To: Dr. Berry
From: Knut Peterson and Garrett Jacobs, Arkin
Date: 1/12/20
Re: Lab 02 - Avoid-Obstacle and Random Wander

Introduction

The purpose of this lab was to introduce subsumption architecture for the robot Arkin. The layers of the subsumption architecture included behaviors to randomly wander, avoid obstacles, and approach goals. The layers were then combined to form smart behaviors such as randomly wandering while avoiding obstacles, or moving to a goal while avoiding obstacles.

Method

For our IR sensor calibration we placed Arkin 20 inches away from a wide wooden box and ran a function which gathered 30 analog distance values and gave us the average. We then proceeded to gather data for every other distance down until 1 inch away, then repeated those steps for the rest of the IR sensors on Arkin. Using those values we were able to generate an equation using a line of best fit, and use that equation to view the distance away from the object Arkin perceived he was. We were then able to compare the actual distance values to that of Arkins readings and determine the percent error. The same steps as above were repeated for the sonars at the front of Arkin's body.

The first layer we implemented was the shy kid behavior where we used a force vector approach to calculate motor speeds. We used the four IR sensors, one from each side of the robot, to determine the distance from the robot to the obstacle, and used the distances to determine motor speeds proportional to how far away the robot was from the obstacle.

For the aggressive kid behavior we used a simple condition where if the robot did not see anything using the front IR sensor it would drive forward, and if it did see something it would stop.

We decided to split random wander into turning and driving. Using the random function, the robot first decided to either turn or drive, then executed the movement with a random value.

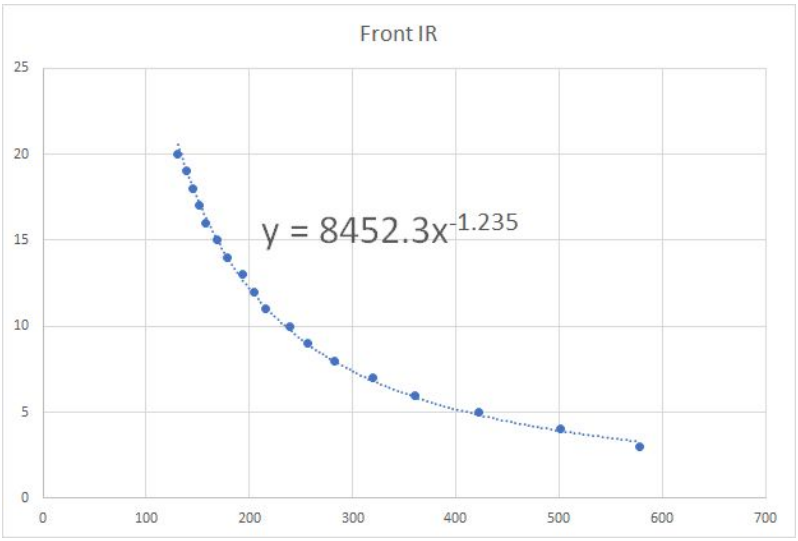
Smart wander combined the functionality of the shy kid behavior and the random wander behavior. If the robot detected an object out of any of the four IR sensors it would use the shy kid function to avoid the obstacle, and if no object was detected it used the random wander function to move around randomly.

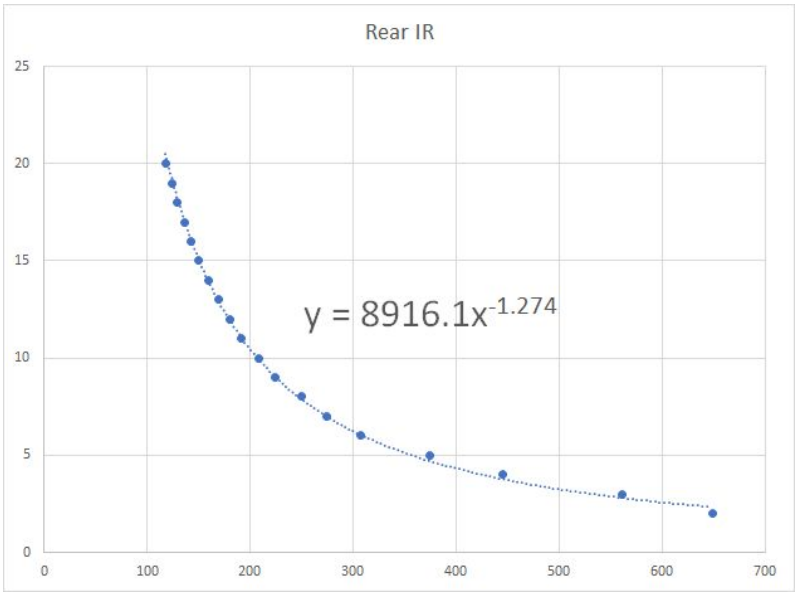
The final functionality was goal homing where the robot moved to the coordinates of a given goal while avoiding obstacles along the way. It was too difficult to track the robot's position when using the shy kid obstacle avoidance, so we decided to use a simpler method where if the robot saw an obstacle it would turn 90 degrees and drive until it couldn't see the obstacle. It would

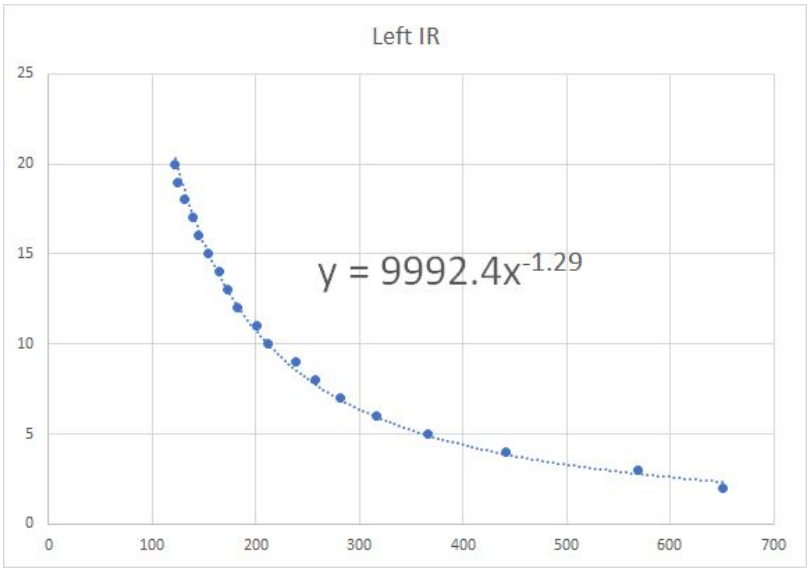
then turn again so that the obstacle was beside it again, then it would drive again until it had cleared the obstacle and would resume moving toward the goal from the new position.

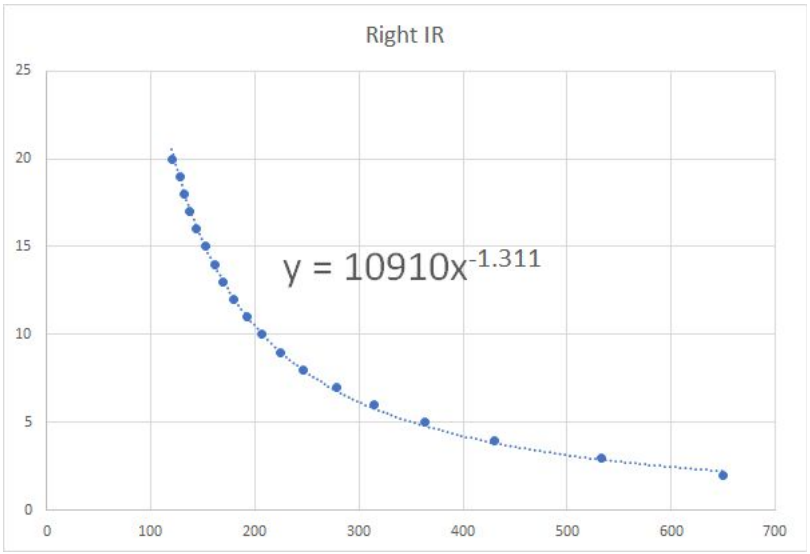
Results

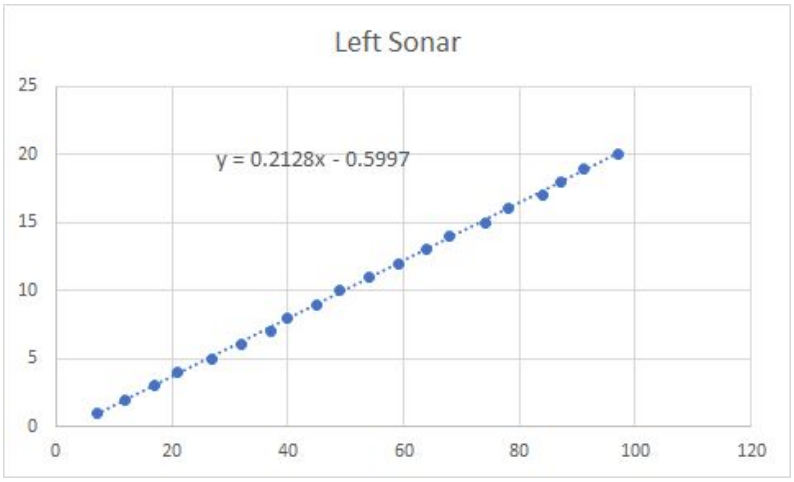
Below are the graphs and fit lines from the IR and sonar sensor calibrations.

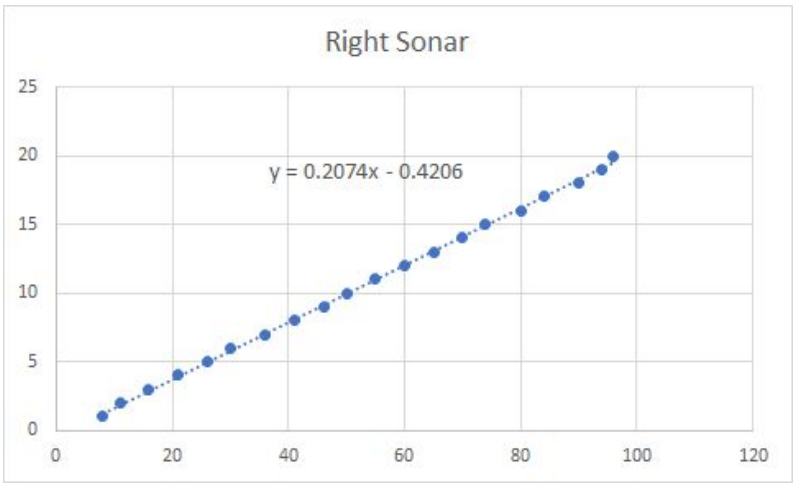





Left IR
25
20
15
10
5
0
0
100
200
300
400
500
600
700
y = 9992.4x-1.29


Right IR
25
20
15
10
5
0
0
100
200
300
400
500
600
700
y = 10910x-1.311





1. What was the general plan you used to implement the random wander and obstacle avoidance behaviors?

   We decided to split random wander into turning and driving. The robot first decided to either turn or drive, then executed the movement with a random value.

   For obstacle avoidance we decided to use the force vector approach which worked quite well, but we had to add a few special cases to alter if force vectors on the side added to or subtracted from the motor speeds.

2. How did you create a modular program and integrate the two layers into the overall program?

   We were able to reuse both the shy kid obstacle avoidance and random wander behaviors in the smart wander functionality. For the goal homing it was too difficult to track the position of the robot using the shy kid obstacle avoidance, so we used a

simpler approach where we turned away from the obstacle, moved until we couldn't see it anymore, and then resumed the go to goal behavior.

3. Did you use the IR and Sonar sensors to create redundant sensing on the robot's front half?

   No, we decided to only use the IR sensors. We had some slight trouble with dead zones that the sonar sensors could account for, but in the end the IR sensors were good enough.

4. How could you create a smart wander routine to entirely cover a room?

   Instead of just wandering randomly and avoiding obstacles, you could have the robot chose a random direction to drive in, then drive until it detects an obstacle. It could then turn away from the obstacle and choose new random direction to drive in.

5. What kind of errors did you encounter with the obstacle avoidance behavior?

   Our motors were unable to change speed proportionally so we had to do it discretely which, though functioned properly, left the robot quite shaky during performance

6. How could you improve the obstacle avoidance behavior?

   Find out the issue with our robot being unable to adjust speed proportionally, this would leave the speed changes much more fluid.

7. Were there any obstacles that the robot could not detect?

   There we some areas such as the corners of the robot where the IR sensors couldn't detect obstacles, but they were small enough that they didn't cause trouble.

8. Were there any situations when the range sensors did not give you reliable data?

   When an obstacle came too close to the sensors it fell within what is referred to as the "dead zone". Here the robot could in fact detect that there was an object there, but the robot perceived the range anywhere from 8 inches to 1 inch away

9. How did you keep track of the robot's states in the program?

   We used an "if" statement to check if the robot was detecting an obstacle. If it was, then it would run the code for that state, otherwise it would default to running the code for whatever other state was a part of the behavior.

10. Did the robot encounter any "stuck" situations? How did you account for those?

There were a few places where the IR sensors could not detect an obstacle, especially near the corners of the robot. This could cause the robot to get stuck in a few situations if it was driving forward, did not see an obstacle, and hit it with the it's wheel. For the go to goal behavior we addressed it by giving the obstacle a wide berth when driving around it so that we wouldn't clip it with the wheel on accident.

11. How did you keep track of the goal position and robot states as it integrated avoid-obstacle and go-to-goal behaviors?

    Whenever the robot turned we updated a variable for the current angle the robot was facing. Then by only moving in 1 inch increments, we were able to use the angle the robot was facing to keep track of the distance we traveled in both x and y.

12. What should the subsumption architecture look like for the addition of the go-to-goal and avoid-obstacle behaviors?

    The first layer would be to drive toward the goal, and the second layer would be to avoid obstacles. Whenever the robot sees an obstacle the second layer would override the first so it would avoid the obstacle. If an obstacle is no longer in sight, then the first layer would take over again and the robot would drive toward the goal.

Conclusions

In this lab we met the goals for robot performance. The random wander, shy kid and aggressive kid behaviors all ran well, and were able to be incorporated into each other modularly. The go-to-goal behavior was more difficult and was sometimes off of the goal by a few inches due to odometry error, but overall it was quite consistent and did the job effectively. One thing we could improve would be to implement the sonar sensors to help cover the dead zones where the IR sensors could not detect obstacles. While this did not cause large problems for us it would make the functions more consistent. Overall this lab was quite successful and we are pleased with Arkin's performance.

Appendix

Sonar Range Sensor Calibration

	Left Sonar			Right Sonar		
Actual Inches	Analog Values	Measured Inches	% error	Analog Values	Measured Inches	% error
1	7	1.55	55.0	8	1.8	80
2	12	2.22	11.0	11	2.65	32.5
3	17	3.51	17.0	16	3.58	19.33
4	21	4.33	8.25	21	4.67	16.75
5	27	5.38	7.60	26	5.54	10.8
6	32	6.47	7.83	30	6.7	11.67
7	37	7.75	10.7	36	7.61	8.71
8	40	8.53	6.63	41	8.64	8.00
9	45	9.69	7.67	46	9.71	7.89
10	49	10.77	7.70	50	10.52	5.20
11	54	11.43	3.91	55	11.67	6.09
12	59	12.55	4.58	60	12.56	4.67
13	64	13.58	4.46	65	13.42	3.23
14	68	14.62	4.43	70	14.53	3.79
15	74	15.69	4.60	74	15.57	3.80
16	78	16.6	3.75	80	16.49	3.06
17	84	17.25	1.47	84	17.56	3.29
18	87	18.35	1.94	90	18.35	1.94
19	91	19.51	2.68	94	19.49	2.58
20	87	20.67	3.35	96	20.29	1.45

Infrared Range Sensor Calibration

	Front IR			Rear IR			Left IR			Right IR		
Actual Inches	Analog Value	Measured Inches	% error	Analog Value	Measured Inches	% error	Analog Value	Measured Inches	% error	Analog Value	Measured Inches	% error
1	Dead Zone	Dead Zone	N/A	Dead Zone	Dead Zone	N/A	Dead Zone	Dead Zone	N/A	Dead Zone	Dead Zone	N/A
2	Dead Zone	Dead Zone	N/A	649	2.3	15.0	651	2.33	16.5	650	2.24	12.0
3	578	3.33	11.0	561	2.64	12.0	569	2.55	15.0	533	2.47	17.7
4	502	3.92	2.00	445	3.51	12.3	441	3.46	13.5	430	3.33	16.8
5	422	4.95	1.00	375	4.49	10.2	366	4.56	8.80	363	4.4	12.0
6	360	5.83	2.83	308	5.53	7.83	316	5.4	10.0	314	5.38	10.3
7	319	6.94	0.86	275	6.6	5.71	282	6.32	9.71	279	6.57	6.14
8	283	7.71	3.63	250	7.4	7.50	258	7.4	7.50	247	7.48	6.50
9	257	8.78	2.44	224	8.69	3.44	239	8.6	4.44	224	8.45	6.11
10	239	9.87	1.30	209	9.69	3.10	212	9.73	2.70	207	9.78	2.20
11	216	10.72	2.55	191	10.78	2.00	201	10.45	5.00	192	10.81	1.73
12	204	11.87	1.08	180	11.61	3.25	182	11.64	3.00	180	11.68	2.66
13	193	13.31	2.38	169	12.84	1.23	173	12.9	0.77	169	12.69	2.38
14	179	14.35	2.50	160	13.87	0.93	165	13.78	1.57	161	14.07	0.50
15	169	15.1	0.67	150	14.43	3.80	154	14.81	1.27	152	15.05	0.33
16	158	16.25	1.56	142	15.7	1.88	144	15.86	0.88	144	16.15	0.94
17	151	17.61	3.59	137	17.5	2.94	139	16.77	1.35	137	17.24	1.41
18	145	18.26	1.44	129	18.43	2.39	131	17.85	0.83	132	17.9	0.56
19	139	19.35	1.84	124	19.32	1.68	125	19.31	1.63	128	19.38	2.00
20	131	20.32	1.60	118	19.8	1.60	122	20.45	2.25	120	20.29	1.45

State Transition Table

Obstacle Avoidance			
States	Inputs	Outputs	Next State
Go To Goal	Given a goal	Obstacle	Random Wander
	No obstacle, has goal	Goal achieved	Avoid Obstacle
	Has goal	Has goal	Go To Goal
Avoid Obstacle	Obstacle	Has goal, no obstacle	Go To Goal
	Obstacle	No goal, no obstacle	Random Wander
	Obstacle	Obstacle	Avoid Obstacle
Random Wander	Goal achieved	Given a goal	Go To Goal
	No goal	Obstacle in the way	Avoid Obstacle
	No goal, no obstacle	No goal, no obstacle	Random Wander
	No goal, no obstacle		
Start		Has goal	Random Wander
		No goal	Go To Goal

State Diagram