation. Another problem was the varying lighting conditions and various object of similar colors in the room. Many times during our test runs, the difference between a failure and success was recalibrating the HSV to account for the change in lighting.
A problem that appeared later in our testing was camera corruption. The cameras supplied with the Wallaby are horrible. A slight bump will cause them to drop/corrupt frames for 10-20 seconds. This corruption was not picked up by the function that grabbed camera frames, so often a corrupted color would trigger the robot to do something that was not expected, like grab a ball that was not there, or shoot at cones that did not exist.
With perfect conditions, the kicker worked almost all the time.
The goalie's job was just to track the ball, which both robots could do very reliably in all situations. The only problem we had was the goalie moved too fast for the kicker. The goalie had the advantage of being able to move between the goal posts without having to turn. This caused the kicker to end up running into the goalie, as the gap between the goalie and cone virtually disappeared, and the kicker couldn't find the other gap because it was too close already. This also caused the cone to be obscured, reducing its size. All of this led to the kicker running into the goalie without kicking. We reduced the speed of the goalie midway through the run and we had more success, but the kicker rarely was able to get the ball past the goalie.

6. If we were to redesign the project, we would have built a better mechanism to hold the ball, as the ball would often fall out when turning. We also would have used better cameras that are more reliable and have a wider color range. Lastly, we would have used more powerful motors on the kicker to be able to outmaneuver the goalie, which has a much easier job when it comes to moving.

7. Overall this was my favorite course this semester, but there were a few things that I thought could have been done better. The most important being the final project.
I thought that we didn't have enough time to finish our final projects. I think that starting them sooner would have yielded better results. I put in 40 hours into the project and it took away a lot of time I had to spend on other final projects. If we were allowed to start sooner I could have spread that time out more evenly and had time to work on my other projects.
Second, I think that the Wallabys are a bad choice for this class. The Wi-Fi range is abysmal, moving it more than a couple feet away would render the Wallaby unusable until it was power cycled (taking about 5 minutes to fully boot back up). The IDE it uses lacks a lot of useful features that would have made coding easier. The IDE is more akin to a text editor. They were also bulky for what they were, so building a robot around them proved challenging. I feel that something else should be used going forward.

Lastly, I thought that the midterm was not an accurate reflection on the material we were given. I spent many hours studying the packet material we were given, but most of it was not on the exam, it was mostly slide questions. I would have preferred a more even mix of the material.