Team 10 - Max Musing (260668840), Ziyin Wei (260706523), Jason Lau (260804731), Francis Duhamel (260746909), Yutian Jing (260680087)

Lab 5 - Search & Localize

ECSE 211: Design Principles and Methods

Design Evaluation

Hardware

Our robot uses two wheels attached to two motors that are linked to the output ports of the EV3 brick as a way of motion. The wheelbase is 17cm which prevents the robot from easily falling on its side. A ball has been mounted at the back of the robot to provide additional support while offering low friction with the floor. Two horizontal pieces have been added in order to align the robot with the 0-degree axis using the wall. This allows us to correct the angle of the robot in case of minor errors during the localization routine.

The first light sensor is located at the back of the robot (behind the support ball) and is mounted in such a way that it is oriented towards the floor and is really close to the surface. This proximity with the ground allows us to mitigate the impact of noise picked up by the light sensor. It is this sensor that is used by the localization algorithm. The second light sensor is located at the front of the robot and is attached to an arm which puts the sensor about at 10.5 cm from the ground, just 5 mm higher than the color blocks. This also allows us to reduce the ambient light and gives us more accurate data. The sensor is mounted so that it points toward the floor, which eliminates the need to move the sensor and, therefore, reduces the sources of errors. The last sensor, the ultrasonic sensor, is attached to the robot on a third motor and allows the sensor to around a vertical axis. This feature is used during the searching routine of the program.

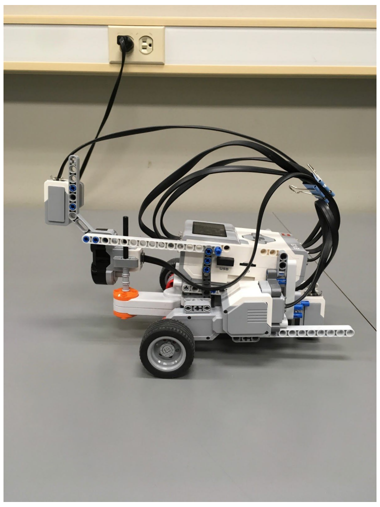


Fig. 1: Side View

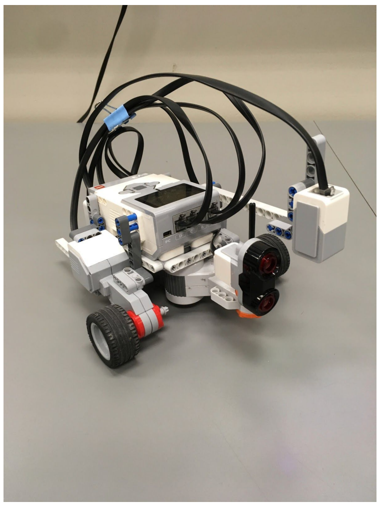


Fig. 2: Perspective View

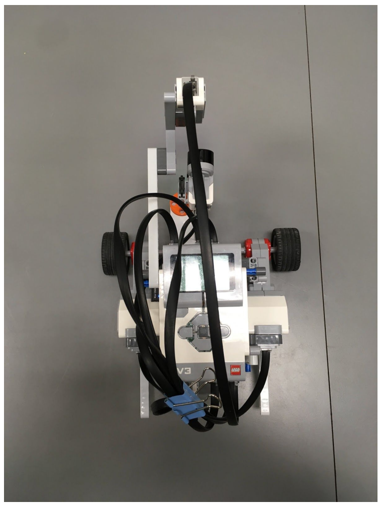
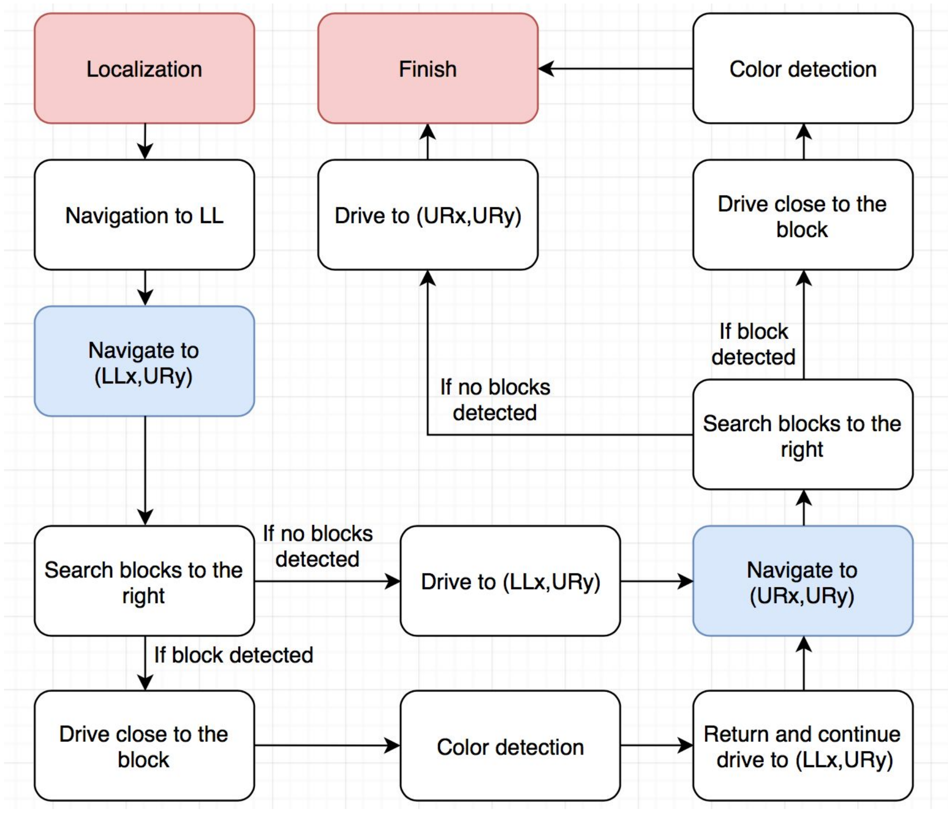


Fig. 3: Top View

Software

The robot's software search algorithm is made up of three main parts: localization, navigation, and searching.



Our localization implementation is similar to lab 4, but with an added hardware correction. After localizing its angle with the ultrasonic sensor, out robot backs up against the wall to ensure its angle is perfectly aligned. This corrects any potential error before light localization even begins. This then frees us to use a simpler implementation of light localization by simply correcting the x and y positions of the odometer independently using 1 vertical line and 1 horizontal line.

After localizing to (1,1) our robot then uses basic navigation to travel to the lower left corner of the search area. If starting at the far corner (starting corner 2) the robot first navigates to an adjacent corner before navigating to the lower left corner to avoid blocks in the search area. We decided not to implement correction for this since it is such a short distance to travel.

Once the robot has navigated to the lower left corner of the search area, it begins its search for the target block. The search algorithm is as follows:

```
SearchPseudocode():
    set waypoints <- { (LLx, URy), (URx, URy) }

    for each waypoint in waypoints
        while not at waypoint
            if usDistance < search area size
                turn right
                turn ultrasonic sensor forwards

                while usDistance < 5
                    move forwards

                if colourDetected = targetColour
                    beep once
                else:
                    beep twice

                move back to line
                turn left
                turn ultrasonic sensor right
            else
                move forwards
```

This algorithm will travel to each block until the target block is found, finally ending up at the upper right corner once finished.

Test Data and Test Analysis

Blue RGB values

Times	Red	Green	Blue
1	0.033326	0.059804	0.058824
2	0.041176	0.064706	0.062745
3	0.038235	0.054902	0.054902
4	0.038235	0.055882	0.055882
5	0.038235	0.064706	0.064706
6	0.038235	0.065686	0.065686
7	0.039216	0.057843	0.057843
8	0.039216	0.056863	0.057843
9	0.037255	0.056863	0.058824
10	0.037255	0.055882	0.057843
Mean	0.038039	0.059314	0.05951
Standard Derivation	0.002006	0.004166	0.003641

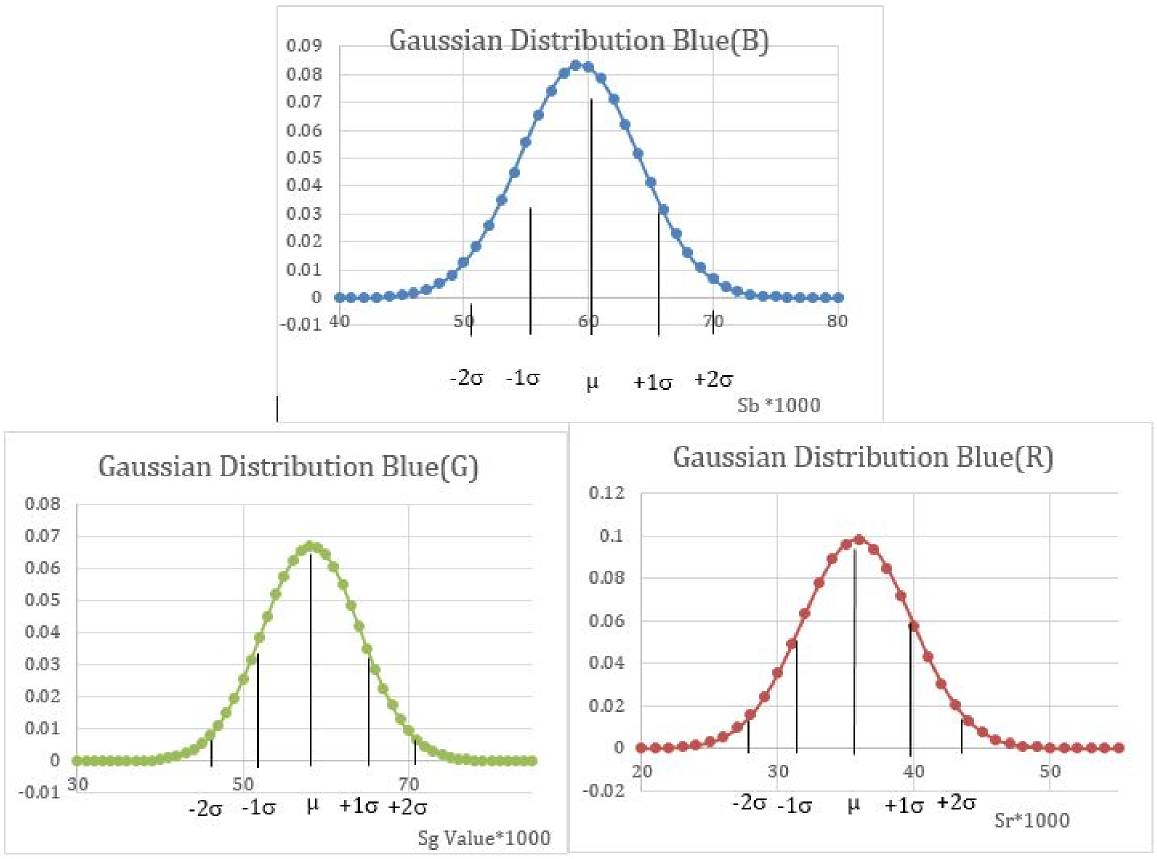


Fig.4:Gassian Distributaion blue block

Red RGB values

Times	Red	Green	Blue
1	0.228431	0.029412	0.018627
2	0.227451	0.029412	0.018627
3	0.209804	0.027451	0.015686
4	0.209804	0.027451	0.015686
5	0.198039	0.024511	0.014706
6	0.195098	0.024512	0.013725
7	0.188235	0.024513	0.012745
8	0.192157	0.026471	0.014706
9	0.189216	0.025497	0.014706

10	0.188235	0.025497	0.014706
Mean	0.202647	0.026471	0.015392
Standard Derivation	0.014719	0.001808	0.001811

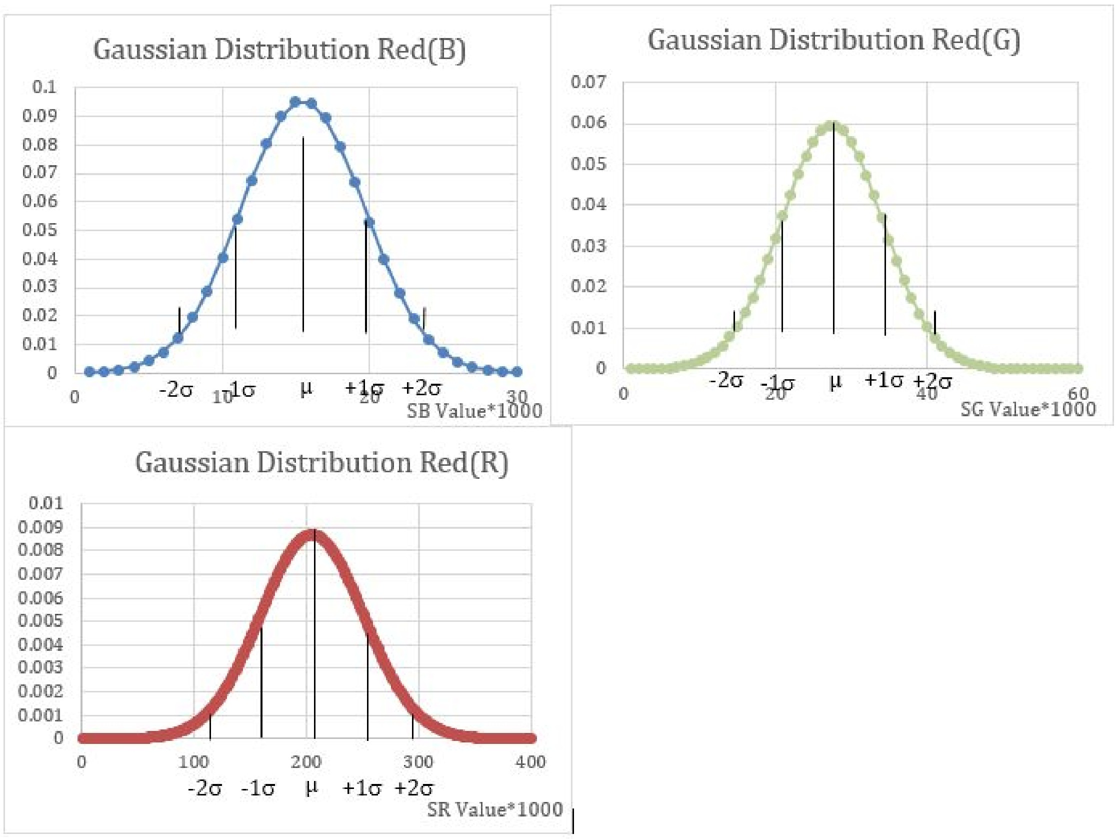


Fig.5:Gassian Distributaion red block

Yellow RGB values

Times	Red	Green	Blue
1	0.290196	0.212745	0.023529
2	0.287255	0.210784	0.023529
3	0.285294	0.209804	0.023529
4	0.286275	0.210784	0.023529

5	0.309804	0.222549	0.022549
6	0.306863	0.221569	0.022549
7	0.297059	0.213725	0.020588
8	0.293137	0.211765	0.020588
9	0.296078	0.213725	0.022549
10	0.293137	0.211765	0.022549
Mean	0.294512	0.213922	0.022549
Standard Derivation	0.007894	0.004246	0.001074

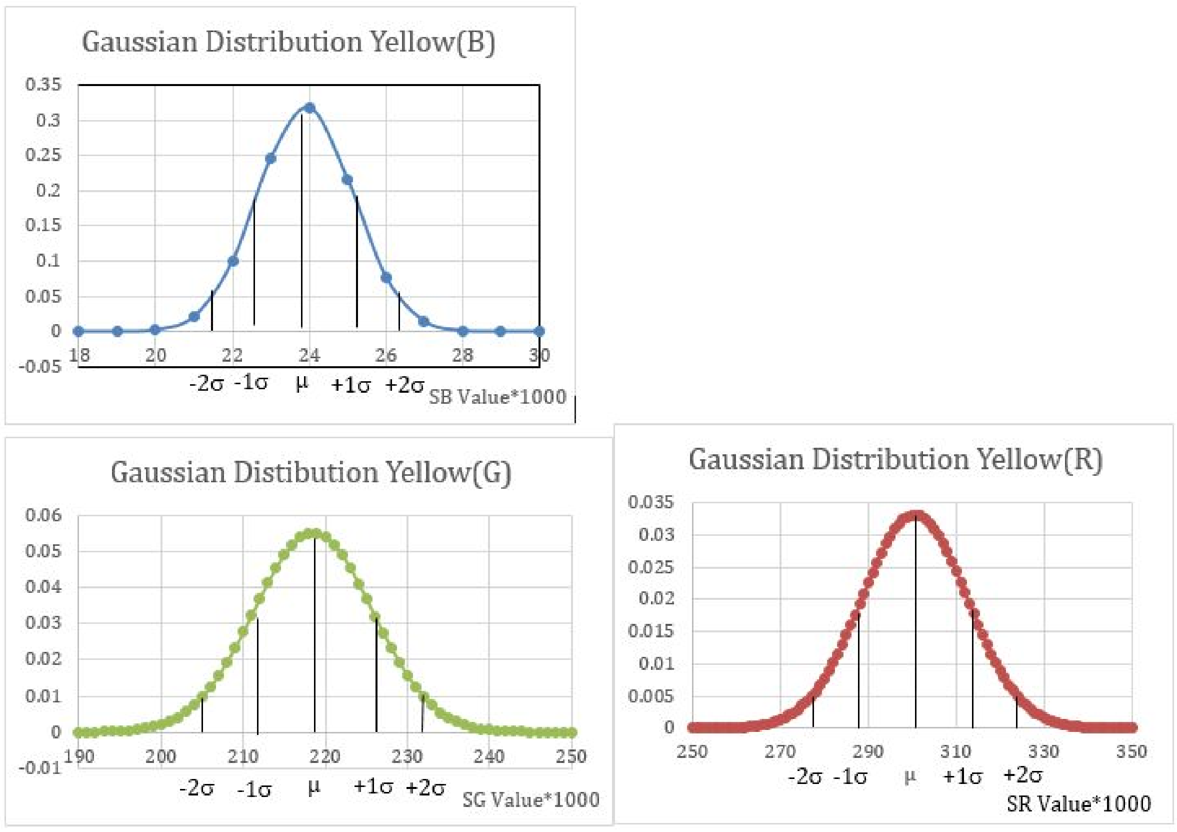


Fig.6:Gassian Distributaion yellow block

White RGB values

Times	Red	Green	Blue
1	0.366667	0.287255	0.167647

2	0.376471	0.301961	0.173529
3	0.342157	0.261765	0.154902
4	0.349021	0.271569	0.160784
5	0.343137	0.259804	0.152941
6	0.353922	0.272549	0.159804
7	0.322549	0.236275	0.140196
8	0.331373	0.243137	0.145098
9	0.318627	0.236275	0.140196
10	0.323529	0.241176	0.143137
Mean	0.342745	0.261176	0.153824
Standard Derivation	0.018337	0.021328	0.011078

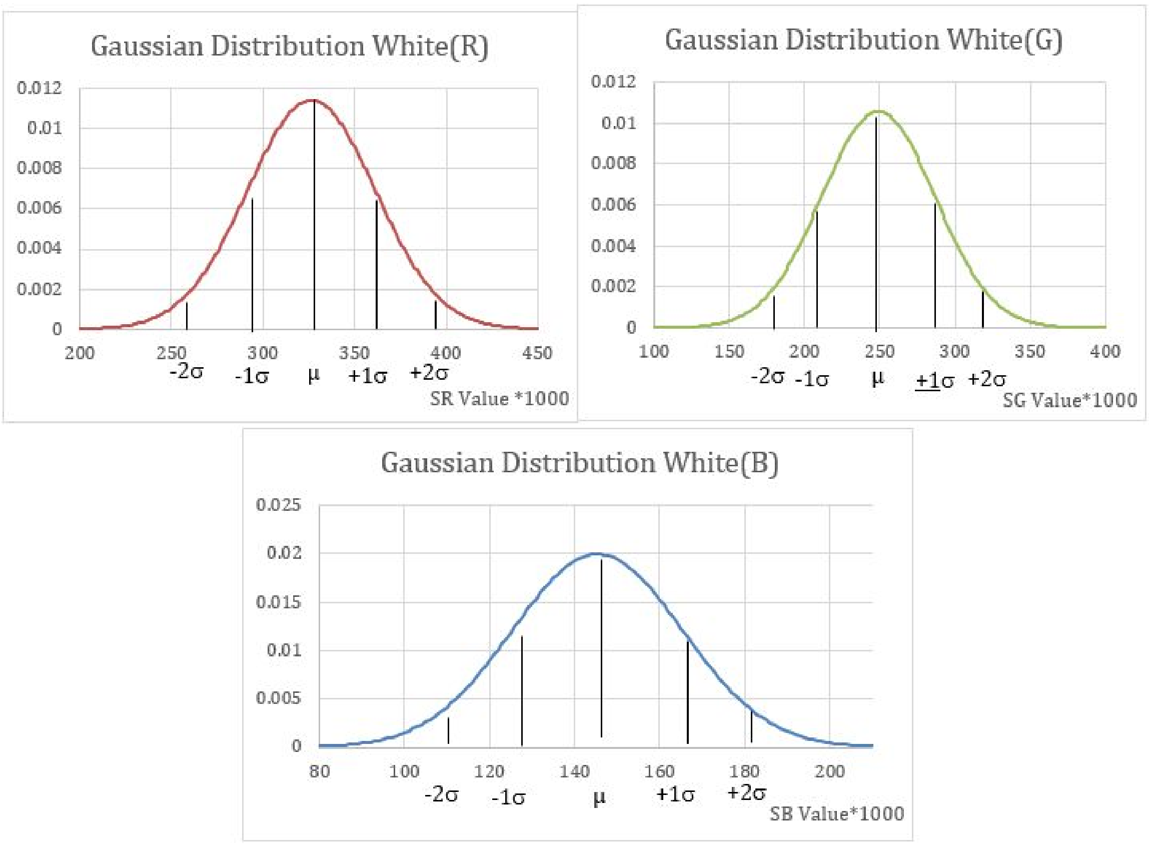


Fig.7:Gassian Distributaion white block

Color and Position Identification

First Trial

Target Block		R	G	B	TPEx	TPEy	TPRx	TPRy
X	Blue	0.033333	0.060784	0.063725	3.1	3.4	4.4	4.2
	Red	0.188235	0.02451	0.012745	4.1	4.9	5.5	5.7
	Yellow	0.293137	0.214706	0.02451	5.5	3.2	6.7	4.2
	White	0.331373	0.243137	0.145098	2.9	4.1	4.4	5.5

Euclidean Distance (first trial)

Blue	0.006482
Red	0.020123
Yellow	0.002449
White	0.024481

Second Trial

Target Block		R	G	B	TPEx	TPEy	TPRx	TPRy
	Blue	0.037255	0.064706	0.063725	3.6	3.6	4.4	4.2
X	Red	0.192157	0.026471	0.014706	4.2	4.9	5.5	5.7
	Yellow	0.296078	0.215686	0.02451	6.1	4.1	6.7	4.2
	White	0.318627	0.236275	0.140196	3.1	4.2	4.4	5.5

Euclidean Distance (Second trial)

Blue	0.004142
Red	0.028124
Yellow	0.002217
White	0.043571

Third Trial

Target Block		R	G	B	TPEx	TPEy	TPRx	TPRy
	Blue	0.037255	0.055882	0.057843	3.7	3.6	4.4	4.2
	Red	0.189216	0.02549	0.014706	4.9	5.1	5.5	5.7
X	Yellow	0.29117	0.21274	0.02451	6.1	4.1	6.7	4.2

		6	5					
	White	0.323529	0.241176	0.143137	2.9	4.1	4.4	5.5

Euclidean Distance(Third trial)

Blue	0.0045821
Red	0.016382
Yellow	0.001823
White	0.031414

Fourth Trial

Target Block		R	G	B	TPEx	TPEy	TPRx	TPRy
	Blue	0.039216	0.057843	0.057843	3.6	3.6	4.4	4.2
	Red	0.188235	0.02549	0.014706	4.2	5.8	5.5	5.7
	Yellow	0.287255	0.210784	0.02451	6.8	4.3	6.7	4.2
X	White	0.366667	0.287255	0.162745	4.7	4.7	4.4	5.5

Euclidean Distance (Fourth trial)

Blue	0.003125
Red	0.019274
Yellow	0.002384
White	0.035915

Rank order table (Euclidean Distance for each block)

Blue

	Classification
0.031258	Blue
0.004142	Blue
0.004582	Blue
0.006482	Blue

Red

	Classification
0.016382	Red
0.019274	Red
0.020123	Red
0.028124	Red

Yellow

	Classification
0.001823	Yellow
0.002217	Yellow
0.002384	Yellow
0.002449	Yellow

White

	Classification
0.024481	White
0.031414	White
0.035915	White
0.043571	White

Sample Calculations

Red mean

$$\frac{\sum_1^n x_i}{n}$$

$$\frac{0.033326 + 0.041176 + 0.038235 * 4 + 0.039216 * 2 + 0.037255 * 2}{10}$$

=
=0.038039

Red standard deviation

$$\sqrt{\frac{\sum_1^n (x_i - u)^2}{n}}$$

=

$$\sqrt{\frac{(0.033326 - 0.038039)^2 + (0.041176 - 0.038039)^2 + 4 * (0.038235 - 0.038039)^2 + 2 * (0.039216 - 0.038039)^2 + 2 * (0.037255 - 0.038039)^2}{10}}$$

From the above test data, conclusions can be made that the RGB mode has a high and stable performance, if the color sensor is operating at a stable and fixed height from the target. Low light conditions would have slight effect on the value retrieved from the sensor, however the RGB combination will still remain a proper portion and thus the sensor would not be affected severely. ColorID mode is also tested, with a value of [0, 2, 3, 6] to [red, blue, yellow, white] respectively, it is simpler to use as only integer values of different color will be returned from the sensor, however the ColorID mode has potential unpredictable variations on its performance, for instance if the light condition is low or there are scratches on the color blocks, in which case the RGB mode will be a more operable and stable option as each color portion can be limited within a threshold value, resulting more precise color detections.

In regard of the ultrasonic sensor, the distance the sensor detects is not straight ahead from the sensor, but with some degree prone to the left or right. The shift angle is measured and estimated to be within 20 degrees. This would have much an influence on the performance of the search-block algorithm, however it can be fixed using software compensation by trigonometric values calculated after many tests.

Observations and Conclusions

1. Are rank-ordering Euclidean distances a sufficient means of identifying block colors? Explain in detail why or why not.

Rank-ordering Euclidean distance is a relatively efficient way of identifying block colors, it certainly is not the best way as there are few problems we need to take into account.If the minimum distance is not set properly then the results based on it will be inaccurate and the correct colors will not be identified.And sometimes if the Euclidean distance is large and is not within the minimum distance, it simply says it is not right and ignores it. But it could actually be the right color since it is within a few standard deviations.

2. Is the standard deviation a useful metric for detecting false positives? In other words, if the block color determined using the Euclidean distance metric, d, is incorrect, can this false positive be detected by using $\mu \pm 1\sigma$ or $\mu \pm 2\sigma$ values instead?

When calculating the standard deviation, more data is used which makes it more accurate. But when calculating the Euclidean distance, we just simply use the mean with the assumption that they are indeed the perfect mean and the difference between the mean and the measurements will be the noises and errors.The noise and errors are indistinguishable using this method. By using standard deviation, the range of noise is estimated from the initial measurements and the cause of difference can be determined whether it is noise or error. General speaking , standard deviation would not give false positives. For instance, the Euclidean distance gives a negative and a standard deviation gives positive, the conclusion can be made that Euclidean distance is wrong and it is giving false negatives.

3. Under what conditions does the color sensor work best for correctly distinguishing colors?

The light sensor works best under the conditions that we initially measured the RGB values at (in the lab). We were able to reduce this dependence by placing the light sensor directly above the blocks, reducing the effects of lighting conditions. Our colour detection algorithm was based on simple absolute ranges of red, green, and blue values for each block. We found that we could distinguish values effectively using this simple technique under differing light conditions.

Further Improvements

1. Depending on how you implemented your color classifier, can your results be improved by using one or more of the noise filtering methods discussed in class?

Another way to filter noise that we have seen in class is the median filter. By using this filter, it eliminates all the values that are greater than the median and replace them with median. By doing so, it gets rid of all noise that potentially affect the color classification.

2. How could you improve the accuracy of your target block's position identification?

To improve the accuracy of the block's position identification, during the search routine, once we notice the presence of a block, we could stop at the halfway point of the distance between the robot and the block. We can then sweep and realign the robot with the block. This would prevent the robot from continuing forward and go past the block.

After identifying the target block, moving to a corner and using odometry correction along the way would allow the robot to provide a block position that is independent from the earlier portion of the routine. That way, if any errors occurred previously, unless they are major errors, their effect on the position of the target block would be nullified by the fact that the position is calculated after the identification.