Implementing Bug Algorithms on Sparki Robot

Kedar More
Mechanical Engineering
University of Colorado, Boulder
Boulder, USA
kedar.more@colorado.edu

***Abstract*—Motion Planning algorithms are a certain set of instructions given to the system which will search for the goal from the staring position automatically irrespective of the obstacles. The earlier algorithms were inspired by the motion of the bugs in the wild. These creatures have to find for food and also search for enemies. The same motion is implemented on the Sparki Robot with the limited sensors i.e Ultrasonic distance sensor and Ambient light sensor.**

***Index Terms*—Motion Planning algorithms, Sparki Robot, Ultrasonic distance sensor, Ambient light sensor**

I. Introduction

Bug algorithms are the most basic path planning algorithms which take inspiration from the motion of a bug. The bug in real life can sense the general direction of the target but can't look much further than a few meters away. Thus it uses its sense of touch to find out if there is an obstacle in front of it. Once the obstacle is detected the bug circumnavigates it to find the shortest path from the obstacle to the target. I intended to replicate the same behavior on Sparki Robot.

II. Literature Review

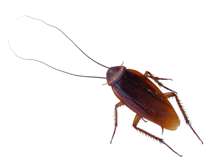

Figure 1. Bio inspiration for Bug algorithms

The bug algorithms were a product of pure observation and implementation was based on that particular motion. The bugs are clearly more intelligent than machines and can change their path according to their feedback. But in a steady and regular environment these algorithms perform better even with a little memory in the processor.

Comparing the bug algorithms it is concluded that each bug does good in some of the obstacle situations [1]. The maximum distance travelled by the robot with the perimeter of the obstacles 'p' and distance between start and end points is 'd' can be calculated as follows:[4] Bug 1 - d+p Bug 2 - d+p/2 Intelligent Bug - less than d+p/2 and more than d

Intelligent Bug Algorithm is a greedy algorithm which will find and go on the best path as soon as it finds it [3]. This is derived from the Bug 0 but in the Intelligent bug it also senses the obstacles much before it reaches them. This leads to reduction in distance travelled and therefore the time taken [2].

BugFlood algorithm is inspired by the motion of the ants where they divide the work so that it can be done faster [6]. Here the robot splits into two when it reaches an obstacle and searches for the shortest path and then all the parts come on the same route.

In the paper [5] we see the behavior pattern of cockroaches when they are put in a maze like situation. It mainly uses its touch receptor antennae to sense its surroundings. To learn the pattern to get out of the maze the roach follows the wall in any random direction till the target is achieved. Here the roach does not know the general direction of the target but uses the walls to reach it. The other bugs (terrestrial or air-borne) show similar behavior to find the food source or escape route.

The RAMBLER algorithm [7] is inspired by the motion of cockroaches. When the entire environment is known the other Path planning algorithms like Probabilistic Road Maps and Rapidly Exploring Random Trees do better in searching for a shortest path or a path with the least cost. But when the environment is not known the path is to be decided as the robot moves forward. Much like the cockroaches who go in search of food. This paper shows the behavior of cockroaches in different obstacle situations i.e the edges of the obstacles and their transparency. Whenever a flat obstacle ends the cockroach goes randomly in any direction in search of a next tactile encounter. Also when the cockroach is presented with a new obstacle after it has already identified the target, the new barrier was encountered twice per trial. The RAMBLER algorithm has 6 states namely Pivot, Arc Search for Contact, Walk Straight, Follow Wall, Steer to Goal, Explore Corner. The robot can randomly walk to find any obstacles or the goal. Once a corner is reached the robot turns at a random angle with a mean of 33 degree. The degree of randomness determines the percent of time the robot will reach the goal.

III. Components

A. Sparki Robot [8]

Saprki is a commercial robot which uses an Arduino controller with a number of sensors and actuators. It is programmed with the SparkiDiuno IDE.

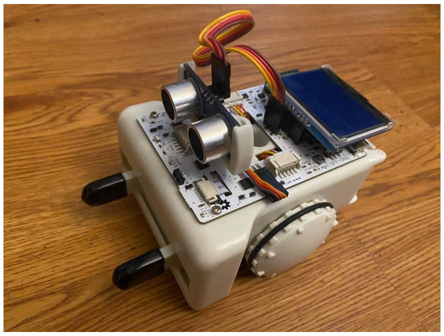

Figure 2. Sparki Robot

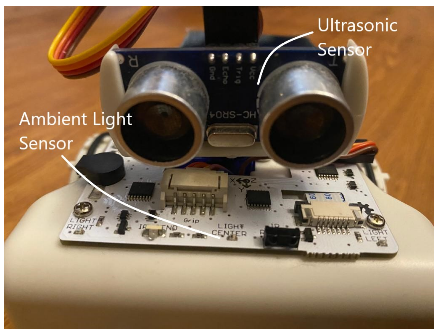

Figure 3. Components used

B. Ultrasonic Distance Sensor

This sensor uses the constant speed of sound travelling in air to calculate the distance between itself and the object in front of it.

It uses two transducers i.e. transmitter and receiver. Some pulse is given to the transmitter pin for 10 milliseconds and the timer is started with it. When the receiver get the high pulse the timer is stopped and the time difference is measured.

This time is multiplied by the speed of sound in air and divided by two. This will give the one sided distance to the object.

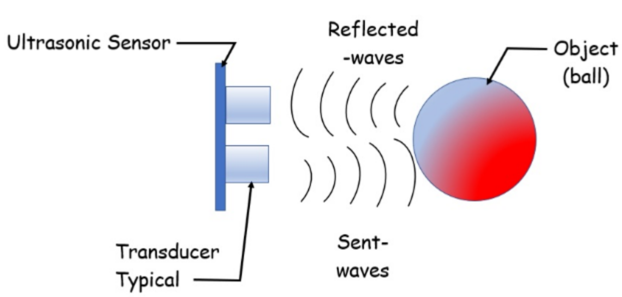

Figure 4. Working of an Ultrasonic Sensor

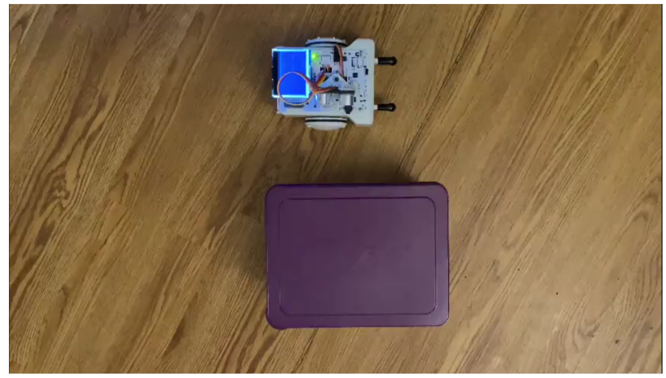

Figure 5. Sparki following the obstacle

C. Ambient Light Sensor

This sensor gives the value of ambient light as a lux value. So three sensors will be used i.e. left, center and right to gauge the approximate direction of the maximum luminosity. This is used to find the direction of the goal from the current location of the robot.

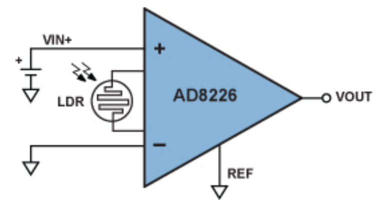

Figure 6. Working of an Ambient Light Sensor

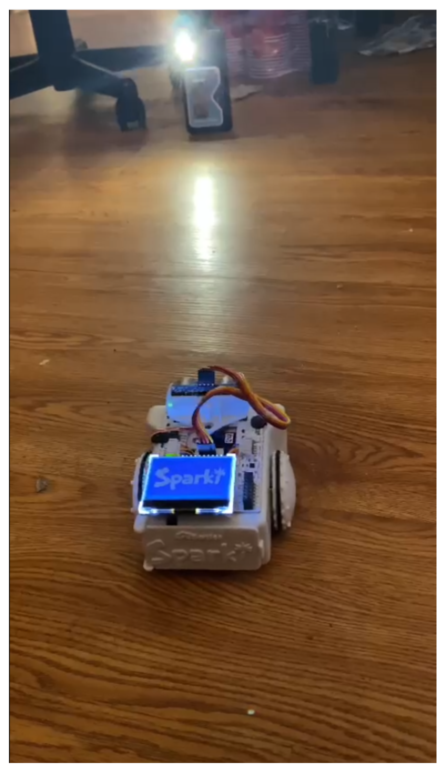

Figure 7. Sparki following light

The incoming light falls on the photo resistor which changes its resistance according to the intensity of the light. The OPAMP filters the raw data from the photo resistor and produces a smooth output.

IV. Algorithm and Implementation

A. Wall Following

The wall following algorithm uses a Proportional controller. The robot takes a feedback from the Ultrasonic distance and feed it to the controller which subtracts the distance from the set distance in which we have to keep the robot. This value tells if the robot is closer or further than expected. The error value is then multiplied by a constant called Kp which is the proportional constant. This value is then input to the actuators.

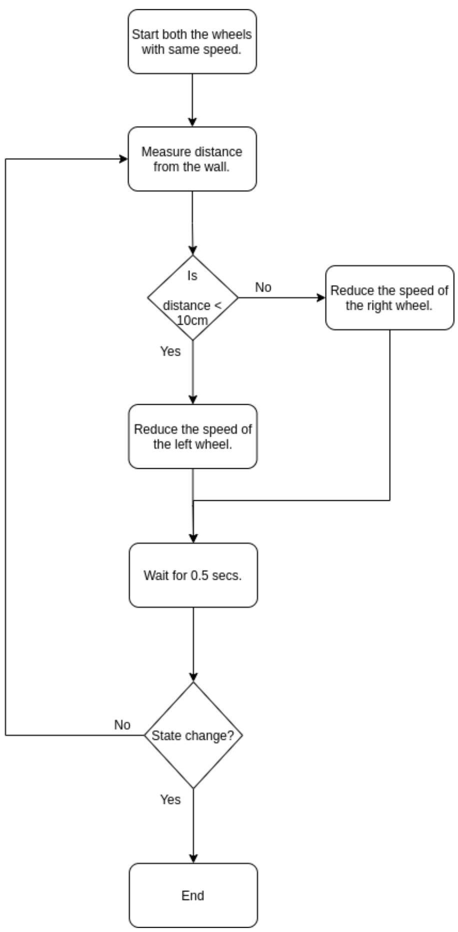

Figure 8. Wall Following Algorithm

B. Towards the goal

There are three light sensors on the front side of Sparki. These are used to gauge a general direction of a goal which is a light source.

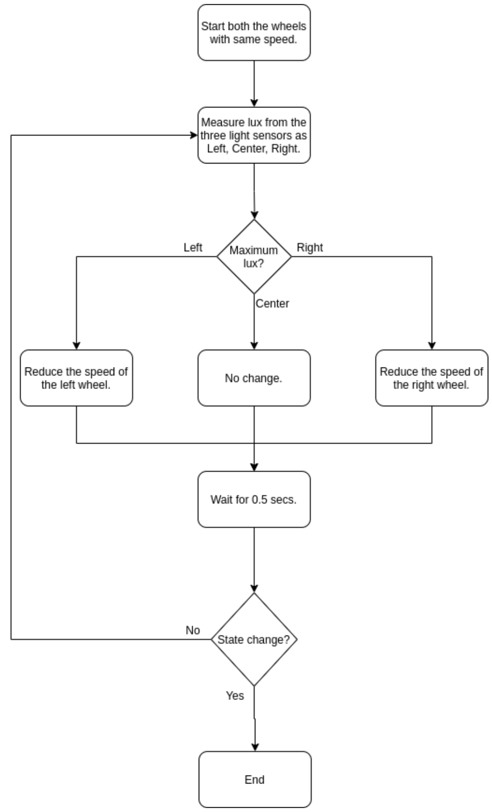

Figure 9. Towards the Goal Algorithm

C. Integration

States of the Robot: There are two states in which the robot will function namely wall_following and towards_goal. The states are decided according to the position of the robot and its next motion.

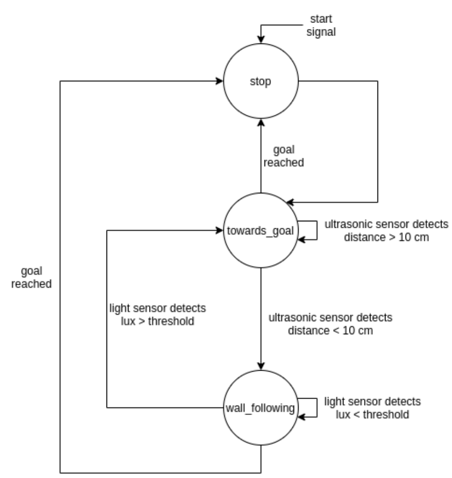

Figure 10. State Diagram

V. Observation

The use of light sensors can be difficult due to the reflected light incident on the sensors. This can be a good thing for this project as there is a small probability of the cockroach to go in any direction in search of the goal. The difference between the way cockroach detects any obstacles is with its antennae. It does not change the algorithm at all so we can use a distance sensor as our convenience.

This robot is much slower than a cockroach would travel. Hence the probability of the robot going in any direction randomly is absent here.

VI. Results

The robot was successfully able to find the goal and dodge the obstacles autonomously. This algorithm is complete with respect to finding the goal and it was verified that it can be implemented on a robot.

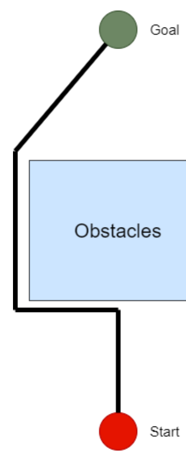

Figure 11. Bug 0 Path

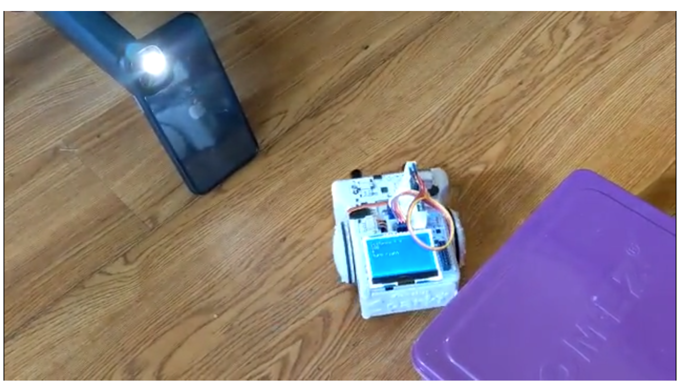

Figure 12. Snap from actual run

The above shown figure is one of the successful runs and the path taken.

References

[1] McGuire, Kimberly N., G. C. H. E. de Croon, and Karl Tuyls. "A comparative study of bug algorithms for robot navigation." Robotics and Autonomous Systems 121 (2019): 103261.

[2] Ng, James, and Thomas Bräunl. "Performance comparison of bug navigation algorithms." Journal of Intelligent and Robotic Systems 50.1 (2007): 73-84.

[3] Zohaib, Muhammad, et al. "IBA: Intelligent Bug Algorithm–A novel strategy to navigate mobile robots autonomously." International Multi Topic Conference. Springer, Cham, 2013.

[4] Chiang, Chia Hsun, Jing-Sin Liu, and Yo-Shing Chou. "Comparing path length by boundary following fast matching method and bug algorithms for path planning." Opportunities and Challenges for Next-Generation Applied Intelligence. Springer, Berlin, Heidelberg, 2009. 303-309.

[5] Turner, Charles Henry. "Behavior of the common roach (Periplaneta orientalis L.) on an open maze." The Biological Bulletin 25.6 (1913): 348-365.

[6] Sharma, Nishant, José Pinto, and P. B. Sujit. "BugFlood: A bug inspired algorithm for efficient path planning in an obstacle rich environment." AIAA Infotech@ Aerospace. 2016. 0254.

[7] Daltorio, Kathryn Mirletz, Brian Sterenstein, Andrea Cheng, Jui Watson, Adam Kesavan, Malavika Bender, John Ritzmann, Roy Quinn, Roger. (2014). How Cockroaches Employ Wall-Following for Exploration. 72-83. 10.1007/978-3-319-09435-9_7.

[8] ArcBotics - Sparki – Programmable Arduino STEM Robot Kit for Kids, http://arcbotics.com/products/sparki/.