# Optimizing the Turning Velocity in a Line Follower Robot

A.B.M. Khalid Hassan
EEE Department
American International University-Bangladesh

Ebad Zahir
EEE Department
American International University-Bangladesh

## ABSTRACT
In this paper, the analysis for data collected for two different line follower robots vehicles are presented. Keeping the hardware identical, the two vehicles were programmed with two different algorithms. When taking a turn, a correlation between its turning angle and the angle of line turn was observed. To find that correlation, experimental data was collected after varying critical parameters. After that, a mathematical representation was derived to calculate the required turning angle for any line angle.

## Keywords
Robotics, Algorithm, Turning Angle, Line Follower.

## 1. INTRODUCTION
When a robot wants to take a turn, the turning angle depends on angle of line turn. It does not depend on other parameters. But, other parameters should have to fulfill a minimum requirement. Like, the turning angle does not depend on torque of the motor or robot body weight. But, motor torque should have to be equal to or more than robot body weight. Again, the turning angle does not depend on the distance between wheels in a single axel. But, it should maintain a minimum distance to prevent capsizing at the time of taking turns. Also, the turning angle does not depend on the moving speed of a robot. It depends on the distance between sensor and front wheel/axel and distance between sensor and rear wheel/axel of a robot. This turning algorithm represents the relation between angle of line turn and angle of robot turn using these two parameters [1].

## 2. EXPERIMENTAL METHODOLOGY
The required turning angle of a robot for different line angle with varying the distance between sensor array and rear wheel/axel and the distance between sensor array and front wheel/axel was observed [2]. Some of the observations have been given in the data tables (see table 1 and table 2). For simplicity, Name of the different parameters have been symbolized. Let,

L = Distance between sensor array and rear wheel/axel (for example, see figure 1),

r = Distance between sensor array and front wheel/axel (for example, see figure 1),

$\Theta_L$ = Angle of line turn,

$\Theta$ = Required angle for a robot to take turn at $\Theta_L$.

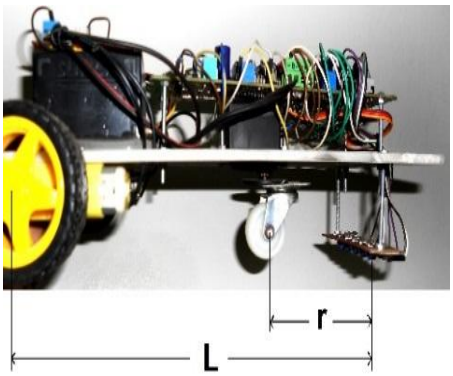


**Fig 1: L and r of a robot.**

### 2.1 Data Collected for Specific Variables
Data collected $\Theta$ for different L, r and $\Theta_L$ are given below (see table 1 and table 2). Here, all the distances are in centimeter (cm) and all the angles are in degrees (˚).

**Table 1. For L = 32**

| $\Theta_L$ | $\Theta$ for r=6 | $\Theta$ for r=8 | $\Theta$ for r=10 | $\Theta$ for r=12 | $\Theta$ for r=14 | $\Theta$ for r=16 |
|---|---|---|---|---|---|---|
| 18 | 14 | 10 | 8 | 7 | 6 | 5 |
| 36 | 28 | 20 | 16 | 14 | 12 | 10 |
| 54 | 41 | 31 | 25 | 20 | 18 | 16 |
| 72 | 55 | 41 | 33 | 28 | 24 | 21 |
| 90 | 70 | 52 | 41 | 34 | 29 | 26 |

**Table 2. For L = 28**

| $\Theta_L$ | $\Theta$ for r=6 | $\Theta$ for r=8 | $\Theta$ for r=10 | $\Theta$ for r=12 | $\Theta$ for r=14 | $\Theta$ for r=16 |
|---|---|---|---|---|---|---|
| 18 | 12 | 9 | 7 | 6 | 5 | 5 |
| 36 | 24 | 18 | 14 | 12 | 10 | 9 |
| 54 | 36 | 27 | 22 | 18 | 15 | 13 |
| 72 | 48 | 36 | 29 | 24 | 20 | 18 |
| 90 | 60 | 45 | 36 | 30 | 26 | 22 |

### 2.2 Graphical Correlations
All the observations, given in table 1 and table 2, have been plotted (see figure 2 and figure 3). To plot the graph, the x-axis is taken as distance between sensor array and front wheel/axel (r) and y-axis is taken as required angle for a robot to take turn at particular angle of line turn ($\Theta$). Each figure has five different graphs where each graph stands for different angle of line turns ($\Theta_L$). And different figures stand for different distances between sensor array and rear wheel/axel (L).

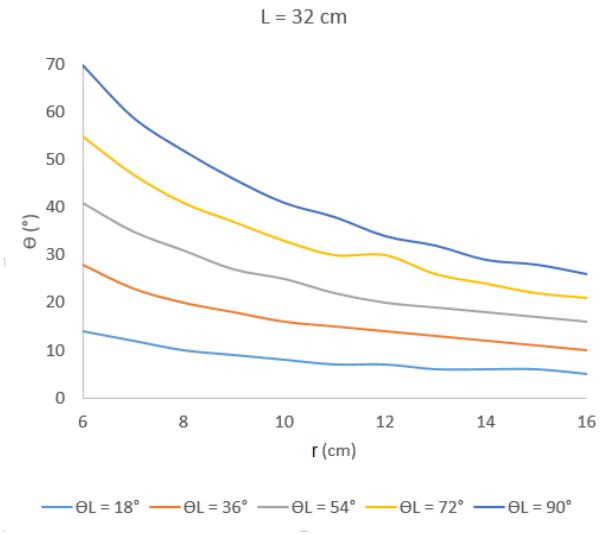


**Fig 2: Correlation of Distance between Sensor Array and Front Wheel with Angle of Line Turn for L=32cm.**

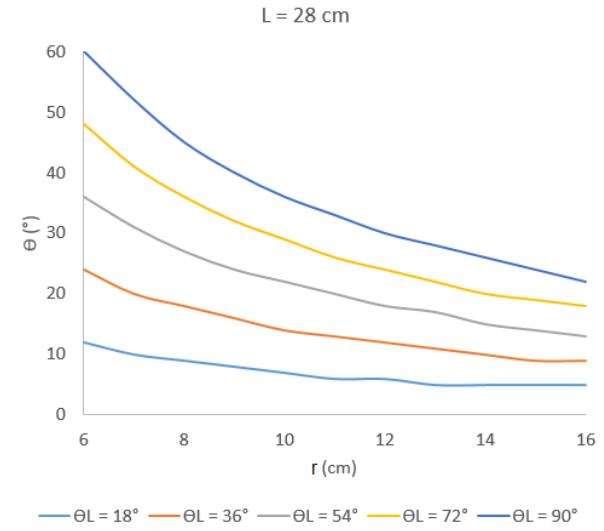


**Fig 3: Correlation of Distance between Sensor Array and Front Wheel with Angle of Line Turn for L=28cm.**

From the data tables as well as the graphs it is observed that, if distance between sensor array and front wheel/axel (r) increases with keeping distance between sensor array and rear wheel/axel (L) unchanged, the required angle to take necessary turns (Θ) decreases. And if distance between sensor array and rear wheel/axel (L) increases with keeping distance between sensor array and front wheel/axel (r) unchanged, the required angle (Θ) increases.

## 3. MATHEMATICAL MODEL

It is an interesting observation (from the tables) that if any arbitrary data is taken and the ratio of L and r is multiplied with the ratio of $\Theta_L$ and Θ, the result is always approximately equal to 7. For example, one arbitrary data from each table is taken,

For, L =32; r = 16; $\Theta_L$ = 18; Θ = 5.

$$\frac{L}{r} \times \frac{\theta_L}{\Theta} = \frac{32}{16} \times \frac{18}{5} = 7.2 \approx 7$$

For, L =32; r = 8; $\Theta_L$ = 54; Θ = 31.

$$\frac{L}{r} \times \frac{\theta_L}{\Theta} = \frac{32}{8} \times \frac{54}{31} = 6.97 \approx 7$$

For, L =28; r = 12; $\Theta_L$ = 90; Θ = 30.

$$\frac{L}{r} \times \frac{\theta_L}{\Theta} = \frac{28}{12} \times \frac{90}{30} = 7$$

For, L =28; r = 8; $\Theta_L$ = 90; Θ = 45.

$$\frac{L}{r} \times \frac{\theta_L}{\Theta} = \frac{28}{8} \times \frac{90}{45} = 7$$

So, if it is taken,

$$\frac{L}{r} \times \frac{\theta_L}{\Theta} = 7 \ (constant)$$

Then,

$$\frac{\theta_L}{\Theta} = \frac{7r}{L}$$

Or,

$$\frac{\Theta}{\theta_L} = \frac{L}{7r}$$

So,

$$\Theta = \frac{L}{7r} \times \theta_L$$

From this equation, it can be possible to find out the value of required turning angle (Θ) of any robot by providing the value of L and r manually and $\Theta_L$ from the sensor. Then the intelligence unit of the robot will be able to calculate the required turning angle by itself and then it will rotate itself at that particular angle [3,4].

At the time of turning, when $\Theta_L$ decreases gradually, Θ will also decrease gradually when using this equation. And when again $\Theta_L$ will come to 0° (that means, the robot has already completed rotating and now it needs to move forward), Θ will also come to 0° and the robot will again start moving forward [5,6].

## 4. ALGORITHM

In the second model, sensors have to be placed at the very front of the robot, which was not mandatory earlier. And also in previous algorithms, rear wheels were not meant for continuous rotation. That's why the robots built with the previous algorithm did not maintain a continuous speed. But, in this algorithm, rear wheels will be rotated at continuous rpm (rotation per minute). So, the speed of robot will remain same at all the times [7,8].

### 4.1 Steps

Step 1: Place the sensor at very front of robot.

Step 2: Detect line by sensor.

Step 3: Send sensor inputs to processor unit.

Step 4: Calculate turning angle by using equation.

Step 5: Rotate the robot at turning angle.

Step 6: Keep continuous speed.

### 4.2 Flow Chart

The flow chart of the algorithm is given below (figure 4) to help explain the process. Here, $\Theta_R$ is the angle of robot turn. This can be used to represent the automation units of robot.

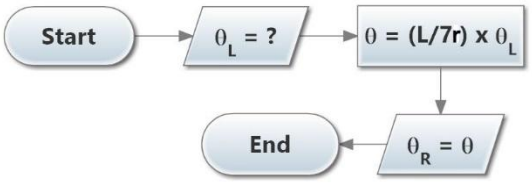


**Fig 4: Flow Chart of This Algorithm.**

At first the sensor unit will detect the line turn and will give necessary information to the processor unit of the robot. Then, the processor unit will calculate the value of Θ by the equation using the value of $\Theta_L$. The processor unit will accept the value of $\Theta_L$ from the information given by the sensor unit. Then the control unit will give necessary commands to the automation parts of the robot to rotate at Θ degree angle either right or left [9,10].

## 5. RESULTS

The time required to complete two paths where one is a straight line and the other is a turn was compared between two identical line follower robots (figure 5).

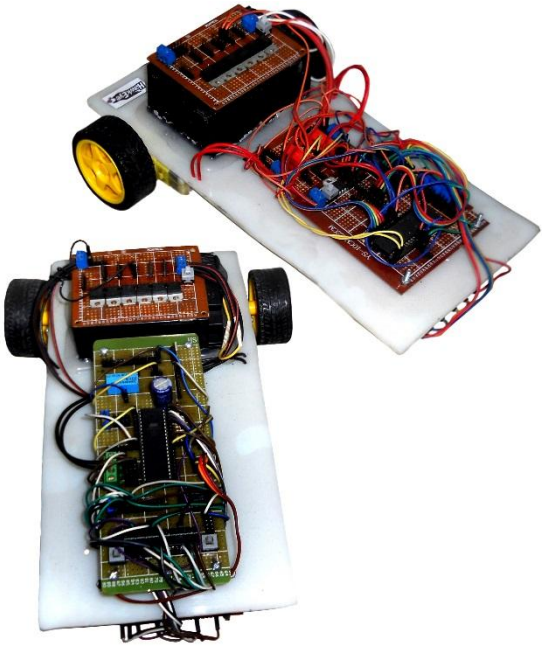

**Fig 5: Two Identical Robots Made With Different Algorithms.**

### 5.1 For a Straight Line

Here,

Length of the line = 190 cm.

Angle of line turn = 0°.

**Table 3. Forward Direction Data**

| Observation No. | New algorithm (sec) | Previous algorithm (sec) |
|---|---|---|
| 1 | 13.5 | 15.2 |
| 2 | 13.9 | 15.8 |
| 3 | 13.8 | 15.3 |
| 4 | 13.8 | 15.6 |
| 5 | 13.9 | 15.4 |
| 6 | 13.1 | 15.6 |
| Average | 13.7 | 15.5 |

**Table 4. Reverse Direction Data**

| Observation No. | New algorithm (sec) | Previous algorithm (sec) |
|---|---|---|
| 1 | 13.7 | 15.6 |
| 2 | 13.6 | 15.8 |
| 3 | 13.5 | 15.7 |
| 4 | 13.6 | 15.4 |
| 5 | 13.4 | 15.7 |
| 6 | 13.4 | 15.8 |
| Average | 13.5 | 15.7 |

Comparison of average time of both forward and reverse directions:

New algorithm = 13.6 seconds.

Old algorithm = 15.6 seconds.

So to cover a 190 cm straight line distance, the previous algorithm takes 2.0 seconds more than the new algorithm.

### 5.2 For a Turning Line

Here,

Length of the line = 625 cm.

Angle of line turn = 0˚ for 196 cm, 5˚ for 60 cm, 13˚ for 39 cm, 23˚ for 95 cm, 27˚ for 72 cm, 30˚ for 163 cm.

**Table 5. Forward Direction Data**

| Observation No. | New algorithm (sec) | Previous algorithm (sec) |
|---|---|---|
| 1 | 37.2 | 44.2 |
| 2 | 37.6 | 46.4 |
| 3 | 37.8 | 45.9 |
| 4 | 37.5 | 48.8 |
| 5 | 37.4 | 48.4 |
| 6 | 36.9 | 48.5 |
| Average | 37.6 | 47.0 |

**Table 6. Reverse Direction Data**

| Observation No. | New algorithm (sec) | Previous algorithm (sec) |
|---|---|---|
| 1 | 38.6 | 50.5 |
| 2 | 38.8 | 50.2 |
| 3 | 38.1 | 49.1 |
| 4 | 38.1 | 48.5 |

| | | |
|---|---|---|
| 5 | 38.0 | 48.9 |
| 6 | 38.4 | 50.6 |
| Average | 38.3 | 49.6 |

Comparison of average time of both forward and reverse directions:

New algorithm = 38.0 seconds.

Previous algorithm = 48.3 seconds.

So to complete a 625 cm turn, the previous algorithm takes 10.3 seconds more than new algorithm.

## 6. DISCUSSION

To measure the difference of times, the rpm (rotation per minute) of motors for both robots was kept very low but exactly same. Result shows that, a robot made with this algorithm is faster than the robot made with previous algorithms. The main reason behind this is the continuous speed of the new method. If the rotation of wheels is controlled to take necessary turns, the speed will decrease at the time of turning. In this algorithm the moving speed of the robot remains the same at all times. So it moves faster than a robot built with the previous method. Moreover, at the time of turning, the required turning angle decreases with time. Because, as the robot starts turning, the line angle starts decreasing. So the required turning angle is not always the same while taking a single turn. That's why to get a better performance, the rotation angle is needed to be changed every time while taking a single turn. As well as, it also saves time [11]. In this algorithm, at the time of taking a single turn, the required turning angle is changing all the time. But in the previous algorithm, there was no such feature. Thus making the second algorithm faster than the previous one. It has further been suggested that this algorithm is compatible with any control system [12]. This algorithm can be developed for optimizing turning velocity of any kind of part movement in any kind of robot. Which will increase the speed of any kind of movement in robotics.

## 7. ACKNOWLEDGMENT

Special thanks to Mr. Anadinath Mondol, Assistant professor, Mathematics department, Dhaka Residential Model College, for his motivational words.

## 8. REFERENCES

[1] Frazzoli, E., Lozano-Perez, T., Roy, N., Rus, D. (Eds.), 2013. Algorithmic Foundations of Robotics X. Springer Tracts in Advanced Robotics, Vol. 86, Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics.

[2] Kober, Jens, Peters, Jan, 2014. Learning Motor Skills. Springer Tracts in Advanced Robotics, Vol. 97, From Algorithms to Robot Experiments.

[3] T. Hongo, H. Arakawa, G. Sugimoto, K. Tange, Y. Yamamoto, 1987. An automatic guidance system of a self-controlled vehicle. IEEE Trans. Ind. Electronics, Vol. IE-34.

[4] Szewczyk, Roman, Zieliński, Cezary, Kaliczyńska, Małgorzata (Eds.), 2014. Recent Advances in Automation, Robotics and Measuring Techniques. Advances in Intelligent Systems and Computing, Vol. 267.

[5] Nonami, K., Kartidjo, M., Yoon, K.-J., Budiyono, A. (Eds.), 2013. Autonomous Control Systems and Vehicles. Intelligent Systems, Control and Automation: Science and Engineering, Vol. 65.

[6] N. Ayache, O. Faugeras, 1989. Maintaining representations of the environment of a mobile robot. IEEE Trans. Robotics Automat., vol.5.

[7] T. Hongo, H. Arakawa, G. Sugimoto, K. Tange, Y. Yamamoto, 1987. An automatic guidance system of a self-controlled vehicle. IEEE Trans. Ind. Electronics, Vol. IE-34.

[8] Becerra, Héctor. M, Sagüés, Carlos, 2014. Visual Control of Wheeled Mobile Robots. Springer Tracts in Advanced Robotics, Vol. 103, Unifying Vision and Control in Generic Approaches.

[9] Ani Hsieh, M., Chirikjian, Gregory (Eds.), 2014. Distributed Autonomous Robotic Systems. Springer Tracts in Advanced Robotics, Vol. 104, The 11th International Symposium.

[10] Ferrier, J.-L., Bernard, A., Gusikhin, O., Madani, K. (Eds.), 2014. Informatics in Control, Automation and Robotics. Lecture Notes in Electrical Engineering, Vol. 283, 9th International Conference, ICINCO 2012 Rome, Italy, July 28-31, 2012 Revised Selected Papers.

[11] B. Krogh, C. Thorpe, 1986. Integrated path planning and dynamic steering control for autonomous vehicles. IEEE Conf. Robotics Automat.

[12] Yoshida, Kazuya, Tadokoro, Satoshi (Eds.), 2014. Field and Service Robotics. Springer Tracts in Advanced Robotics, Vol. 92, Results of the 8th International Conference.