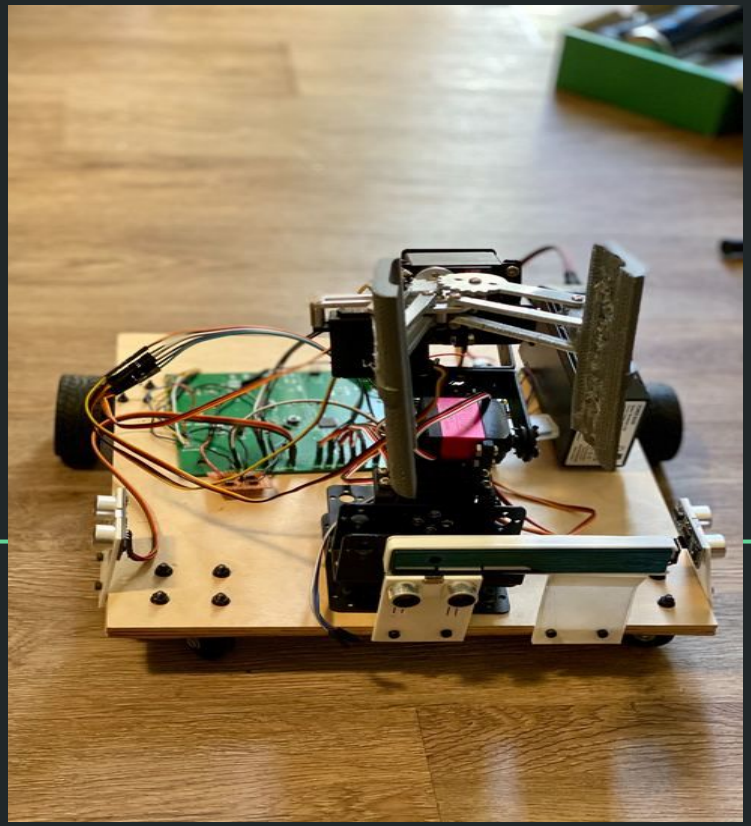
Rolling Arm



Senior Design Project
CECS 490B
CSULB Fall 2019
Bryan Linares, Thomas Nguyen, Kevin & Cristian Lopez

Overview

- Project Goal
- Product Specification and Constraints
- Functional Components Used
- Mechanical Solutions
- Problems Identified and Addressed
- Hardware Selection & Development
- SBC Software Solutions & Summary
- User Interfaces
- Power Requirements and Solutions
- Demo

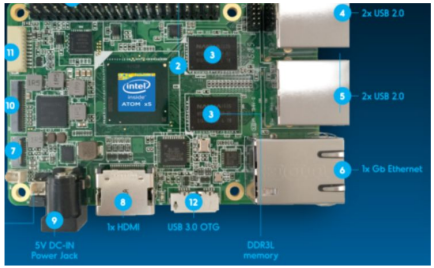
Project Goals Reached

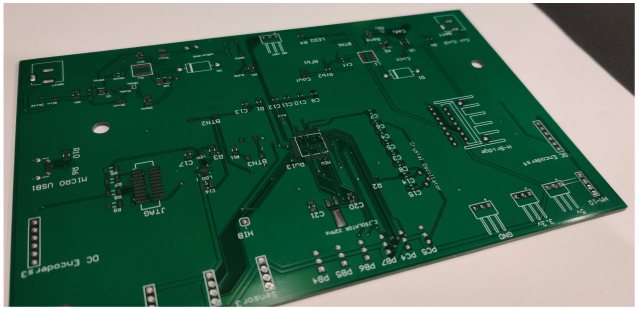
An autonomous can picking robot car using a camera and an arm with a claw to pick up cans and put them in a bin.

A. Autonomous Mobile Robot
B. 6 DOF axis arm programmed to pick up cans
C. Ability to detect cans using computer vision
D. Balanced and stable while picking up cans.
E. 2 hour battery life.
F. Safe to operate
G. Cleaner environment in mind.

Product Specifications and Constraints

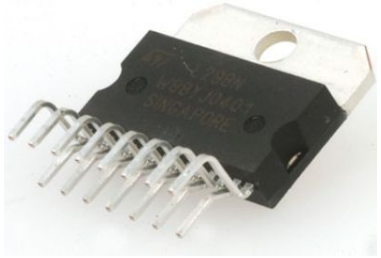
- Single board computer that is capable of running a camera
- Microcontroller with 8 PWM pins and 1 UART input and output.
- Dimensions x-y 40-40 cms and height about 60 cm height
- Minimum of four axes of rotation for the arm. Is able to reach up to 15 centimeters.
- Strong servo motor to withstand weight of the arm.
- Able to grasp Plastic/Aluminum bottles
- Wood for car body to withstand 55g weight of the arm and 20g for the load.
- DC Motors for wheels, Speed controlled by PWM
- Front facing Camera to view objects, specifications give an outside range of 10 meters, inside up to 3.5m.
- PCB for efficiency and reproducibility





Functional Components Used

- Intel Realsense R200(480p30)
- Ultrasonic Sensors
- Intel UP Board
- 6 axis robotic arm
- 6 high torque servos
- 2 DC encoder motors
- TM4C123 Microcontroller
- L298 H-bridge chip
- 2 buck converter modules at 5V
- Lithium Ion 2500mAh 20A Battery



Specific Mechanical Solutions

Aideepen ROT3U 6 axis Arm

Calibrated each individual servo(MG996R 55g).

Servos with Higher torque (ANNIMOS 20KG)

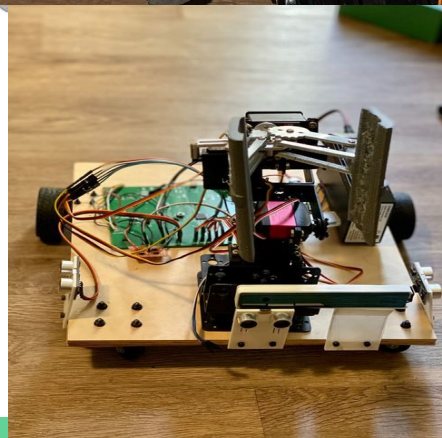
Custom 3D Printed Claw For easier Grabbing

2 Motorized Wheel on base platform

2 55 RPM DC Motor 2 Caster Wheel

Wood Base Car

Custom 3D printed Sides.



Problems Identified and Addressed

Servo Torque limited to 10 kg/cm at 5V.

MCU Software with UltraSonic Sensors. SBC Software compatiblity

Built in buck Power supply, did not work, had to solder one onto the board

JTAG interface circuit

Soldering issues, the pins bridging and board not behaving like we expect

Collisions at corners, added switches

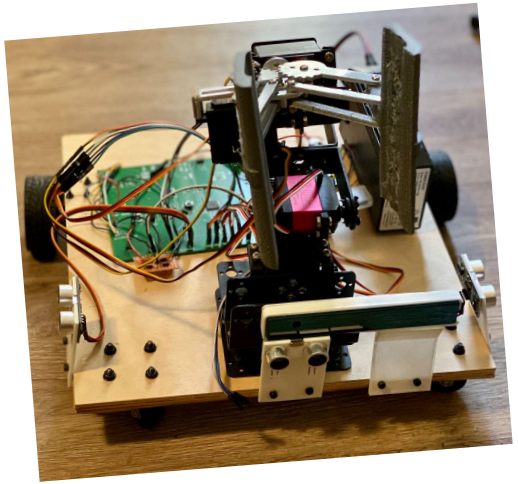
Detecting cans by computer vision

High speed motors would not operate constantly at required speeds, were too fast and weren’t reliable to reach cans

Major Improvement needed: Improve image recognition speed

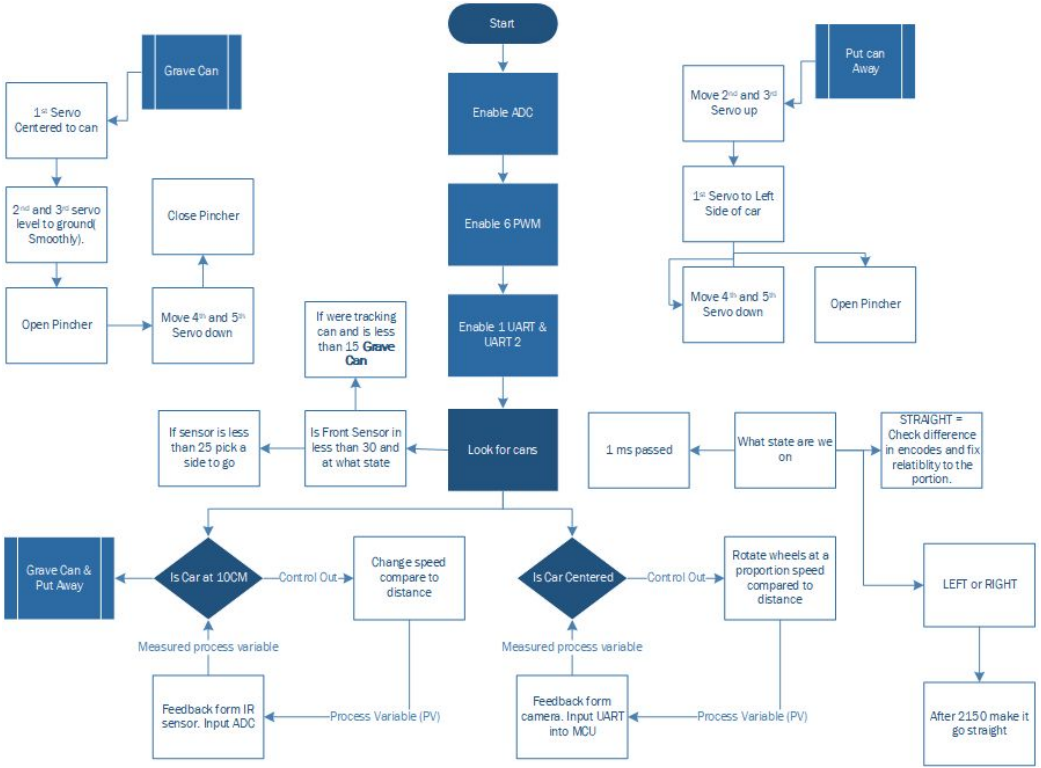
Hardware: Selection and Development

- **Intel UP-Board SBC over other SBC like Raspberry Pi**
  - For full compatibility with intel camera
  - Necessary power to process camera data; Atom x5-z8350 Processor 1.92 GHz CPU, 4GB RAM
  - Onboard memory so less worry about corrupt SD card; 32GB+
  - Shares similar external pin count and layout as Raspberry Pi
  - Professional and Industrial use
- **Aideepen ROT3U arm used**
  - Six possible axis
- **TM4C123G, hardware features used**
  - 48 pins(Motor direction, UltraSonics)
  - 2 UARTS
  - 8 PWM pins



Arm Software Solution

- 6 PWMs are used to controlled the arm
- 6 Functions are created to ease the control of arm
- 2 Functions are created to grab a can and put away a can
- Once we get a UART signal Arm attempts to grab can and puts it away
- Sensors are Used to check when we are in front of the can or when we have approached a wall and send interrupt



UP Board SBC Software Solutions:

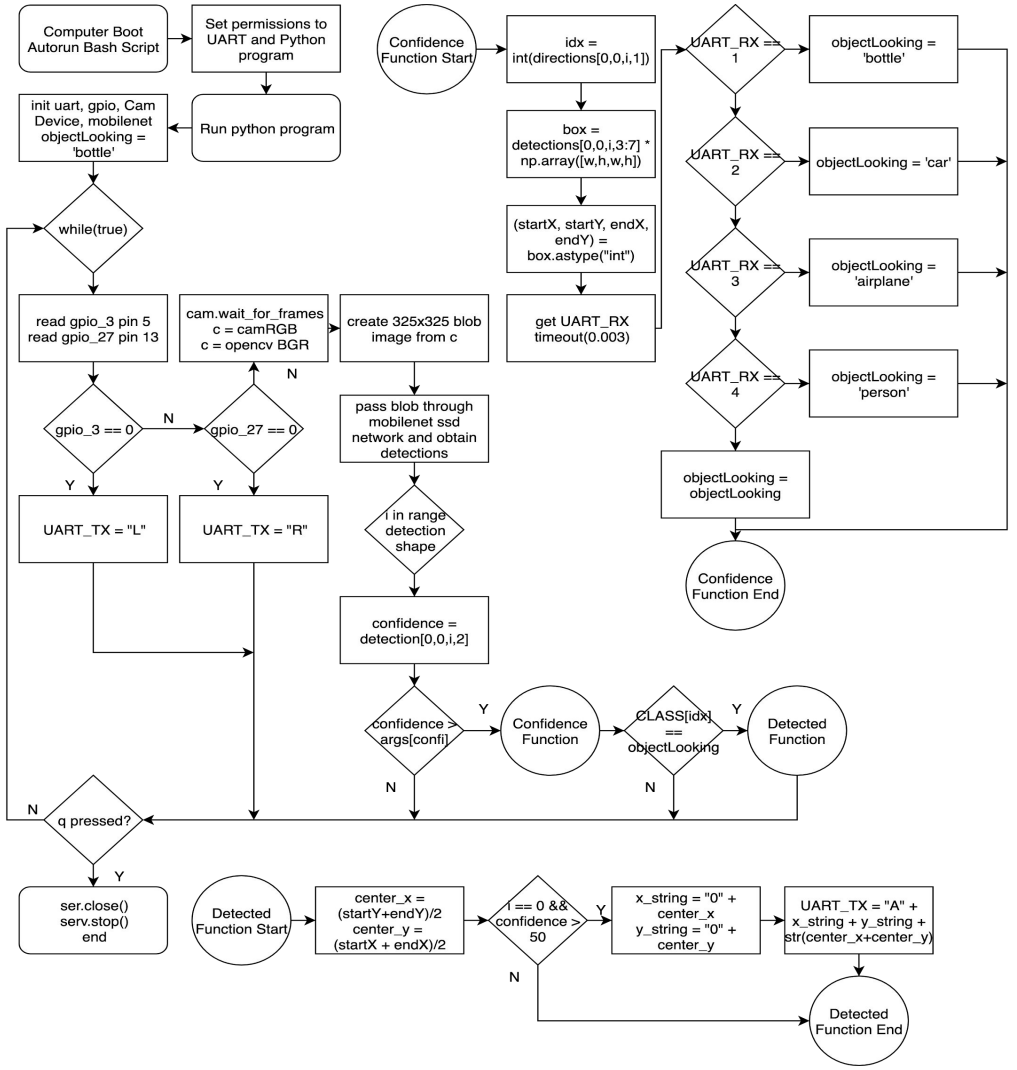
- Intel RealSense, OpenCV, Python 2.7, and Bash Script on Linux 16.04
- Script sets up UART permissions, and begins program on startup
- General detection method is using pretrained caffemodel file, checks for confidence of object
- OpenCV code draws a box around the object and transmits the center location

pythonscript.sh | real_time_object_detection.py

```
#!/bin/bash
sleep 1
echo sending data...
echo q to quit...

echo "asdf123" | sudo -S -k chmod 666 /dev/ttyS4

cd /home/kevin/Desktop/real-time-object-detection/

echo "asdf123" | sudo -S -k python /home/kevin/Desktop/real-time-object-detection/real_time_object_detection.py --prototxt MobileNetSSD_deploy.prototxt.txt --model MobileNetSSD_deploy.caffemodel &

while : ; do

#secho 1 > echo "asdf123" | sudo -S -k /sys/class/leds/upboard\:green\:/brightness

read -n 1 k <&1
if [[ $k = q ]] ; then
pkill python;
fi

done

```

SBC Software Solutions Summary:

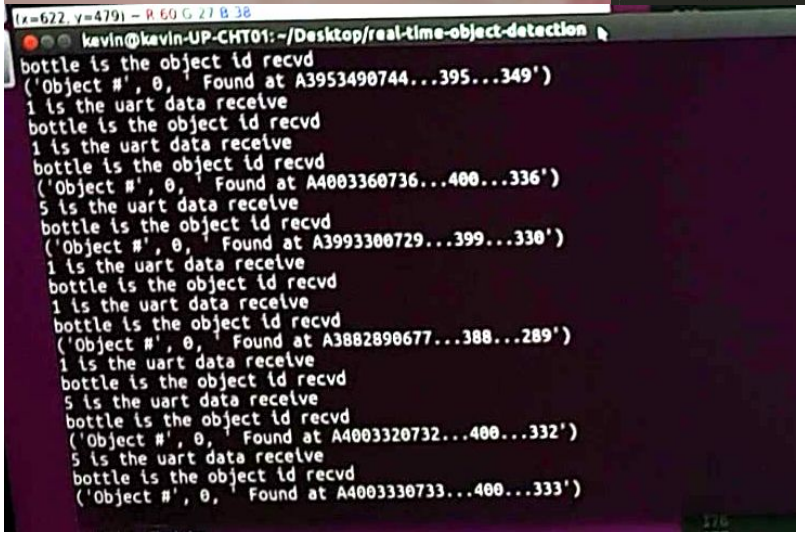
- Initializes 8O1, 115200 baud UART for pin 8 & 9. UART Timeout = 0.003 sec. GPIO pins as input for 5 & 13. Set object classes. Define device port for Realsense Camera at 640x480
- Main loop: Check active low GPIOs else run camera code. If active, transmit UART “L” or “R”
- Convert RealSense frames to OpenCV matrix and create blob image at 325x325. Use blob image to detect object class (shapes).
- Checks received UART for selected object ID from bluetooth. Default: Bottles/Cans. Others: Cars, Planes, People.
- Draws box around shapes and checks confidence then transmit center x, y coord through UART. If no detection, coord is all 0.
- 11 bytes. “A” + “x coord” + “y coord” + “x+y total”



SBC Software Solutions Continued

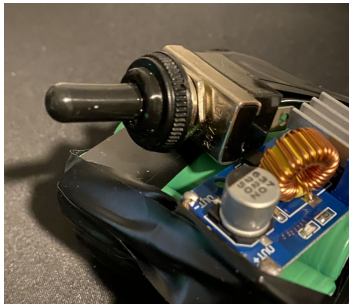
- Visible box and labels and text dumps are for debugging purpose. (can be hidden, including video stream to increase speed)
- Multiple same shapes can be detected but we only look at the first shape in the array.
- Object to be detected can be changed through bluetooth (default initialized to bottle)

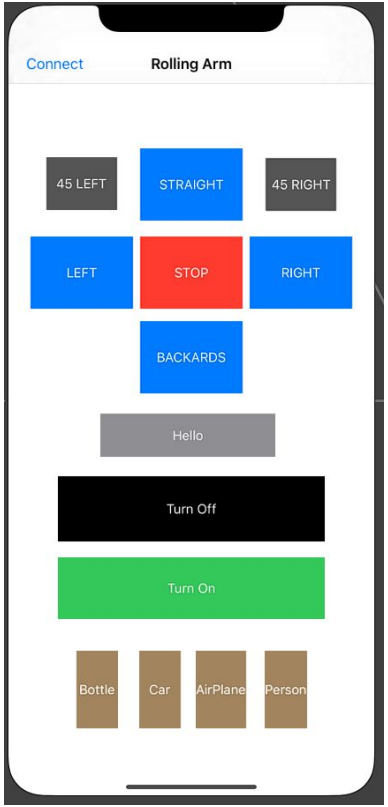

```
uartDataIn  = ser.read()

print(uartDataIn +" is the uart data receive")


if uartDataIn == '1':
    objectLooking = 'bottle'
elif uartDataIn == '2':
    objectLooking = 'car'
elif uartDataIn == '3':
    objectLooking = 'aeroplane'
elif uartDataIn == '4':
    objectLooking = 'person'
else:
    objectLooking = objectLooking


print(objectLooking +" is the object id recvd")


# draw the prediction on the frame
# used to check and display only specified object shape
if CLASSES[idx] == objectLooking:
```



User Interfaces

User interactions are minimal since the project aims to work autonomously other then turn on button and bluetooth for different object recognition.





Power Requirements and Solution

Consumer	Voltage (V)	Current (A)	Average Power (W)	Peak Power (W)
Up Board	5V	3-4A	15W	20W
Servo1	5V	500-900mA	2.5-4.5W	12W
Servo2	5V	500-900mA	2.5-4.5W	12W
Servo3	5V	500-900mA	2.5-4.5W	12W
Servo4	5V	500-900mA	2.5-4.5W	12W
Servo5	5V	500-900mA	2.5-4.5W	12W
Servo6	5V	500-900mA	2.5-4.5W	12W
Dc Motor1	12v	300-400mA	3.6-4.8W	30W
Dc Motor2	12v	300-400mA	3.6-4.8W	30W
TM4C Lunchpad	5v	23-328mA	115-1615mW	1.615W
Total Power			37.2-43W	153.6W

- High power system
- Intel Up board requires a lot of power.
- Supply constant power to Up board and TM4C Launchpad.

Power Requirements and Solution

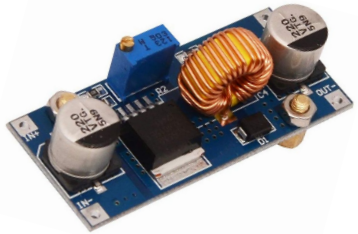
Rechargeable Battery 12v - 6A powers all motors

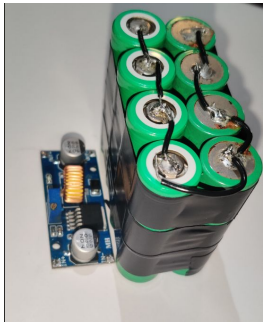
Regulates Voltage from 9 to 12v and it can deliver up to 6 Amps.

Lithium ion Samsung 25R-18650 delivers 25 Amps at 3.6V.

8-battery set can deliver up to 100 Amps at 7.2V

Output is regulated using a buck converter module to 5V.





Hardware Block Diagram

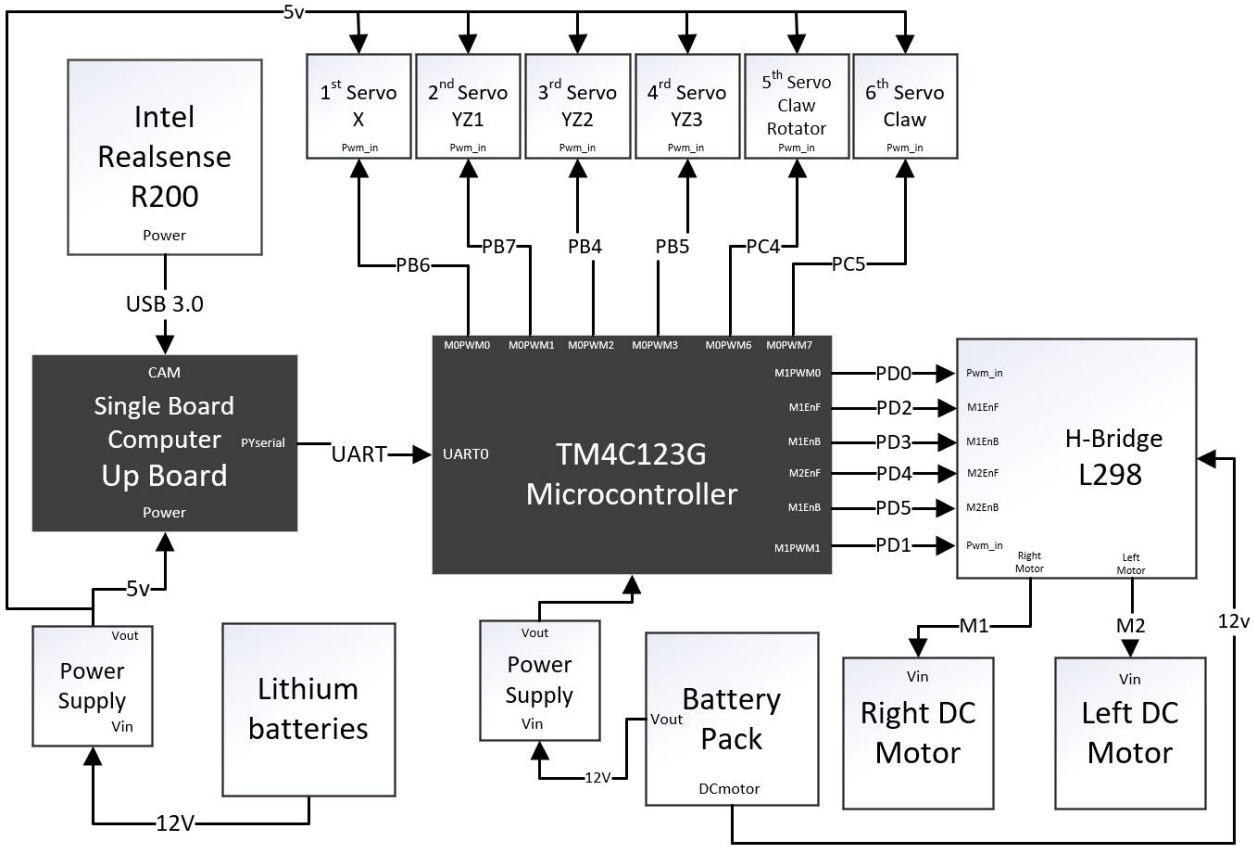
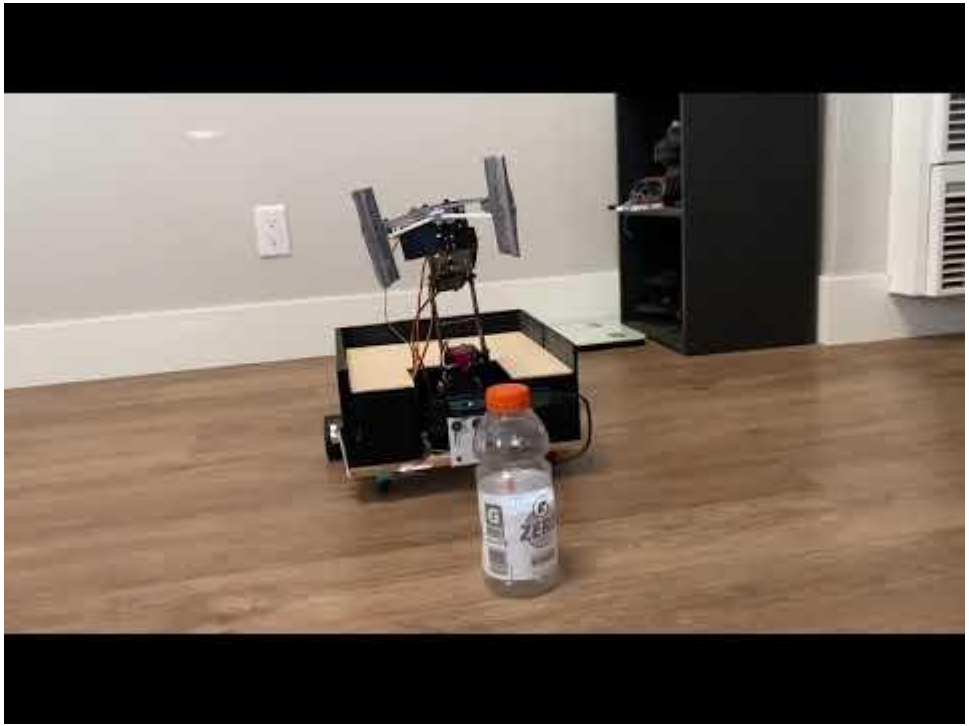


Image detection & Bluetooth app

Airplane and Car Detection

Intro Video



Rolling Arm

Rolling Arm