CLEMSON UNIVERSITY

Robot Arm Project:

Raghavendra Rao

Presentation Outline

- Level 1 Concept
  - Gripper Design
  - Block Handling
  - Sensor Integration
  - Pick and Place Strategy
  - Code Samples
- Level 2 Concept
  - Concepts for future work

Level 1 Concept

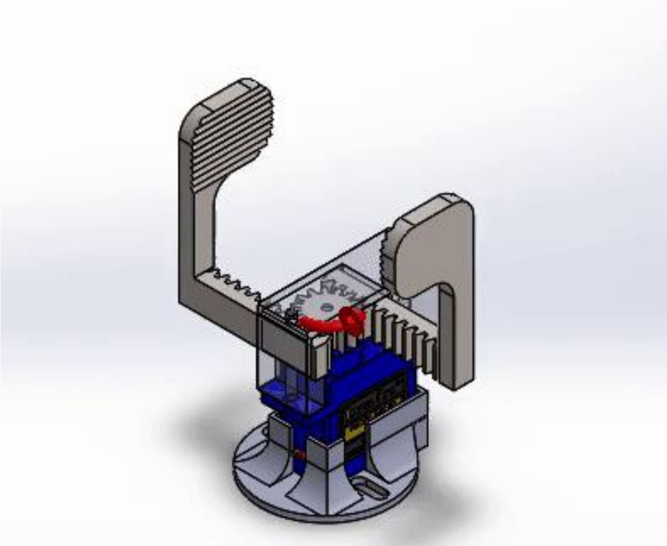
Robotic arm Design, Integration, Testing & Validation
for pick-and-place material movement

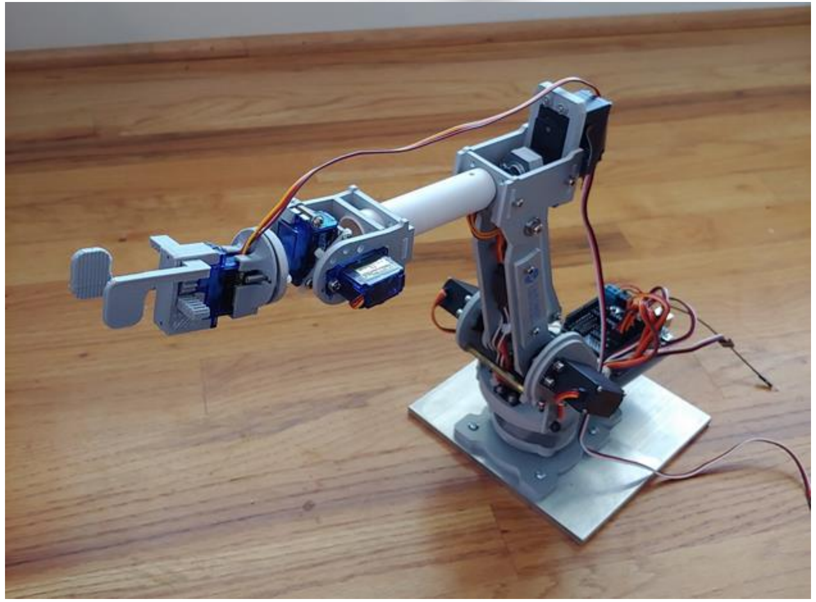
Introduction

- Known pick and place locations
- Sort blocks by color
- RGB sensor integrated to block magazine
- Pick and place strategy iterations
- Arduino code samples to achieve robot motion

Gripper

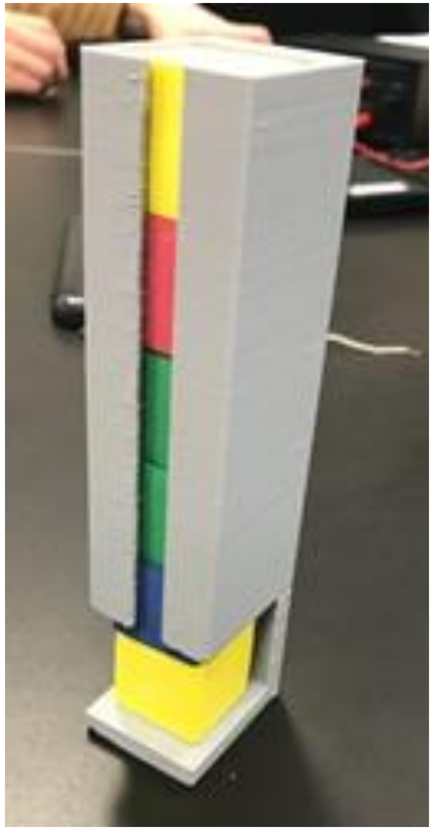
- A rack and pinon design for the gripper was chosen after experimentation with 3D-Printing a few different styles
- This design bolts to the mounting plate and aligns the enter of the grip with the center line of the arm
- Large surface area jaws with ridges help grip block without letting it rotate or slip between them during movement

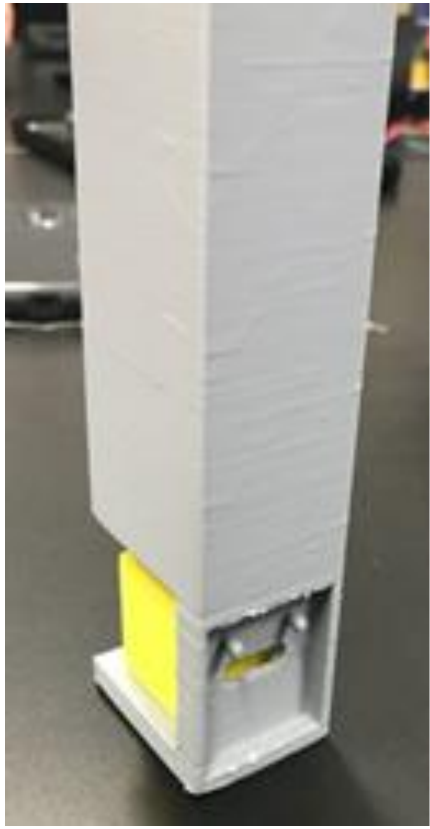




Block Magazine and Sensor Integration

- The vertical block magazine can hold up to 6 blocks.
- The RGB sensor will be integrated into our robotic system is mounting the sensor to the back of the block magazine.
- The RGB sensor will detect the color of the block. The colors that can be detected with the sensor is red, blue, green, and yellow.





Pick and Place Strategy

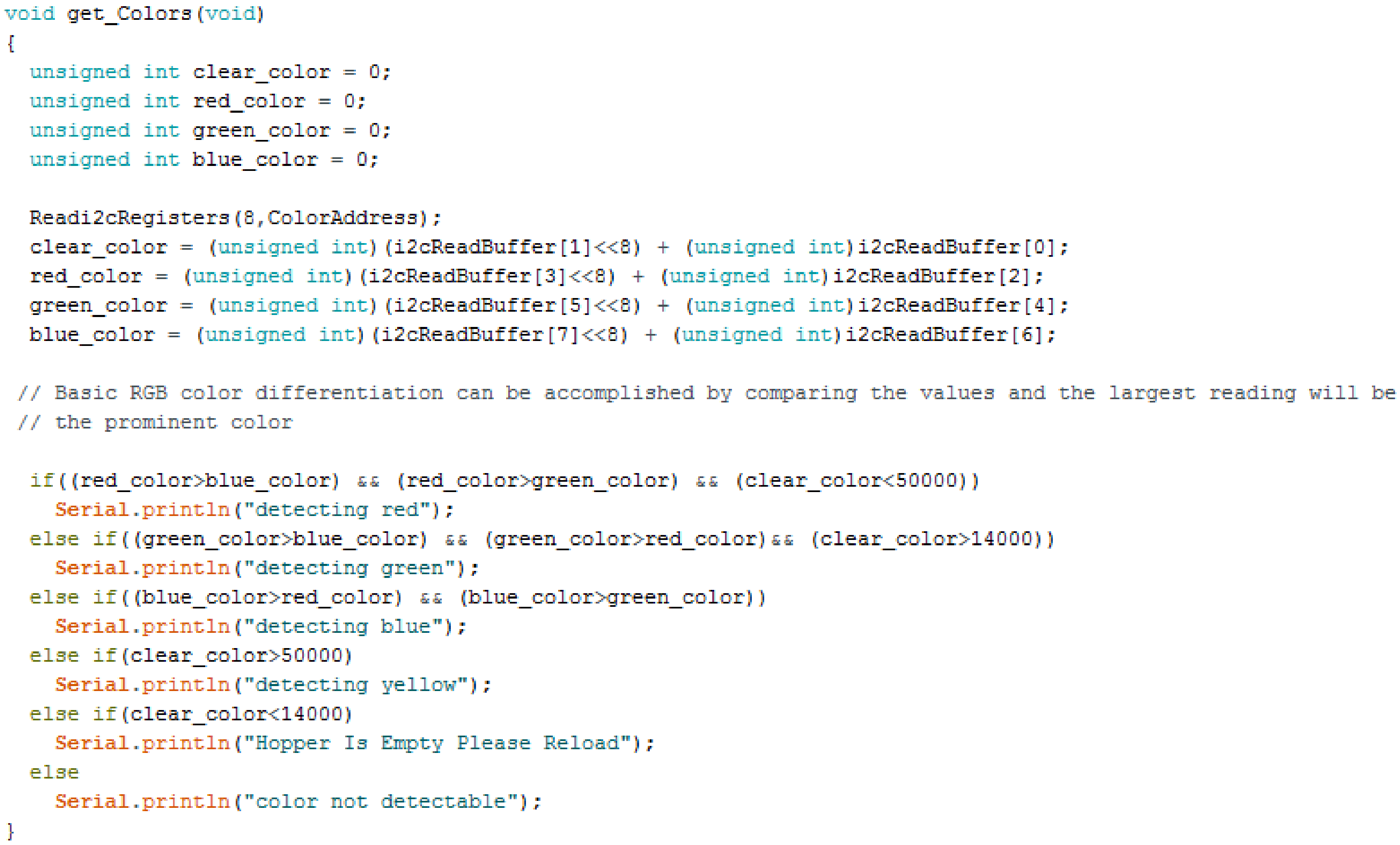
- 1. Rotate base servo to angle needed to pick up block from magazine.
- 2. Power elbow and wrist servos to angle needed to pick up block from magazine.
- 3. Power shoulder servo to angle needed to pick up block from magazine
- 4. Close pincher to grab block.
- 5. Using RGB sensor determine color and input into system, find for XYZ-Wrist angle place coordinates.
- 6. Solve for arm angles for placing.
- 7. Retract shoulder servo to neutral position.
- 8. Retract elbow servo to neutral position.

Pick and Place Strategy (continued)

- 9. Rotate base servo to angle needed to place block.
- 10. Power elbow and wrist servos to angle needed to place block.
- 11. Power shoulder servo to angle needed to place block.
- 12. Index XYZ-Wrist angle coordinate of color being placed to the next place location for that color block.
- 13. Open pincher to place block.
- 14. Retract shoulder servo to neutral position.
- 15. Retract elbow servo to neutral position.
- 16. Repeat.

Arduino Code Example

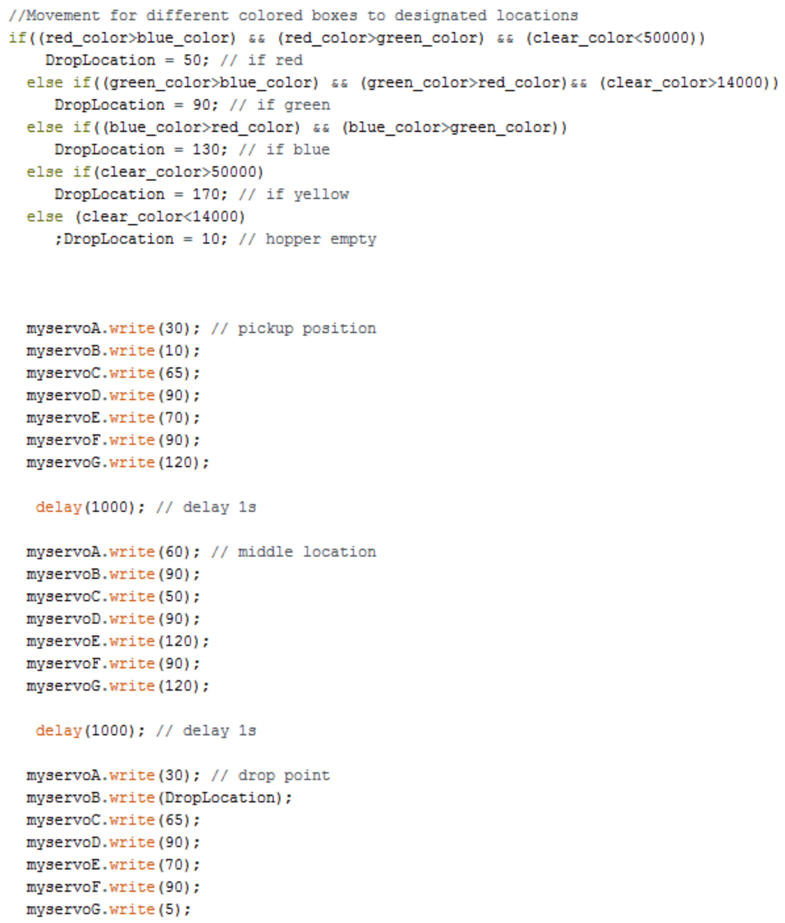
- Code example to read colors from RGB sensor



```
void get_Colors(void)
{
  unsigned int clear_color = 0;
  unsigned int red_color = 0;
  unsigned int green_color = 0;
  unsigned int blue_color = 0;

  Readi2cRegisters(8,ColorAddress);
  clear_color = (unsigned int)(i2cReadBuffer[1]<<8) + (unsigned int)i2cReadBuffer[0];
  red_color = (unsigned int)(i2cReadBuffer[3]<<8) + (unsigned int)i2cReadBuffer[2];
  green_color = (unsigned int)(i2cReadBuffer[5]<<8) + (unsigned int)i2cReadBuffer[4];
  blue_color = (unsigned int)(i2cReadBuffer[7]<<8) + (unsigned int)i2cReadBuffer[6];

 // Basic RGB color differentiation can be accomplished by comparing the values and the largest reading will be
 // the prominent color

  if((red_color>blue_color) && (red_color>green_color) && (clear_color<50000))
    Serial.println("detecting red");
  else if((green_color>blue_color) && (green_color>red_color)&& (clear_color>14000))
    Serial.println("detecting green");
  else if((blue_color>red_color) && (blue_color>green_color))
    Serial.println("detecting blue");
  else if(clear_color>50000)
    Serial.println("detecting yellow");
  else if(clear_color<14000)
    Serial.println("Hopper Is Empty Please Reload");
  else
    Serial.println("color not detectable");
}
```

Arduino Code Example

- Code example to place block based on color



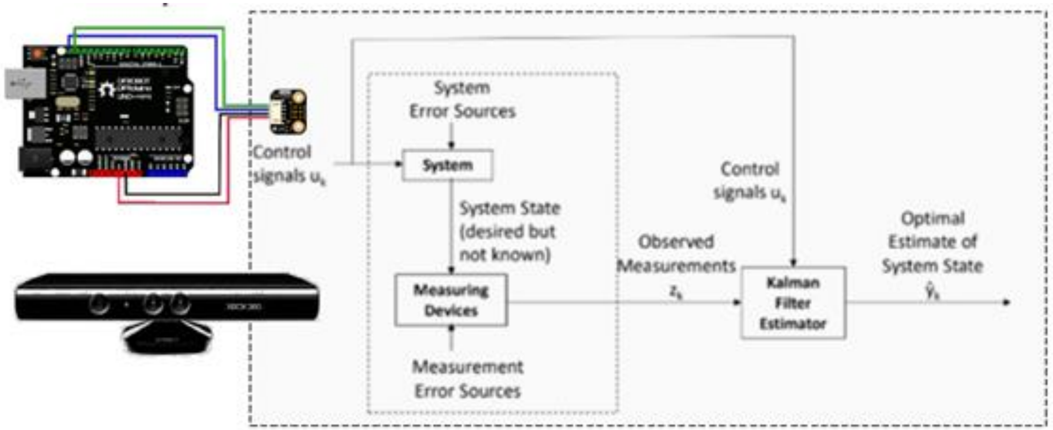
```
//Movement for different colored boxes to designated locations
if((red_color>blue_color) && (red_color>green_color) && (clear_color<50000))
    DropLocation = 50; // if red
  else if((green_color>blue_color) && (green_color>red_color)&& (clear_color>14000))
     DropLocation = 90; // if green
  else if((blue_color>red_color) && (blue_color>green_color))
     DropLocation = 130; // if blue
  else if(clear_color>50000)
     DropLocation = 170; // if yellow
  else (clear_color<14000)
     ;DropLocation = 10; // hopper empty



  myservoA.write(30); // pickup position
  myservoB.write(10);
  myservoC.write(65);
  myservoD.write(90);
  myservoE.write(70);
  myservoF.write(90);
  myservoG.write(120);

   delay(1000); // delay 1s

  myservoA.write(60); // middle location
  myservoB.write(90);
  myservoC.write(50);
  myservoD.write(90);
  myservoE.write(120);
  myservoF.write(90);
  myservoG.write(120);

   delay(1000); // delay 1s

  myservoA.write(30); // drop point
  myservoB.write(DropLocation);
  myservoC.write(65);
  myservoD.write(90);
  myservoE.write(70);
  myservoF.write(90);
  myservoG.write(5);
```

Level 2 Concept

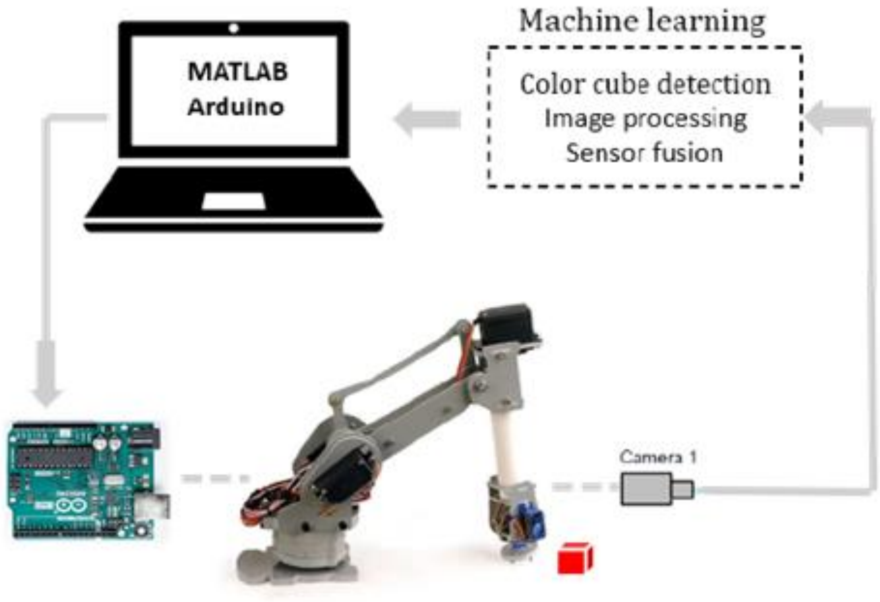
Computer vision, Sensor fusion, Machine learning

Future Scope

- Kinect for image processing to convert image coordinates from camera to pixel frame.
- Tsai’s model is used to calibrate the camera using the MATLAB code
- Correlation and feature based methods can be used depending on the requirements of a sparse disparity mapping of the surrounding.
- Kalman filter is used in the sensor fusion to refine the positioning data of the cubes.
- Implement motion planning using trapezoidal decomposition and probabilistic approaches
- Faster R-CNN can be used as a region-based approach for object detection



System Error Sources
System
Control signals u_k
System State (desired but not known)
Measuring Devices
Measurement Error Sources
Observed Measurements z_k
Control signals u_k
Kalman Filter Estimator
Optimal Estimate of System State $\hat{y}_k$



Machine learning
Color cube detection
Image processing
Sensor fusion
MATLAB
Arduino
Camera 1