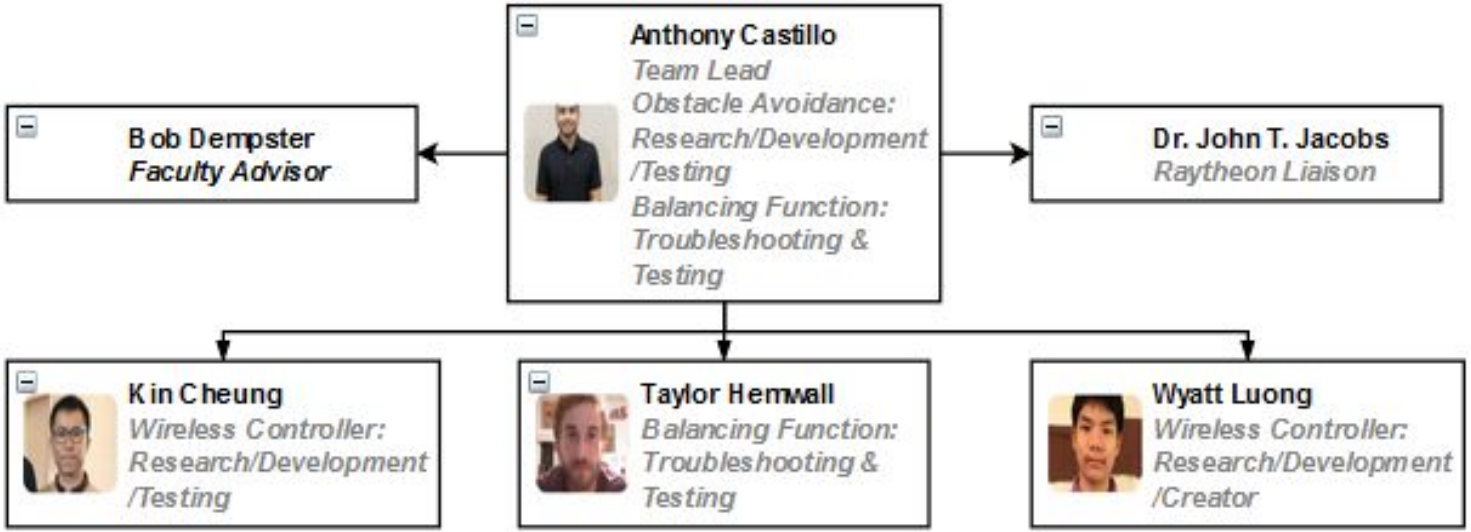
2-Wheel Obstacle Avoidance Robot (30)

Raytheon Liaison: Dr. John T. Jacobs
Faculty Advisor: Bob Dempster
Group Members: Anthony Castillo, Taylor Hemwall, Wyatt Luong, Kin Cheung
University: California State University, Los Angeles
Department: Electrical Engineering
Sponsor: Raytheon Technologies
Date: May 7, 2021

Work Breakdown Structure

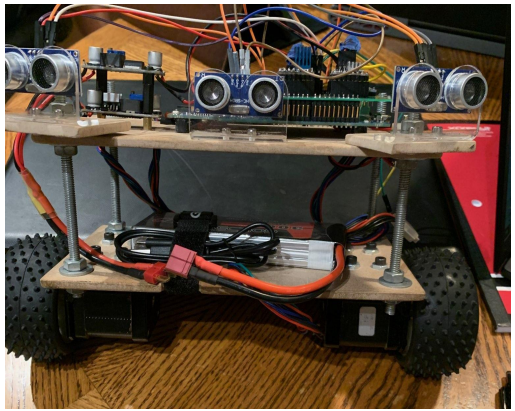


Agenda: 2-Wheel Obstacle Avoidance Robot

1. Project Introduction - **Anthony Castillo**
    a. Introduction
    b. Scope of Project/Requirements
2. Project Overview - **Kin Cheung**
    a. Current Design
    b. System Architecture
    c. Software/Electrical Overview
    d. Battery Trade Study
3. Balancing Function - **Taylor Hemwall**
    a. Balancing Overview
    b. Pendulum Dynamics
    c. PID Control System
4. Obstacle Avoidance Function - **Anthony Castillo**
    a. Obstacle Detection
    b. Obstacle Avoidance
    c. Flowchart
    d. Test Results
5. Wireless Control Function - **Wyatt Luong**
    a. Wireless Controller Overview
    b. Joystick Function
6. Conclusion - **Wyatt Luong**
    a. Future Implementation & Testing
    b. Accomplishments & Conclusion

Project Introduction

- This is an ongoing project
  - Objective 1: Self Balance Function
    - Method: Using an IMU (Inertial Measurement Unit) & PID (Proportional, Integral & Derivative) controller to control the inherently unstable system.
  - Objective 2: Obstacle Avoidance Function
    - Method: Using ultrasonic sensors to detect objects within 1 ft.
  - Objective 3: Wireless Control
    - Method: Using an RF transceiver to give wireless commands to the robot from a joystick.



Scope of Project & Requirements

Scope:

The objective of the 2-Wheel Obstacle Avoidance Robot team is to have the robot balance on two wheels while being controlled wirelessly and avoiding objects autonomously.

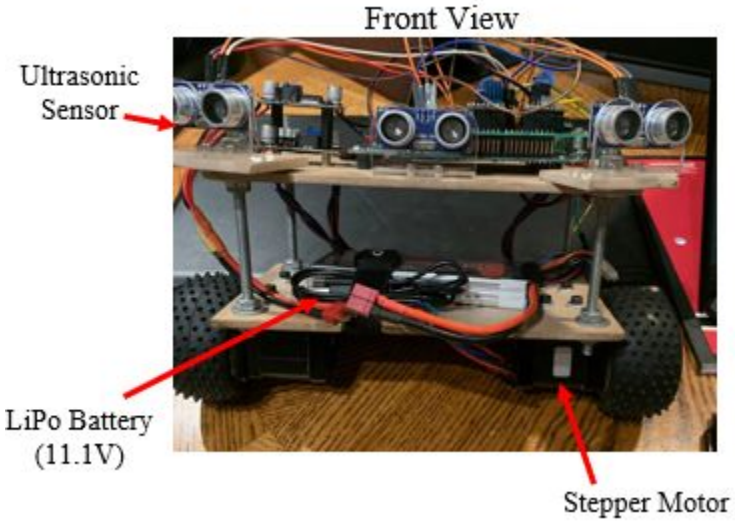
Performance Requirements:

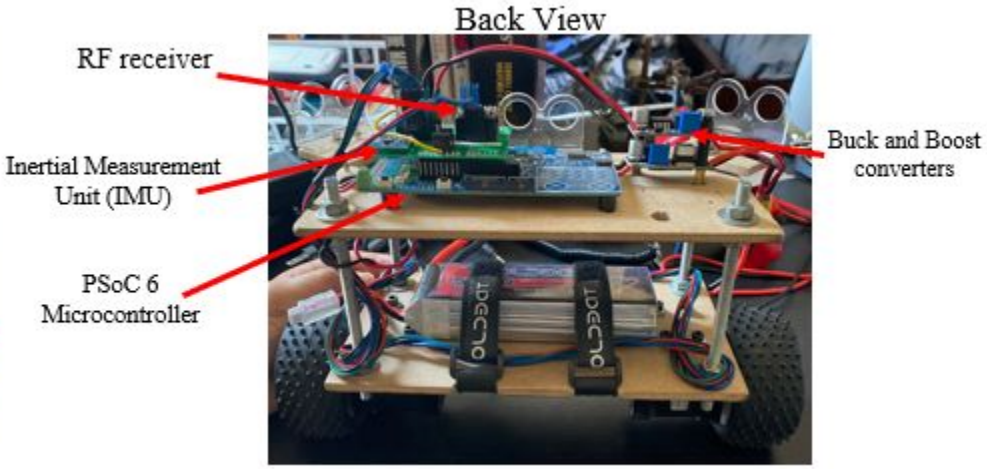
1. The robot shall be capable of balancing continuously for 15 minutes.
2. The robot shall accept and execute movement commands from a wireless controller.
3. The robot shall autonomously avoid obstacles within 1 foot of the robot.

Agenda: 2-Wheel Obstacle Avoidance Robot

1. Project Introduction - **Anthony Castillo**
    a. Introduction
    b. Scope of Project/Requirements
2. Project Overview - **Kin Cheung**
    a. Current Design
    b. System Architecture
    c. Software/Electrical Overview
    d. Battery Trade Study
3. Balancing Function - **Taylor Hemwall**
    a. Balancing Overview
    b. Pendulum Dynamics
    c. PID Control System
4. Obstacle Avoidance Function - **Anthony Castillo**
    a. Obstacle Detection
    b. Obstacle Avoidance
    c. Flowchart
    d. Test Results
5. Wireless Control Function - **Wyatt Luong**
    a. Wireless Controller Overview
    b. Joystick Function
6. Conclusion - **Wyatt Luong**
    a. Future Implementation & Testing
    b. Accomplishments & Conclusion

Current Design

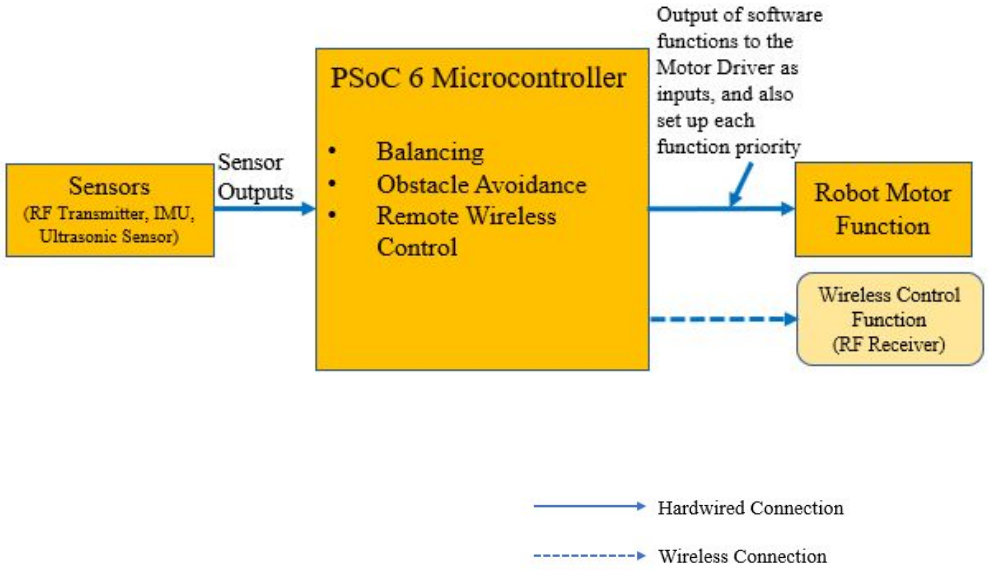




- Top platform: five ultrasonic sensors (only three are shown), PSoC 6, buck and boost converters, IMU, RF receiver, PCB, and motor drivers
- Bottom platform: battery, stepper motors, wheels

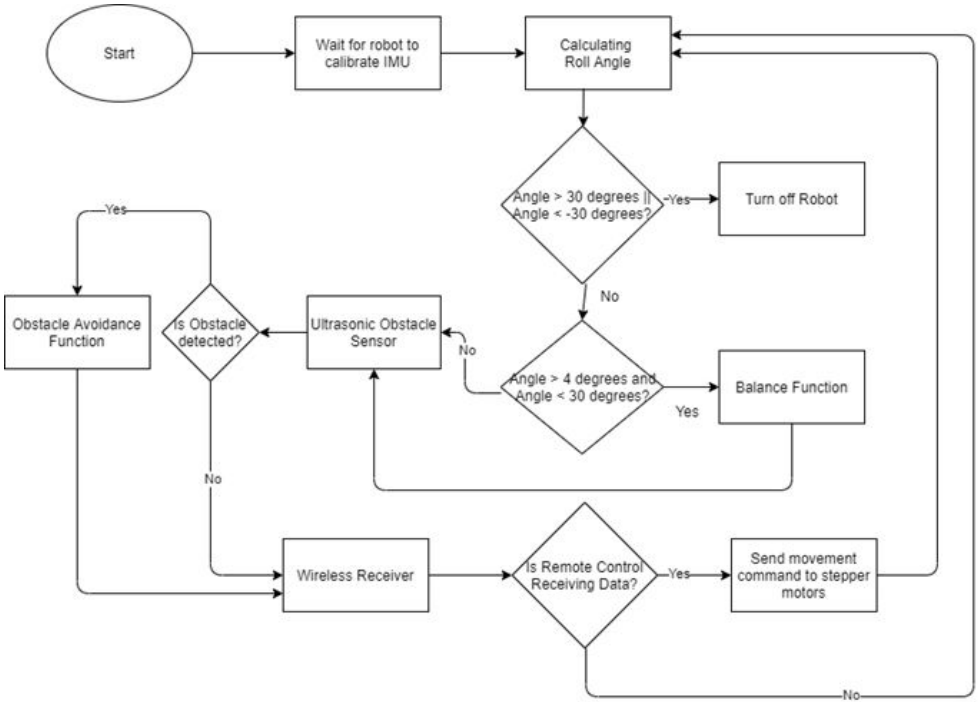
System Architecture

● The 2 wheeled balancing robot needs to handle multiple functions
They are: Balancing, Wireless Control, and Obstacle Avoidance
● Balancing function has highest priority
● Obstacle avoidance is secondary priority and wireless control will be third priority
● The ultrasonic sensors and RF module provide inputs for the PSoC 6 Microcontroller functions
● Sensors and RF module on the robot provide inputs to the PSoC 6 microcontroller that allow different functions to be executed to achieve Balancing, Obstacle avoidance, Controlled Motion functions to be executed
● Finally, the PSoC 6 microcontroller provides commands to the motor drivers



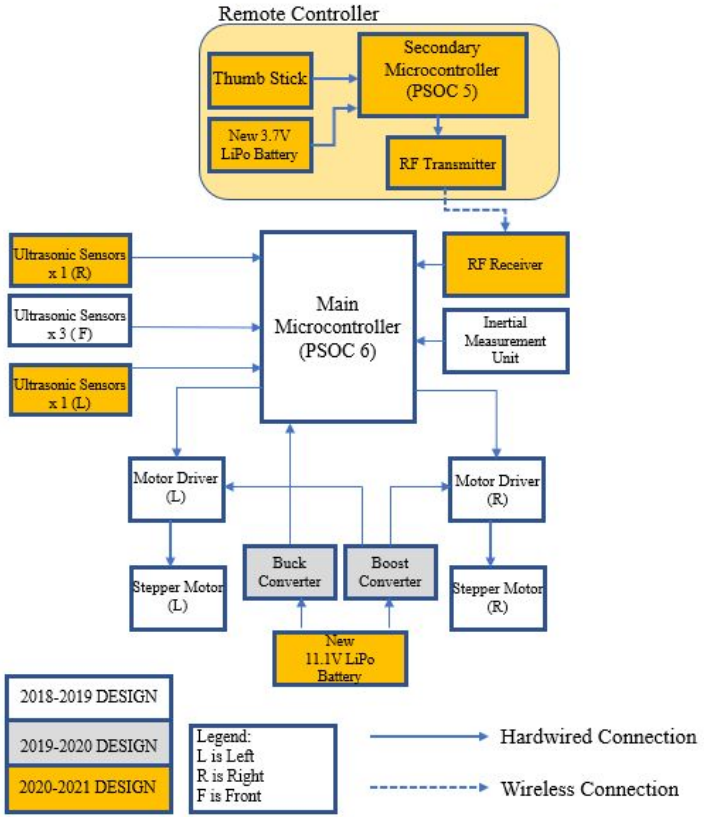
Software

- In order for the robot to perform each of the three functions,
we want to use interrupts to execute 3 tasks in order of priorities to ensure a functional transition between tasks
- 1st priority is balancing function
- 2nd priority is obstacle avoidance
- 3rd priority is wireless control



Electrical Block Diagram

- PSoC 6 is the main microcontroller
- Robot uses five ultrasonic sensors to detect obstacles, three sensors on the front, and one sensor on each side
- PSoC 5 will be used as a means to control the joystick and communicate the wireless controller with the PSoC 6 microcontroller
- Inertial Measurement Unit (IMU) will be used for balancing
- Installed new 11.1V LiPo battery that is connected to Buck and Boost Converters, Buck Converter reduces voltage for PSoC 6, and Boost Converter increases voltage to 12V for the motor drivers.
- Battery will provide enough power headroom to ensure the robot can balance for 15 minutes and provide sufficient voltage to for different components
- PSoC 6 gives commands to the motor drivers



Battery Trade Study

Battery Name	Ovonic 50C 3S 5200 mah 11.1V with EC3 Connector	GOLDBAT 5200mAh 3S 11.1V 60C	HOOVO 11.1V 80C 5200 mAh	Protek RC 3S 120C Low IR SI-Graphene with 11.4V and 4500mAh	Gens ACE Adventure 3600 mAh 3Slp 11.4V 50C battery with XT60 Plug	Turnigy 4000 mAh 3S 30C	POVWAY 3S 50C LiPo battery 11.1V 4500mAh	Venom LiPo Battery 11.1V 4000mAh
Weights of battery	298 grams	350 grams	380 grams	236 grams	215 grams	347 grams	314 grams	253 grams
Dimensions (Length x Width x Height)	5.45 in x 1.84 in x 1.53 in	5.35 in x 1.65 in x 1.18 in	5.47 in x 1.85 in x 1.46 in	3.66 in x 1.69 in x 1.01 in	3.54 in x 1.65 in x 0.98 in	5.64 in x 1.96 in x 0.86 in	5.31 in x 1.57 in x 0.98 in	5.39 in x 1.77 in x 0.94 in
%headroom	42.31%	42.31%	42.31%	35%	19%	25%	33.33%	25%
Costs	$37.99	$25.99	$32.99	$59.99	$43.99	$29.68	$27.99	$48.99
Weight of robot(new battery included)	2065 grams	2117 grams	2147 grams	2003 grams	1982 grams	2114 grams	2081 grams	2020 grams

11

Final choice: GOLDBAT 5200 mAh 11.1V battery

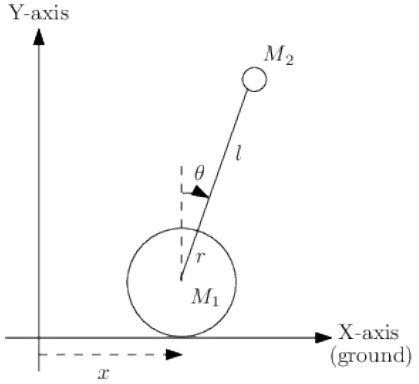
Reasons for choosing: provides the most headroom for power at the lowest cost without adding too much additional weight to the robot

Agenda: 2-Wheel Obstacle Avoidance Robot

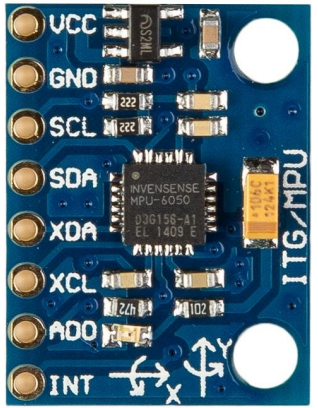
1. Project Introduction - **Anthony Castillo**
    a. Introduction
    b. Scope of Project/Requirements
2. Project Overview - **Kin Cheung**
    a. Current Design
    b. System Architecture
    c. Software/Electrical Overview
    d. Battery Trade Study
3. Balancing Function - **Taylor Hemwall**
    a. Balancing Overview
    b. Pendulum Dynamics
    c. PID Control System
4. Obstacle Avoidance Function - **Anthony Castillo**
    a. Obstacle Detection
    b. Obstacle Avoidance
    c. Flowchart
    d. Test Results
5. Wireless Control Function - **Wyatt Luong**
    a. Wireless Controller Overview
    b. Joystick Function
6. Conclusion - **Wyatt Luong**
    a. Future Implementation & Testing
    b. Accomplishments & Conclusion

Balancing Overview

- Robot is unstable in the theta (θ) direction, must have a control system to stop it from falling over
- The Inertial Measurement Unit (IMU) onboard the robot takes measurements needed by the control system
- The IMU is a combination of a gyroscope (measures the angle of the robot relative to gravity) and an accelerometer (measures angular acceleration)
- These measurements are taken in the roll (θ), pitch & yaw directions and passed along to the PSoC6 Microcontroller
- Software control system tells the robot which direction to move (via roll angle) and how quickly to move (via roll acceleration)



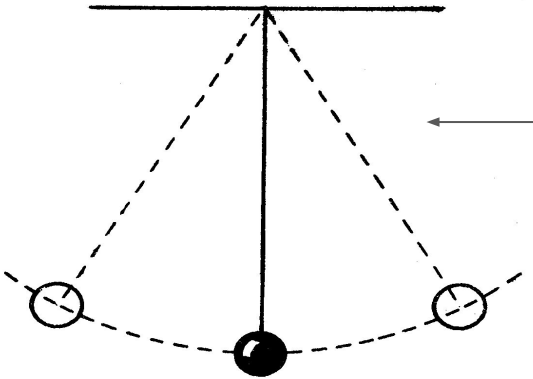
Balancing Robot Diagram



IMU (MPU6050)

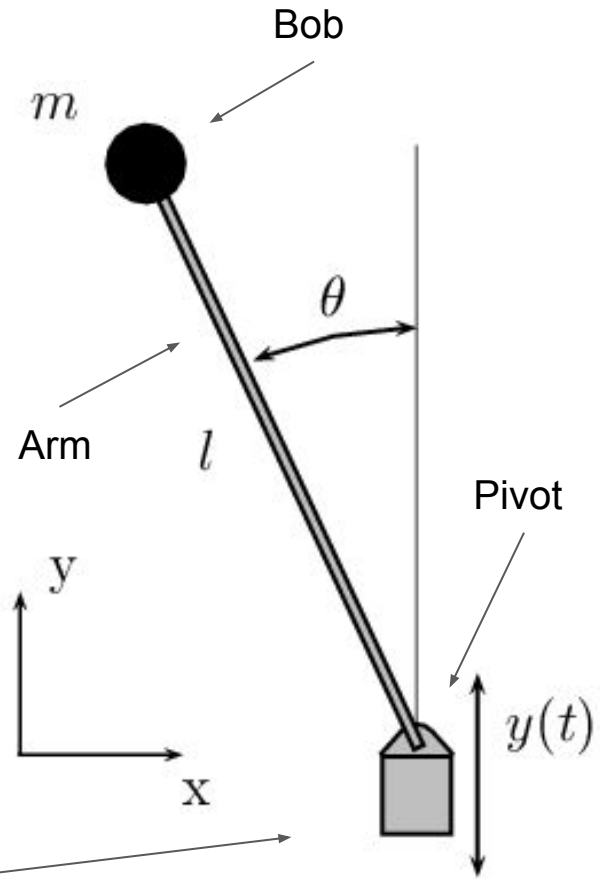
Pendulum Dynamics

- Regular pendulum: inherently stable
- Inverted pendulum: inherently unstable
- Needs control system
- Pivot must be moved below bob
- Our balancing robot moves pivot in direction of fall to balance (changes center of gravity)



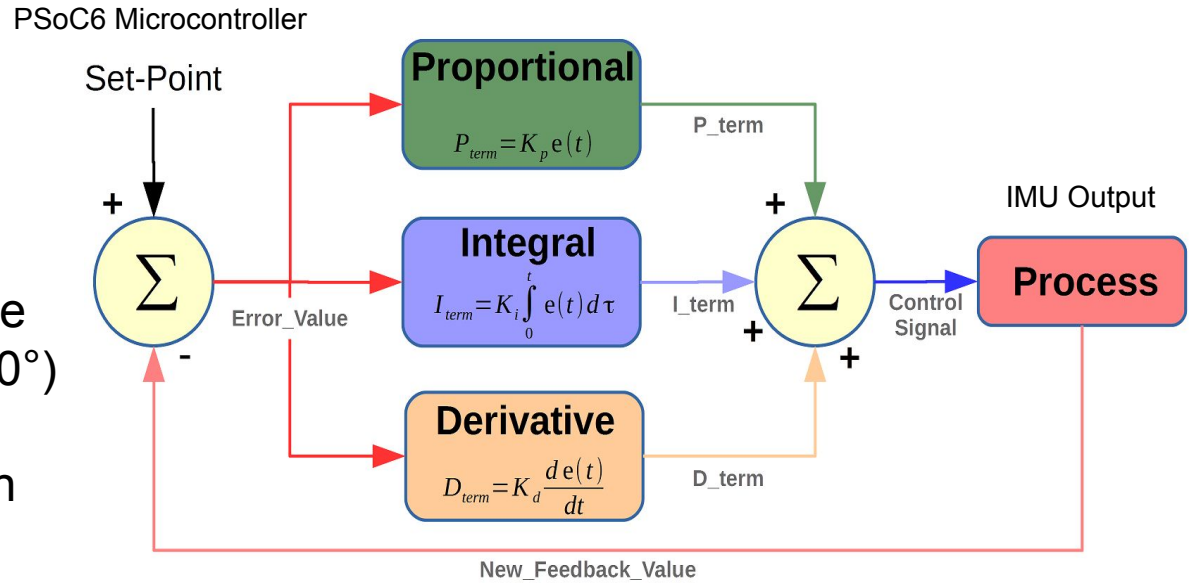
Stable under gravity

Unstable under gravity



PID Controller

- Proportional, integral, derivative controller
- Uses feedback loop to generate error signal
- Error signal is the difference between the set point (θ = 0°) and current output
- Error signal is input through PID controller and a new control signal is generated



- Proportional term Kp: based on current error
- Integral term Ki: based on the accumulation of previous errors
- Derivative term Kd: based on current rate of change (anticipates future errors)
- Controller signal: summation of Kp, Ki, and Kd terms

PID Values Generated by MATLAB Script

- PID values used by first year's team (random guessing):
  - Kp = 30, Ki = 0.1, Kd = 16
- PID values used by previous year's team (via observation and trial & error):
  - Kp = 36, Ki = 1, Kd = 20
- PID values generated by our MATLAB script:
  - Kp = 21.6, Ki = 3.6, Kd = 3.78
- MATLAB script uses top & bottom platform weights of our robot to calculate PID values
- The complex frequency variable 's' represents $\sigma+j\omega$ (real & imaginary numbers)
- The expression on the right shows how PID values are summed and generate a control signal

$$Kp + Ki * \frac{1}{s} + Kd * s$$

with Kp = 21.6, Ki = 3.6, Kd = 3.78

Continuous-time PID controller in parallel form.

Balancing Function Software Programming Steps

1. Initializes the IMU
2. User positions the robot upright for the calibration process
3. Performs the calibration process by taking 2,000 measurements of the gyroscope & accelerometer roll, pitch & yaw values and averaging them
4. Uses these measurements to generate offsets to account for IMU physical alignment
5. Once the calibration process is over, the robot balances on its own using the PID controller to generate movement commands for the motors

Agenda: 2-Wheel Obstacle Avoidance Robot

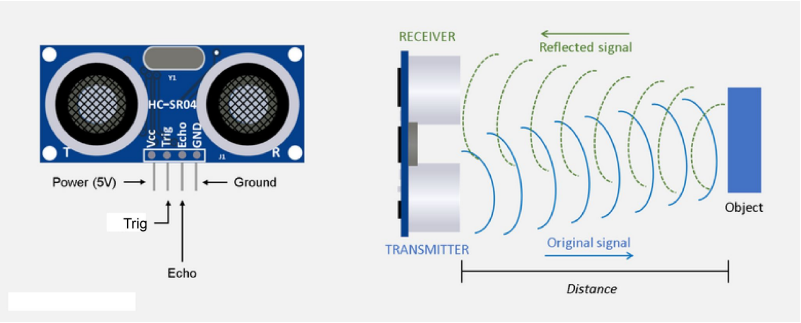
1. Project Introduction - **Anthony Castillo**
    a. Introduction
    b. Scope of Project/Requirements
2. Project Overview - **Kin Cheung**
    a. Current Design
    b. System Architecture
    c. Software/Electrical Overview
    d. Battery Trade Study
3. Balancing Function - **Taylor Hemwall**
    a. Balancing Overview
    b. Pendulum Dynamics
    c. PID Control System
4. Obstacle Avoidance Function - **Anthony Castillo**
    a. Obstacle Detection
    b. Obstacle Avoidance
    c. Flowchart
    d. Test Results
5. Wireless Control Function - **Wyatt Luong**
    a. Wireless Controller Overview
    b. Joystick Function
6. Conclusion - **Wyatt Luong**
    a. Future Implementation & Testing
    b. Accomplishments & Conclusion

Obstacle Avoidance Function Overview

- Two functions must work before implementing the obstacle avoidance function into the robot:
  - The ability to detect obstacles (**Obstacle Detection**):
    - The objective of this task is to detect objects within 1 ft in front or on the right or left side of the robot.
  - The ability to avoid obstacles (**Obstacle Avoidance**):
    - The objective for this task is to avoid detected obstacles in front and on the right or left side of the robot.

Obstacle Detection

- Method: Ultrasonic sensors
- Ultrasonic sensors are programmed to send out a small voltage to the “TRIG” (Trigger) pin that will then trigger the sensor to send out a 10us, 8-cycle sonar pulse signal that is then reflected back from the object in front of it. Once the signal returns to the sensor, it is flagged with the received “ECHO” signal. The time between TRIG and ECHO signals (***time***) can then be calculated and be used to find the distance of the object as shown in the equations below.



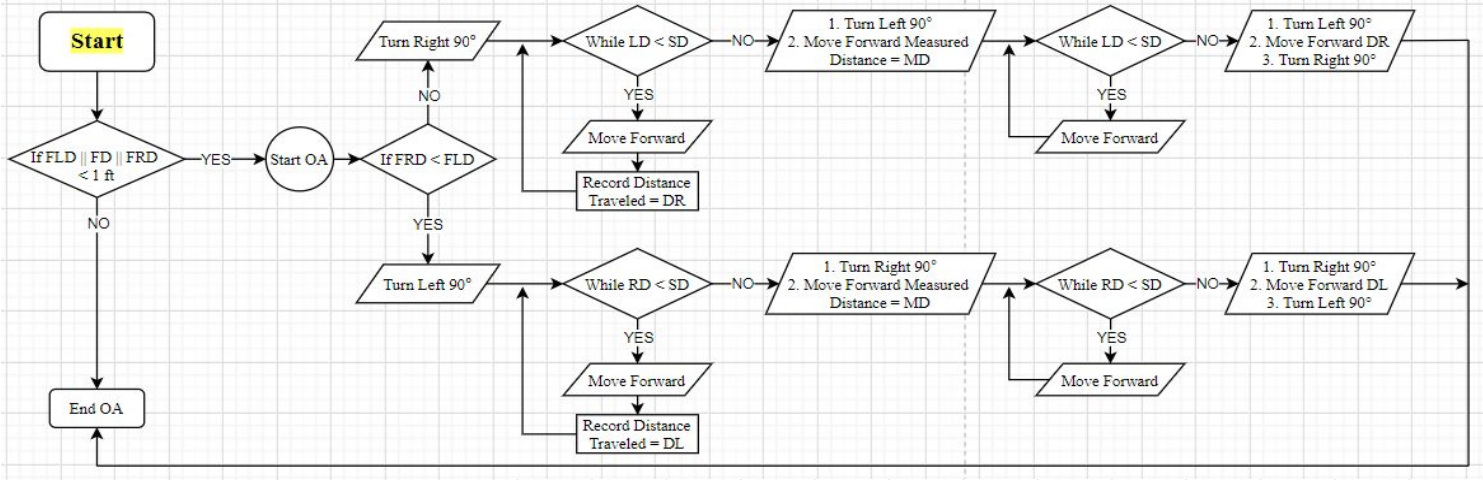
$$Distance_{counts} = \frac{time \times (speed\ of\ sound)}{2}$$

$$Distance_{inches} = \frac{Distance_{counts}}{148}$$

Obstacle Avoidance

- Method: The robot will default to turning right when an obstacle is detected straight ahead or to its left side. It will then continue to move forward while checking for obstacles to the left or right side ultrasonic sensors. Finally, it will turn back to the starting direction when the obstacle is cleared.
- The opposite commands will execute if an object is detected on its right side ultrasonic sensor.
- During the obstacle avoidance loop, the robot will not process any commands from the remote controller until obstacle avoidance is complete however, the robot will give control back to the balancing function after each turn is completed.

Obstacle Avoidance Flowchart



- Robot checks if an object is in front of it.
- If object is detected, obstacle avoidance begins automatically
- Robot turns left/right based on the shortest path
- Robot maneuvers around obstacle
- Robot returns control to the user

Legend:

LD = Left Distance
FLD = Front Left Distance
FD = Front Distance
FRD = Front Right Distance
RD = Right Distance

OA = Obstacle Avoidance
MD = Measured Distance
SD = Side Distance
DL = Distance to the Left
DR = Distance to the Right

MD: Depending on the method of turning (moving left wheel only, moving right wheel only, moving both wheels) move forward a measured distance until the obstacle being avoided is in view of the left or right sensor (whichever is facing the obstacle).

DL & DR: Keep track of left/right distance traveled so that the robot will travel back the same distance to its original vertical (foreward-facing) position.

SD: Depending on the method of turning (moving left wheel only, moving right wheel only, moving both wheels) select a reasonable distance value that the side sensor will be within.

Ultrasonic Sensor Accuracy Test Results

Sensor 1			
@ 0.5 ft (ft)	@ 1 ft (ft)	@ 1.5 ft (ft)	@ 2 ft (ft)
0.62	1.16	1.74	2.31
0.61	1.16	1.75	2.32
0.62	1.16	1.73	2.33
0.62	1.16	1.74	2.31
0.62	1.16	1.75	2.31
0.61	1.16	1.75	2.31

Sensor 2			
@ 0.5 ft (ft)	@ 1 ft (ft)	@ 1.5 ft (ft)	@ 2 ft (ft)
0.6	1.17	1.74	2.32
0.6	1.16	1.74	2.32
0.58	1.16	1.74	2.32
0.59	1.17	1.74	2.32
0.59	1.17	1.74	2.32
0.59	1.16	1.74	2.32

Sensor 3			
@ 0.5 ft (ft)	@ 1 ft (ft)	@ 1.5 ft (ft)	@ 2 ft (ft)
0.59	1.15	1.75	2.33
0.59	1.15	1.75	2.33
0.59	1.15	1.74	2.33
0.59	1.15	1.75	2.33
0.59	1.15	1.73	2.34
0.59	1.15	1.74	2.33

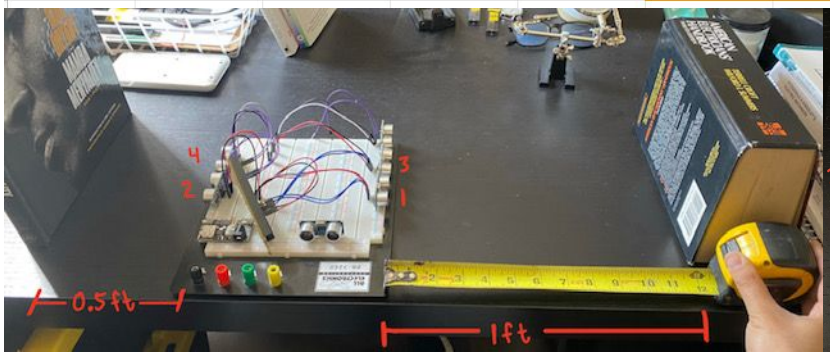
Sensor 4			
@ 0.5 ft (ft)	@ 1 ft (ft)	@ 1.5 ft (ft)	@ 2 ft (ft)
0.62	1.14	1.73	2.3
0.62	1.14	1.73	2.3
0.62	1.14	1.73	2.3
0.64	1.14	1.73	2.31
0.62	1.14	1.73	2.3
0.62	1.14	1.73	2.3

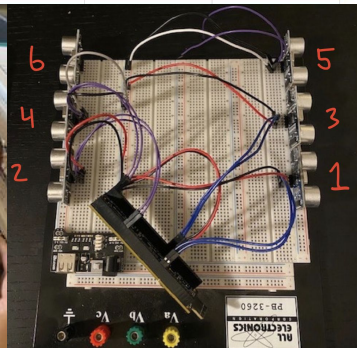
Sensor 5			
@ 0.5 ft (ft)	@ 1 ft (ft)	@ 1.5 ft (ft)	@ 2 ft (ft)
0.61	1.18	1.74	2.31
0.61	1.18	1.74	2.31
0.61	1.18	1.74	2.31
0.61	1.19	1.74	2.33
0.61	1.18	1.74	2.32
0.61	1.18	1.74	2.33

Sensor 6			
@ 0.5 ft (ft)	@ 1 ft (ft)	@ 1.5 ft (ft)	@ 2 ft (ft)
0.65	1.2	1.76	2.36
0.66	1.19	1.76	2.36
0.65	1.2	1.76	2.36
0.65	1.2	1.76	2.36
0.65	1.2	1.76	2.36
0.65	1.2	1.76	2.36

Actual Obstacle Distance			
@ 0.5 ft (ft)	@ 1 ft (ft)	@ 1.5 ft (ft)	@ 2 ft (ft)
0.5	1	1.5	2

All measurements have an error of +- 10%



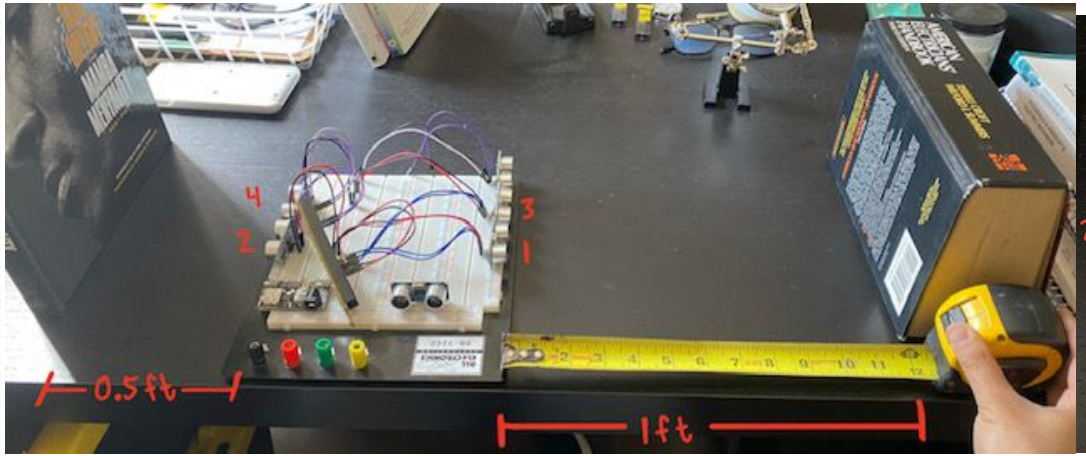


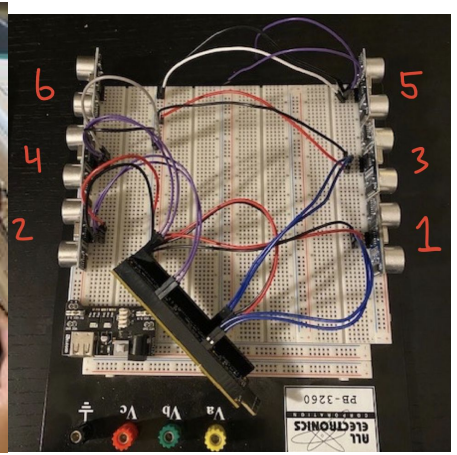
- Green: error <= 0.18 ft
- Yellow: 0.18 <= error <= 0.25 ft
- Red: error > 0.25 ft

Ultrasonic Sensor Synchronization Test Results

```
Distance 1 = 1.20 ft
Distance 2 = 0.59 ft
Distance 3 = 1.18 ft
Distance 4 = 0.57 ft
Distance 1 = 1.20 ft
Distance 2 = 0.57 ft
Distance 3 = 1.16 ft
Distance 4 = 0.58 ft
Distance 1 = 1.20 ft
Distance 2 = 0.59 ft
Distance 3 = 1.17 ft
Distance 4 = 0.58 ft
Distance 1 = 1.18 ft
Distance 2 = 0.59 ft
Distance 3 = 1.16 ft
Distance 4 = 0.59 ft
Distance 1 = 1.19 ft
Distance 2 = 0.59 ft
Distance 3 = 1.16 ft
Distance 4 = 0.58 ft
Distance 1 = 1.18 ft
Distance 2 = 0.59 ft
Distance 3 = 1.17 ft
Distance 4 = 0.57 ft
```

- We were able to synchronize up to 4 sensors at once.
  - Method: Create an infinite loop and trigger each sensor consecutively with a 30 millisecond delay in between sensors to give the software program time to complete its function.
    - Optimal readings for sensors 1 & 3: 1 ft
    - Optimal readings for sensors 2 & 4: 0.5 ft



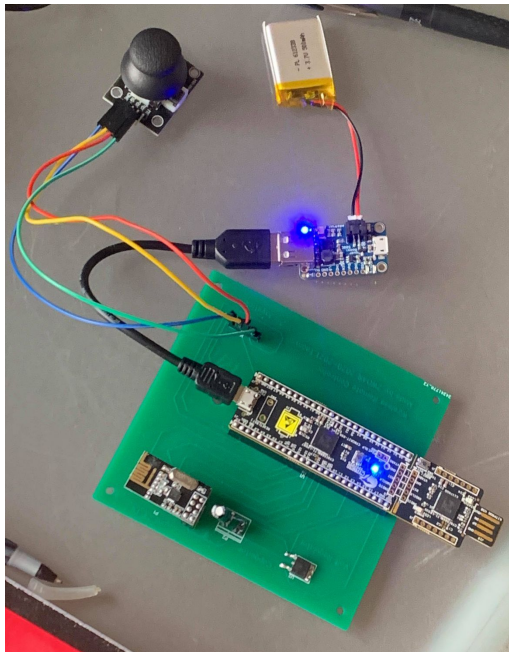


Agenda: 2-Wheel Obstacle Avoidance Robot

1. Project Introduction - **Anthony Castillo**
    a. Introduction
    b. Scope of Project/Requirements
2. Project Overview - **Kin Cheung**
    a. Current Design
    b. System Architecture
    c. Software/Electrical Overview
    d. Battery Trade Study
3. Balancing Function - **Taylor Hemwall**
    a. Balancing Overview
    b. Pendulum Dynamics
    c. PID Control System
4. Obstacle Avoidance Function - **Anthony Castillo**
    a. Obstacle Detection
    b. Obstacle Avoidance
    c. Flowchart
    d. Test Results
5. Wireless Control Function - **Wyatt Luong**
    a. Wireless Controller Overview
    b. Joystick Function
6. Conclusion - **Wyatt Luong**
    a. Future Implementation & Testing
    b. Accomplishments & Conclusion

Wireless Remote Controller Overview

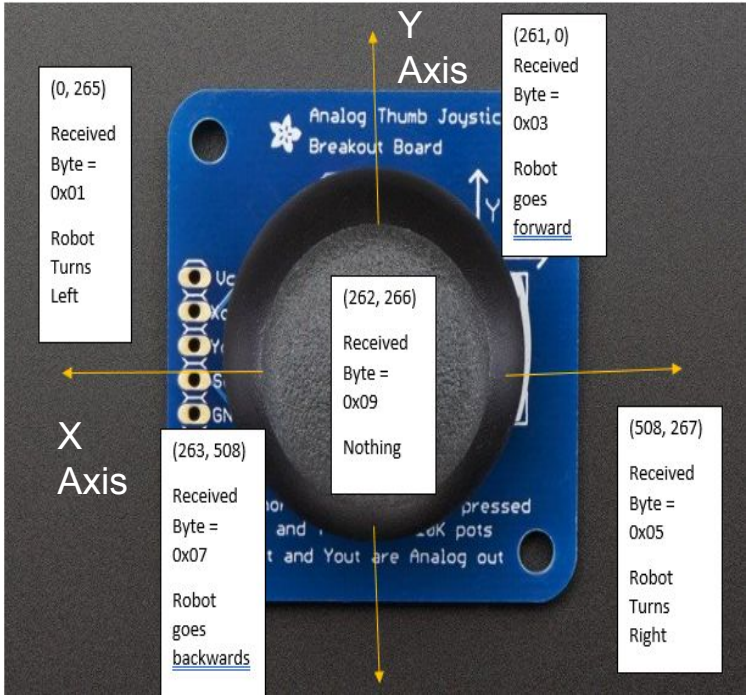
- One of the requirements we want the robot to achieve is to receive a control signal from the remote controller which controls the robot's movement
- The transmitter(controller) and receiver(robot) use Serial Peripheral Interface(SPI) to send and receive data
- The controller consists of a PSoC 5LP board, an nRF24 transmitter module, a 3.3V buck converter chip, a 3.7V lipo battery, adafruit boost converter, and a thumbstick
- Wireless function has been confirmed to be working properly through numerous experiments.
- Experiments consist of range testing, data consistency testing, and voltage/current measurements.
    - The power drawn from the battery indicates an operation time of at least an hour which exceeds the 15 minutes requirement.
    - The transmitter can send consistent data to the receiver within a range of 50 feet.



Complete Wireless Controller Design

Joystick Function

- We expect the robot to move according to the joystick position.
- Software program in the robot will continuously monitor if user toggles joystick at any moment while the robot is balancing.
- Figure on the right shows how joystick, toggled at certain (x,y) positions as shown, can output 4 possible directions of robot movement.
- The only time a wireless control function is not applicable/active is when obstacle avoidance function is activated.



Agenda: 2-Wheel Obstacle Avoidance Robot

1. Project Introduction - **Anthony Castillo**
    a. Introduction
    b. Scope of Project/Requirements
2. Project Overview - **Kin Cheung**
    a. Current Design
    b. System Architecture
    c. Software/Electrical Overview
    d. Battery Trade Study
3. Balancing Function - **Taylor Hemwall**
    a. Balancing Overview
    b. Pendulum Dynamics
    c. PID Control System
4. Obstacle Avoidance Function - **Anthony Castillo**
    a. Obstacle Detection
    b. Obstacle Avoidance
    c. Flowchart
    d. Test Results
5. Wireless Control Function - **Wyatt Luong**
    a. Wireless Controller Overview
    b. Joystick Function
6. Conclusion - **Wyatt Luong**
    a. Future Implementation & Testing
    b. Accomplishments & Conclusion

Implementation

- The following code has been created and needs implementation:
  - Obstacle Detection
  - Integrate wireless control function onto robot
- The hardware implementation needed:
  - Add two more ultrasonic sensors to the robot
  - Manufacture 3D-printed case for the wireless controller

Testing

- Remaining tests to be completed
  - Obstacle Avoidance function
  - Test Wireless function when integrated on the robot
  - Check voltage ratings to ensure motor drivers + stepper motors work properly
  - Demonstrate dynamic balancing on 2 wheels
- Future Improvements
  - Modify existing software to handle inputs from 2 more ultrasonic sensors
  - Use simulated PID values as a starting point for testing
  - Measure the execution time for each function with a timer and modify the software for robot to prioritize all 3 functions

Accomplishments and Conclusion

- Wireless controller is fully functional as verified by experimental trials
- 4 ultrasonic sensors can accurately detect an object in front of the sensors simultaneously.
- Replaced old LiPo battery with a new one to increase the power headroom.
- Conclusion: We have made progress on the battery choice, obstacle avoidance, IMU signal and wireless controller functions.

Thank you