Botball® x FortLee2 (Team 210)

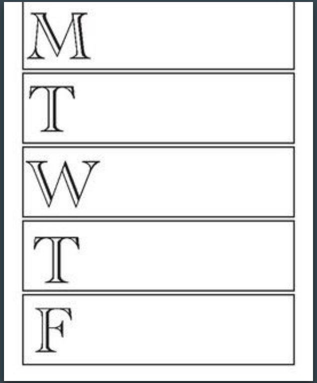
Presenters: Matthew Koretsky and Reghan Lao

Team Information

Meetings: in-class (every day of the school year) with occasional time allotted after school for additional work

Team Organization:

- In class, individuals and groups worked on assignments to get familiar with the robots and code
- Adult team leader Mr. Glebas organized projects and guided the team



Learning Goals for this Year

- Learn how to use KISS institute robots, including understanding how every external sensor and built in sensor works
- Learn to program robots and understand fundamentals of programming and engineering
- Learn about the process of trial and error in robotics
- Gain experience planning, documenting, experimenting, and reflecting on our process

Handling Conflict

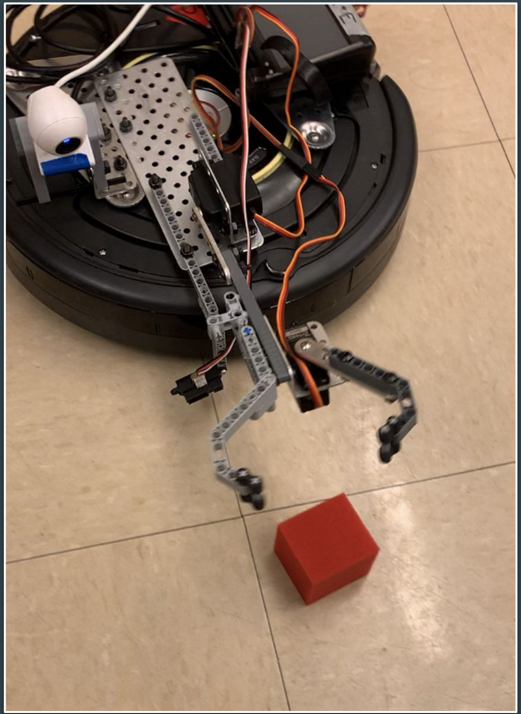
Example of team handling conflict:

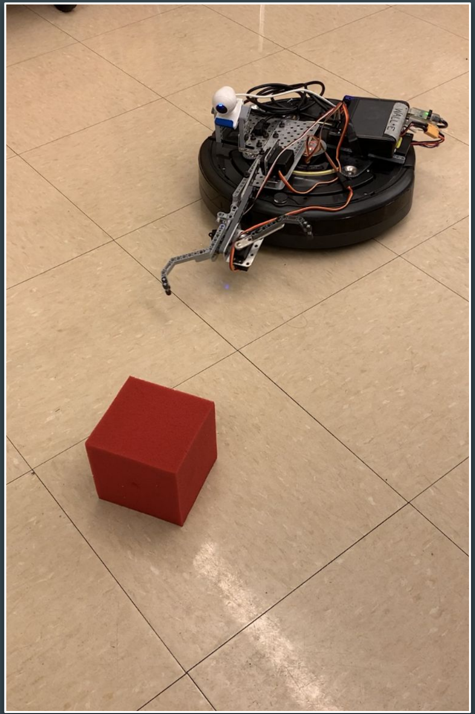
- Have a formal discussion regarding the conflict with sufficient evidence on both sides, if a team vote fails, consult adult team leader (Mr. Glebas), and if his resolution fails there will be a final vote, which would include the adult team leader (Mr. Glebas).
- An alternate method of handling conflict was the identifying of points of agreement and points of disagreement and forming a general consensus as a result. For example, if two parties agreed that a specific partition of code should be added/removed, but disagreed that a specific robot part should be included/excluded, a general consensus would be formed in regards to adding/removing parts of code at the cost of abandoning the robot parts conflict, ultimately dissolving the conflict.

Division of Labor

Work was divided into individual tasks and projects based on each sensor and programming concepts.

Started with small project involving simple movement using motors, and eventually learned to track and pick up a red cube using sensors and cameras.





Initial Game Strategy

Initial Game Strategy:

- Release and grab tennis balls with the wombat and put them into the transporter
- Grab Botguy with the Create and place in the starting box

Initial Design Concept:

- Wombat: claw used to grab tennis balls
- Create: build an arm that is long enough to grab botguy but still functional / not too heavy

Initial pseudocode:

- Wombat: Follow black tape line to tennis balls door, open door, follow black tape lines to transporter, pick up tennis balls and place in transporter while carrying the transporter
- Create: Draw an accurate path to Botguy, grab Botguy, place in starting box

Mid-Season Game Strategy

Mid-season grame strategy:

- Realized the Wombat was too slow to open the door to release the tennis balls, so we decided to make the Create do this instead
- Added a task for the Create to remove rings from the rock-a-stack and onto the side

Mid-season robot design:

- We swapped a servo on the Wombat with a mini servo to give the Create a way to open the door to the tennis balls
- Modified the front of the Wombat with legos to push pom poms into the transporter
- Worked on a versatile claw that could effectively grab rings

Programming status:

- Wombat: Added a function to get rings off the rock-a-stack
- Create: worked on drawing an accurate path to the rings and removing them properly

Final Game Strategy

Final Game Strategy:

- Due to inconsistencies with tracking tennis balls and grabbing rings, we decided to take these tasks out of our strategy and focus on getting points more consistently

Final robot design:

- Wombat: A sensor to follow the black line, a mini servo to attach the back of the transporter to the Wombat, a claw that would take rings off the rock-a-stack
- Create: An arm that could reach Botguy and grab accurately and a servo claw at the front to open the doors of the tennis balls

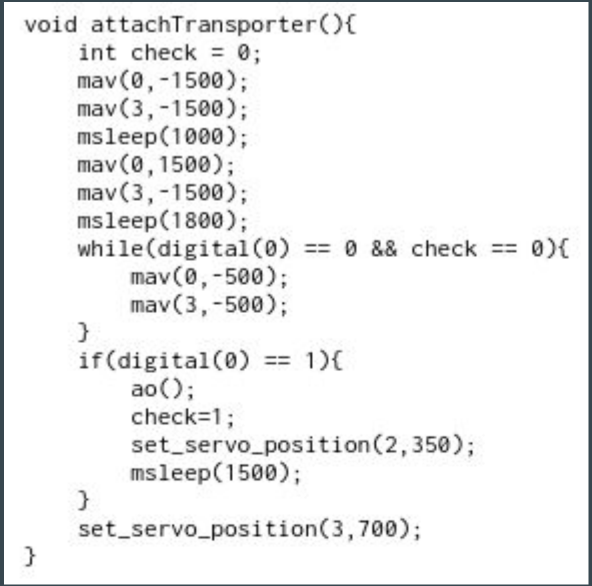
Final Programming Status

Final programming status

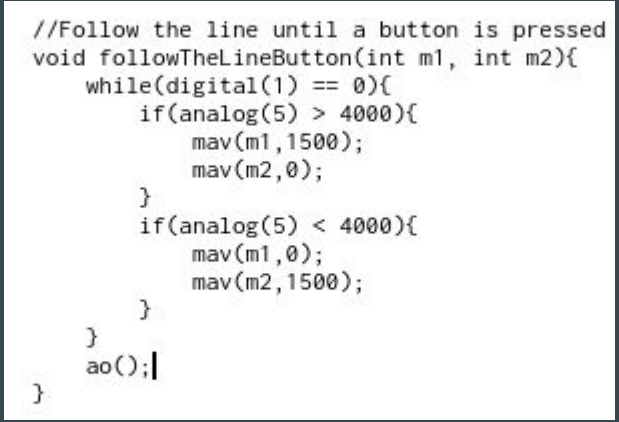
- Wombat: Take rings off the rock-a-stack, follow the black line to the transporter while pushing pom poms to the transporter, push pom poms into transporter and attach transporter to the back of the Wombat, carry transporter to the starting box.
- Create: Drives to the face the gate containing the various balls and drives towards the gate and stops at a certain range using the abilities of the E.T. range sensor. Gate containing the various balls are then lifted using a claw lift mechanism and various balls fall out. Botguy is obtained simultaneously and placed into another claw of the robot. After, botguy is ideally transferred to the starting box to score points.

Wombat Bot Code

Wombat code to attach the transporter using back servo (below)



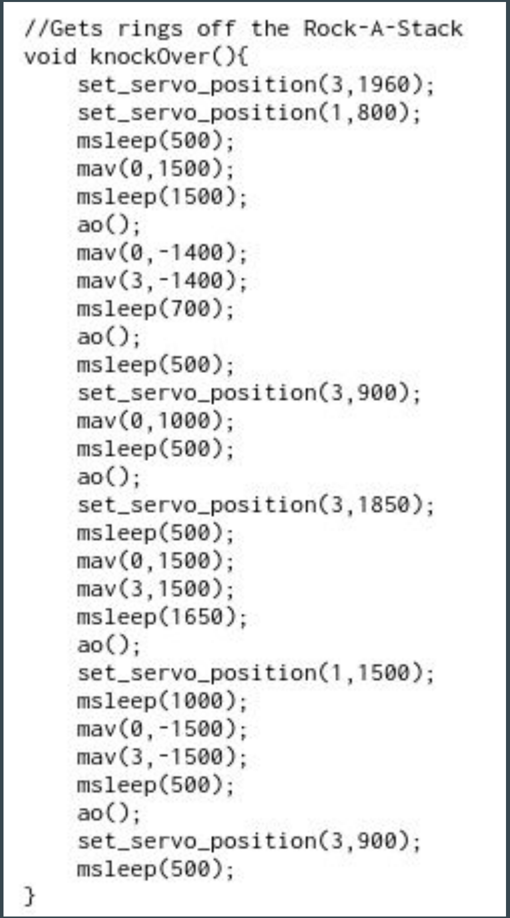
```
void attachTransporter(){
    int check = 0;
    mav(0,-1500);
    mav(3,-1500);
    msleep(1000);
    mav(0,1500);
    mav(3,-1500);
    msleep(1800);
    while(digital(0) == 0 && check == 0){
        mav(0,-500);
        mav(3,-500);
    }
    if(digital(0) == 1){
        ao();
        check=1;
        set_servo_position(2,350);
        msleep(1500);
    }
    set_servo_position(3,700);
}
```



```
//Follow the line until a button is pressed
void followTheLineButton(int m1, int m2){
    while(digital(1) == 0){
        if(analog(5) > 4000){
            mav(m1,1500);
            mav(m2,0);
        }
        if(analog(5) < 4000){
            mav(m1,0);
            mav(m2,1500);
        }
    }
    ao();
}
```

Wombat code to follow the black tape line until a button is activated (above)

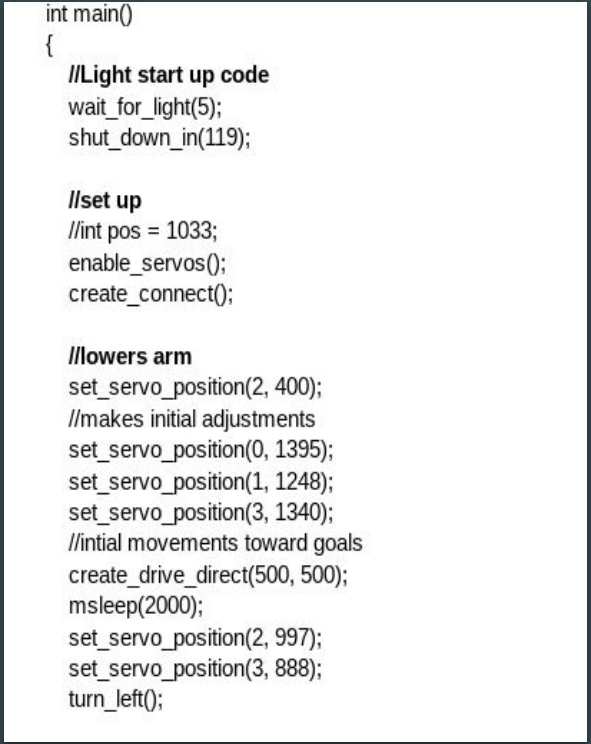
Wombat code to get the rings off the Rock-A-Stack (right)



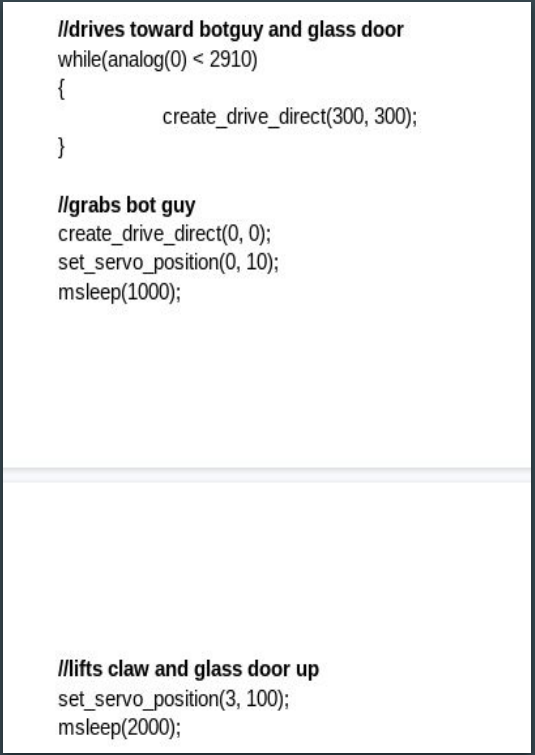
```
//Gets rings off the Rock-A-Stack
void knockOver(){
    set_servo_position(3,1960);
    set_servo_position(1,800);
    msleep(500);
    mav(0,1500);
    msleep(1500);
    ao();
    mav(0,-1400);
    mav(3,-1400);
    msleep(700);
    ao();
    msleep(500);
    set_servo_position(3,900);
    mav(0,1000);
    msleep(500);
    ao();
    set_servo_position(3,1850);
    msleep(500);
    mav(0,1500);
    mav(3,1500);
    msleep(1650);
    ao();
    set_servo_position(1,1500);
    msleep(1000);
    mav(0,-1500);
    mav(3,-1500);
    msleep(500);
    ao();
    set_servo_position(3,900);
    msleep(500);
}
```

Create Bot Code

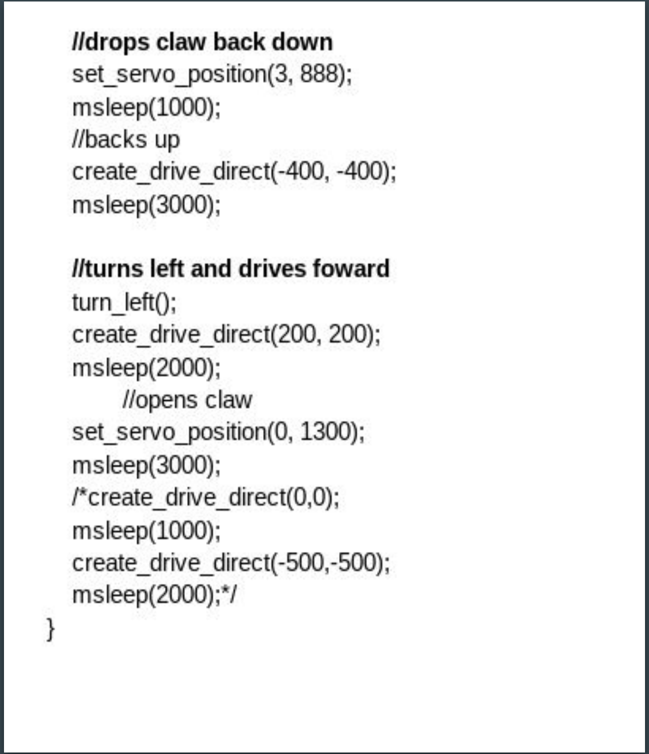
*Purpose of code partitions explained via comments in screenshots



```
int main()
{
    //Light start up code
    wait_for_light(5);
    shut_down_in(119);

    //set up
    //int pos = 1033;
    enable_servos();
    create_connect();

    //lowers arm
    set_servo_position(2, 400);
    //makes initial adjustments
    set_servo_position(0, 1395);
    set_servo_position(1, 1248);
    set_servo_position(3, 1340);
    //intial movements toward goals
    create_drive_direct(500, 500);
    msleep(2000);
    set_servo_position(2, 997);
    set_servo_position(3, 888);
    turn_left();
```



```
//drives toward botguy and glass door
while(analog(0) < 2910)
{
                create_drive_direct(300, 300);
}

//grabs bot guy
create_drive_direct(0, 0);
set_servo_position(0, 10);
msleep(1000);


//lifts claw and glass door up
set_servo_position(3, 100);
msleep(2000);
```



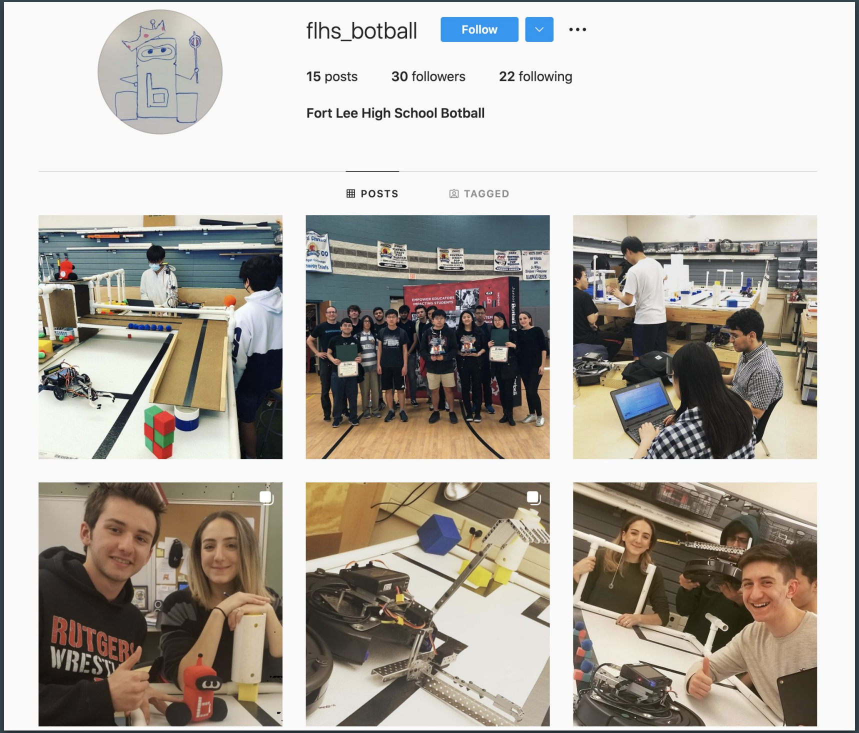
```
    //drops claw back down
    set_servo_position(3, 888);
    msleep(1000);
    //backs up
    create_drive_direct(-400, -400);
    msleep(3000);

    //turns left and drives foward
    turn_left();
    create_drive_direct(200, 200);
    msleep(2000);
        //opens claw
    set_servo_position(0, 1300);
    msleep(3000);
    /*create_drive_direct(0,0);
    msleep(1000);
    create_drive_direct(-500,-500);
    msleep(2000);*/
}
```

Greatest Risk: Inconsistency

As a result of the numerous inconsistent results produced as a result of coding the actual robots, the greatest identified risk would be the concept of inconsistency.

- Various unwanted/inaccurate coding results were produced as a result of volatile factors such as battery life, power/efficiency of servos, accuracy of sensors, and general environment conditions.
- To combat this, we changed our code to adapt in the case of unexpected events, for example using the IR sensor to track black lines.

Social Media Influence



flhs_botball Follow

15 posts 30 followers 22 following

Fort Lee High School Botball

POSTS TAGGED

Thank you!