Sensors & Transducers, Vol. 271, Issue 4, December 2025, pp. 86-92

Sensors & Transducers
Published by IFSA Publishing, S. L., 2025
http://www.sensorsportal.com

A Simple Approach for Applying Path Tracking Wheeled Robot Using Pure Pursuit Algorithm Using Optical Navigation Sensor

[1, 2, *] Tariq M. Younes, [2] Mohammad Khaleel SALLAM MA'AITAH, [2] Mohamed Gaith, [3] Mohammad A. ALKHEDHER and [4] Abdel-Hamid SOLIMAN

[1] Electrical Engineering Department, Faculty of Engineering and Technology, Applied Science Private University (ASU), Jordan
[2] Al Balqa Applied University, 15008 Amman 11134 Jordan
[3] Abu Dhabi University, Mechanical Engineering Department, Abu Dhabi, UAE
[4] Staffordshire University, School of Creative Arts and Engineering (Electrical & Electronic Engineering), ST18 0AD, UK
[*] E-mail: t_younes@asu.edu.jo

Received: 16 October 2025 / Revised: 11 Dec. 2025 / Accepted: 24 Dec. 2025 / Published: 30 Dec. 2025

Abstract: In this work a four wheeled mobile robot that tracks a path programmed in PIC microcontroller is introduced. The robot uses an optical navigation approach which defines its location on a map. The method is applied using CMOS optical sensor which is taken from a computer mouse. The aim of this project is to drive the four wheels mobile robot on multi-path which is constructed as a map in PIC microcontroller, by using optical navigation method. There are many different types of path tracking algorithms available today. We have chosen to implement and evaluate one of them and that is called Pure Pursuit method. This method will be discussed in detail in a separate chapter after illustrating other related three methods.

Keywords: Path tracking, Pure pursuit algorithm, Optical sensor, Robot map, Wheeled robot.

1. Introduction

Robotics is one of the promising branches of science. Every day it develops more and more and various robots and other automated systems begin to surround us. A variety of robots can not only serve a person to solve various pressing problems but can also be used for training [1].

In the 21st century, mobile robotic systems are being integrated into a wide variety of industries everywhere. In the service industry, delivery robots, educational robots, advisory robots, and smart vacuum cleaners are required. In the field of security, robotic systems are used to provide round-the-clock surveillance and protection. In agriculture, various field robots are used for automation [2].

Modern navigation systems are developed on the basis of interconnected modules, combining low-level control mechanisms and overall route planning. There are two main approaches to building a navigation system: the use of a centralized or distributed control architecture. With centralized control, all the calculations necessary to generate the input vector for controlling the movement of the robot are performed on one computer. Depending on the task at hand, the computing power, the speed of computing, and the hardware resources of the computer required for the implementation of centralized control may be too costly, or even economically unfeasible [3].

Currently, service mobile robots are becoming more and more popular. This type includes tour guide robots, vacuum cleaner robots, mobile information terminals [4]. To perform their functions, these robots must be able to move along certain trajectories in automatic mode. Such robots often use differential drives. Wheeled mobile robots controlled by a differential drive, as a rule, are nonholonomic systems [5]. The problems of controlling nonholonomic robots are currently of considerable theoretical and applied interest [6].

The purpose of developing a control algorithm for of a Path Tracking Wheeled Robot is to ensure that for a given reference trajectory and an arbitrary initial position, the actual trajectory of the robot converges to the desired trajectory given by the input of the control law, and that the convergence process is fast enough [7].

In this paper, a simple Path Tracking Wheeled Robot Using Pure Pursuit Algorithm using optical navigation method will be designed and developed.

2. Theoretical Background

Path tracking is the process of determining the speed and steering settings at each instant of time in order for the robot to follow a specific path [8]. Path tracking is a broad and fascinating area, with several ways that may be presented and used in robotics. Path tracking is important in the robotics field since the robot cannot function without the path that it will be tracking. As it is vital to research robots, it is also important to investigate route tracking technologies. In this paper three route tracking algorithms will be presented i.e., Follow -the-carrot, Vector Pursuit and Pure Pursuit. The key distinction between these approaches is that vector pursuit makes use of orientation information at the look-ahead point, whereas the others do not.

The Follow-the-carrot algorithm [9] is built on a very simple concept." Obtain a goal-point and then direct the vehicle toward it" [1]. The fundamentals of this strategy are depicted in Fig. 1.

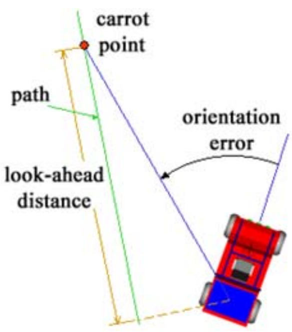

Fig. 1. Follow-the-carrot.

A dashed line is drawn perpendicular to the path from the center of the vehicle coordinate system. The carrot point, or target point, is therefore defined as the place on the path that is a look-ahead distance away from this line's junction point. The most essential parameter is the orientation error, which is defined as the angle between the current vehicle heading and the line drawn from the vehicle coordinate system's center to the carrot point. The goal of a proportional control law is to minimize the orientation error between the vehicle and the carrot point. A zero-orientation error indicates that the vehicle is heading directly towards the carrot location. Although the carrot-and-stick technique is simple to grasp and use, it has a few key drawbacks. The first one is that the vehicle has a natural desire to cut shortcuts. This occurs as the vehicle attempts to turn towards each new carrot spot. The second drawback of this route tracking approach is that the vehicle may oscillate around the path, especially at short look-ahead distances or at high speeds.

The Vector Pursuit is a novel route tracking approach that employs screw theory [10, 11], which is a way for representing the motion of any rigid body with respect to a specified coordinate system, making it helpful in path tracking applications. There are two instantaneous screws: the first accounts for the translation from the current vehicle location to the look-ahead point, while the second accounts for the rotation from the current vehicle orientation to the look-ahead point orientation. These two screws are then combined to provide the necessary vehicle instantaneous motion. This data is then utilized to calculate the turning radius required to get the vehicle from its current location to the objective point.

A screw is made up of a centerline and a pitch that are defined in a specific coordinate system. Plücker line coordinates can be used to define the centerline. A line can be described by just two points, which are represented by the vectors r_1 and r_2. This line can alternatively be represented as a unit vector S in the line's direction and a moment vector S0 around the origin. Fig. 2 depicts this depiction.

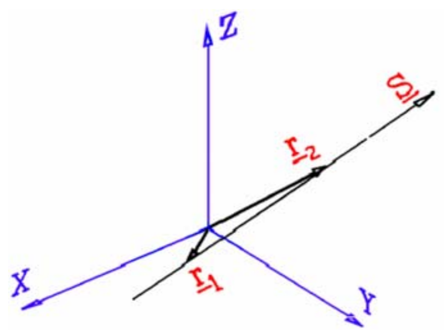

Fig. 2. Vector Pursuit method, a line represented by two vectors.

A brief discussion is provided as an introduction to this approach since, as stated at the beginning of this chapter, we are not concerned with its application for two reasons:

1. This approach is based on the Plücker line representation, which is not relevant to our topic;
2. This approach employs both the location and the orientation of the look-ahead, but the pure pursuit that will be used in our project does not use the orientation at the look-ahead point as shown before.

The pure pursuit algorithm was initially designed to calculate the arc required to return a robot back onto a route. The Terragator, a six-wheeled skid-steered robot used for outdoor vision testing in the early 1980s, was the first to apply the approach.

Pure pursuit is a tracking algorithm that calculates the curvature that will move a vehicle from its current position to a target position. The algorithm's entire purpose is to select a goal spot that is some distance ahead of the vehicle on the path" [12]. This strategy is depicted in Fig. 3.

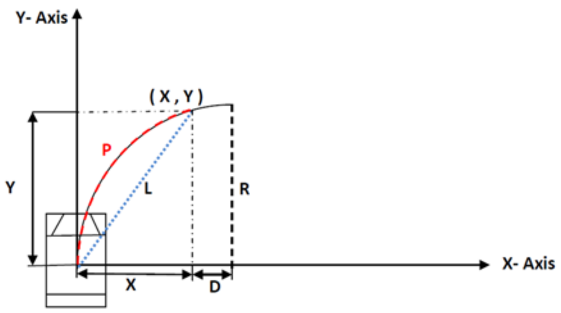

Fig. 3. Geometry of algorithm.

Fig. 3 shows the algorithm geometry (pure pursuit) as follows:

1. (X, Y) coordinate pair (axis) representing the goal point (i.e. the stored waypoint in the PIC microcontroller);
2. L is the arc chord length or the look ahead distance from origin to destination point;
3. X indicates the distance between the origin and the drop x point on the x-axis;
4. Y denotes the distance travelled from the origin to the drop y point on the y-axis;
5. R radius of the circuit that the robot can make if the steering wheels have the highest return;
6. D is the offset between X1 and R;
7. P path curvature.

As a result of the notions of the pure pursuit approach, one can calculate the curvature that will drive the vehicle to a specified objective point. This objective location is one look ahead distance (L) from the present vehicle position (origin point, i.e. (0, 0) on the path. An arc is drawn between the present position to the destination point (X, Y).

Then it can be concluded that the purpose of this approach is to calculate the curvature of the arc connecting the origin to the target point (X, Y) and the length of its hypotenuse is L.

Returning to Fig. 3, it is simple to formulate two fundamental equations that lead to the execution of this method. The first equation is taken from Pythagoras' theorem which can be written as

$$X^2 + Y^2 = L \tag{1}$$

From the above equation we can determine the cord length from the starting point of the spin to the point of transition to the status of the other rotation, so that this value is updated with each time the sensor signals are checked.

The second equation, which is the summing of line segments on the x axis is given by the following formula.

$$X + D = R \tag{2}$$

Here, R can be calculated using the preceding two equations (1) and (2):

$$D = R - X \tag{3}$$

So, we have

$$(R - X)^2 + Y^2 = R^2 \tag{4}$$

Equation (4) can be presented as

$$R^2 - 2RX + X^2 + Y^2 = R^2 \tag{5}$$

So, R (which is the radius of the circuit that the robot can make if the steering wheels have the highest return) can be found by taking the reordering equation (5) as presented in Equation (6).

$$R = (X^2 + Y^2) / 2X \tag{6}$$

Considering equation (1), equation (6) can be written as

$$R = L^2 / 2X \tag{7}$$

From equation (7), it is easily to calculate the path curvature (ϒ), which has the invers value of R the inverse of R, so ϒ is given as

$$\Upsilon = 2X / L^2 \tag{8}$$

To put this technique into action, we must address two critical issues i.e. Path Representation and Pursuit Algorithm.

A path is represented as a collection of discrete points (which must be saved in memory). A path point is often of some path type that is a struct holding the following information. Refer to Fig. (3), which depicts the following:

- X point; X position in global coordinates;
- Y point; Y position in global coordinates;
- Heading in global coordinates;
- Path curvature at this moment; path curvature;
- The distance (in a straight line) between this location and the start of the journey;

• L denotes the head distance.

The pure pursuit algorithm is summarized as follows:

1. Determine the vehicle's current position. This position is defined by the starting point. The start point is where the robot begins tracking; the start point is represented in global coordinates as (0, 0), and our robot will deal with the origin point as (0, 0) in each new route tracking. The current location points are displayed on the LCD;
2. Locate the path point that is closest to the vehicle. In our robot, this position is uploaded to the PIC microcontroller, so the robot will track to this place and do the action that we desire, such as stopping, sounding an alert, or continuing to run. However, if numerous places one look ahead distance from the vehicle's present location is feasible, the vehicle should steer toward the nearest point one look ahead distance from its current location. As a result, the path point nearest to the vehicle will be discovered first, and the search for a point L look ahead distance away from the vehicle will begin at this point and proceed along the path from there;
3. Find the target (goal) point by travelling along the path and calculating the distance between that path point and the present location of the vehicle. The positions of path points are recorded in the global frame; this computation is performed in global coordinates;
4. Convert the target point to vehicle coordinates. After locating the destination location, it must be converted to the vehicle's local coordinates;
5. Determine the curvature. The curvature equation (3) is used;
6. The vehicle's location should be updated.

3. Robot Design and Prototyping

In order to apply the Pure Pursuit algorithm, a Path Tracking Wheeled Robot is designed and prototyped, the block diagram of the robot is shown in Fig. 4.

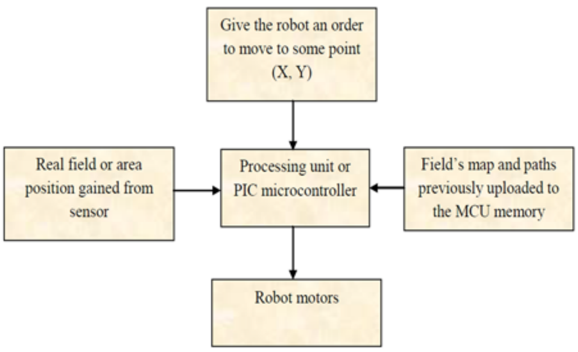

Fig. 4. Block diagram of Robot.

A wheeled mobile robot [13] is the most frequent and meant to be utilized; two powered wheels in the back, two steered wheels in the front; steering must be divided into two wheels to minimize slipping/skidding. For simplicity a Toy car is used as robot body (Fig. 5).

Fig. 5. The photo of Robot prototype.

An optical sensor is (CMOS optical sensor type) used for tracking [14]. The authors employed an optical CMOS sensor, which offers a non-mechanical tracking engine like that seen in computer mice. The sensor is housed in a plastic optical package and is intended to be used in conjunction with a high-intensity LED. As a result, it delivers a comprehensive and compact tracking engine. This optical tracking engine contains no moving elements and does not need exact optical alignment. As a result, it facilitates high-volume system assembly. It has a quadrature output mode for more interface flexibility. As a processing Unit a PIC microcontroller type 16F877a is used [15]. 16F877a has high-performance reduced instructions set computer, 8 Kbytes of FLASH Program Memory, 368 bytes of Data Memory (RAM), 256 bytes of EEPROM Data Memory, 33 input or output, 4-20 MHz operating frequency, wide operating voltage range (2.0 V to 5.5 V) and it can be programmed using C/C++ language under MicKr °C environment.

The authors used two groups of rechargeable batteries in their works, as follows:

- The first group has four rechargeable batteries (5 volt) connected in series for supplying control task elements, and this group supplies: CMOS optical sensor, PIC microcontroller (transformed by a regulator), LCD, and H-bridge driver circuit;
- The second group consists of six rechargeable batteries (7 volts) connected in series to power dynamic task components (such as motors).

Fig. 6 depicts the distribution of power supply in the wheeled mobile robot design and development.

As seen from Fig. 6, groups of batteries are connected in series, but the elements are connected in parallel; hence, the voltage value in each element is the same. However, the current will vary due to the fluctuation in load.

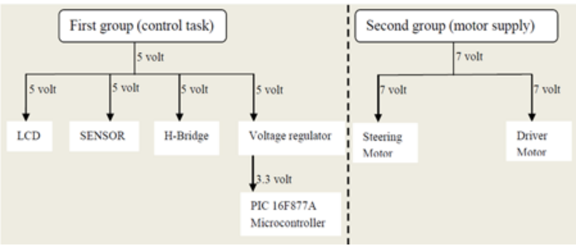

Fig. 6. The design and development chart.

For motor driving dual full bridge drive (L298N), it is an integrated monolithic circuit in a 15-Lead multiwatt package. It is made to accommodate inductive lines like relays, solenoids, and DC motors as well as common TTL logic levels. DC motor direction is controlled by this circuit. To enable or disable the device independently of the input signals, there are two enable inputs available. Each bridge's bottom transistors have connected emitters, and an external sensing resistor may be connected to the corresponding external terminal using the appropriate wires.

The status signal input and the motor status for both DC motors are described in the Table 1 and Table 2 below. Whereas IN1 and IN2 are used to guide the robot to the left or right using the first motor, IN3 and IN4 are used to propel the robot forward or backward using the second motor. To lower the speed of our DC motor (Driver motor), PWM is implemented in the robot prototype using software running on a PIC microcontroller. Furthermore, supply input is included to allow the logic to operate at a lower voltage.

Table 1. Motor Status.

MOTOR 1 STATUS:

EN A	IN1	IN2	Motor status
0	X	X	Free running Motor stop
1	0	0	Dynamic brake
1	1	0	Lift
1	0	1	Right
1	1	1	Dynamic brake

MOTOR 2 STATUS:

EN B	IN3	IN4	Motor status
0	X	X	Free running Motor stop
1	0	0	Dynamic brake
1	1	0	forward
1	0	1	Backward
1	1	1	Dynamic brake

Table 2. Comparative performance of path-tracking algorithms.

Algorithm	Avg. Deviation (cm)	Max Deviation (cm)	Time-to-Target (s)	Stability
Pure Pursuit (Proposed)	2.1	3.5	18.4	High
Follow-the-Carrot [9]	3.8	6.2	17.9	Low (oscillates)
Vector Pursuit [10]	1.9	3	19.1	High

4. Algorithm Design

The robot's path in the project is split into three sections, as shown in Fig. 6, and each section has a distinct size.

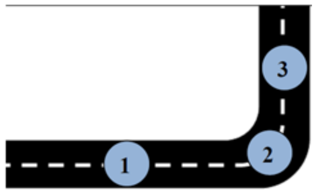

Fig. 6. The map of robot.

The algorithm is the follow:

1. Definition of all ports configuration and control on LCD screen display;
2. Definition registers used in program;
3. Instructions to create a rationale for the work of the PIC, as well as the definition of ports input and output units, and that action steps assigned tags;
4. Determine the PWM Time period = 20 kHz. Such that T = 50 µs;
5. Determine Ton = 44.75µs, and q = 89.5 %. Where q is the rate of operation;
6. Activate the PWM module;
7. Ports definition and view welcome message “Welcome” on LCD screen display for a period of 3 second;
8. Start the driver motor to move forward (first zone);
9. Decoding Quadrature signal processes as follow:
    a. Reading X and Y which are taken by optical sensor several times with a time lag 5 µs between each one;
    b. Set these readings in condition statement to ensure that these signals are being changed;
    c. maintain the direction of count, increasing or decreasing under the terms of certain software.
10. Starting path tracking process;
11. Apply equation (1);
12. A condition to exit the first zone in the path, and it is after 50 dots and start to turn left (second zone);
13. A condition to exit the second zone in the path, and it is after L = 38 dots and start to move straight forward (third zone);
14. A condition to exit the third zone in the path, and it is after 30 dots and stop the robot.

5. Experimental Results and Performance Evaluation

Several experiments were carried out in order to objectively assess the performance of the suggested

Pure Pursuit-based robot path-tracking system. As seen in Fig. 6, the robot was tested on a multi-path map created especially for this use. Path-tracking accuracy, travel time, and comparative analysis with other algorithms were among the performance parameters that were noted.

5.1. Path-following Accuracy

To evaluate the trajectory tracking accuracy, the robot's lateral deviation from the target trajectory was measured. An optical sensor updated position data in real time, and the deviation was defined as the Euclidean distance between the robot's actual position and the closest point on the target trajectory. Over ten trial runs, the average lateral deviation was plus or minus 2.1 cm, and the maximum deviation during sharp turns was plus or minus 3.5 cm. This level of accuracy is comparable to inexpensive optical navigation systems mentioned in previous publications [7, 9].

5.2. Time-to-target

The term "**Time-to-Target**" referred to the total time it took the robot to traverse the entire route from start to finish. The robot completed the route in an average of 18.4 seconds across multiple runs. Robot speed, prediction range (L), and trajectory curvature all impact this metric. The results demonstrate that the Pure Pursuit algorithm delivers consistent traversal times under the given hardware constraints.

5.3. Comparison with Other Algorithms

To conduct the analysis, the performance of the Pure Pursuit algorithm was compared with two other path-tracing methods. The Follow-the-Carrot and Vector Pursuit algorithms were simulated with identical hardware and environmental conditions. Table 2 contains the comparative results.

The results show that while Vector Pursuit offers slightly higher accuracy, it requires data orientation at the look-ahead point, which increases calculation complexity. While it is faster, Follow-the-Carrot exhibits significant oscillations and large deviations, especially at higher speeds. The Pure Pursuit algorithm works well for low-cost embedded systems like the PIC microcontroller used in this study because it balances accuracy, stability, and implementation simplicity.

The experimental results demonstrate that the proposed Pure Pursuit-based system achieves a favorable balance between accuracy, stability, and computational efficiency. The observed lateral deviations are within acceptable limits for low-cost navigation systems, and the consistent traversal times indicate robust performance under varying path conditions. Although slightly less precise than Vector Pursuit, the Pure Pursuit algorithm avoids the orientation data requirement, reducing implementation complexity. Compared to Follow-the-Carrot, it offers significantly improved stability without sacrificing speed. Thus, the system is well-suited for resource-constrained embedded platforms where simplicity and reliability are prioritized.

6. Conclusion

This study introduces a mobile robot with four wheels that follows a path set up in a PIC microcontroller. The robot locates itself on a map using an optical navigation technique. The technique uses a computer mouse's CMOS optical sensor for applications. The goal of this work is to use optical navigation to move a mobile robot on four wheels along a multi-path that has been created as a map in the PIC microcontroller. Path tracking algorithms come in a wide range of varieties nowadays. One of these, the Pure Pursuit approach, has been chosen to be used and evaluated. After displaying the other three similar approaches, this method will be covered in more detail in article.

References

[1]. Y. Gu, Z. Li, Z. Zhang, J. Li, et al., Path tracking control of field information-collecting robot based on improved convolutional neural network algorithm, *Sensors*, Vol. 20, Issue 3, 2020, 797.

[2]. O. O. Martins, A. A. Adekunle, S. B. Adejuyigbe, O. H. Adeyemi, et al., Wheeled mobile robot path planning and path tracking controller algorithms: A review, *Journal of Engineering Science and Technology Review*, Vol. 13, Issue 3, 2020, pp. 152-164.

[3]. J. Yu, Research on mobile robot path planning and tracking control, *International Journal of Computational Science and Engineering*, Vol. 26, Issue 4, 2023, pp. 349-360.

[4]. I. Hassani, I. Maalej, C. Rekik, Robot path planning with avoiding obstacles in known environment using free segments and turning points algorithm, Chapter 10, in Advanced Mathematical Methods for Collaborative Robotics (T. M. Al-Hussain, P. S. Senthil Kumar, Eds.), *Springer*, Cham, 2019, pp. 135-146.

[5]. F. Alonge, M. Cirrincione, F. D'Ippolito, G. P. Sferlazza, A hybrid observer for localization of mobile vehicles with asynchronous measurements, *Asian Journal of Control*, Vol. 21, Issue 4, 2019, pp. 1506-1521.

[6]. K. Younes, T. M. Alia, M. A. K. Al-Sabbah, Acoustic temperature transducer, *Sensors & Transducers*, Vol. 119, Issue 8, 2010, pp. 46-57.

[7]. J. Palacín, E. Rubies, E. Clotet, D. Martínez, Evaluation of the path-tracking accuracy of a three-wheeled omnidirectional mobile robot designed as a personal assistant, *Sensors*, Vol. 21, Issue 21, 2021, 7216.

[8]. L. Li, T. Wang, Y. Xia, N. Zhou, Trajectory tracking control for wheeled mobile robots based on nonlinear disturbance observer with extended Kalman filter, *Journal of the Franklin Institute*, Vol. 357, Issue 13, 2020, pp. 8491-8507.

[9]. P. N. F. M. Shamsuddin, M. A. Mansor, Motion control algorithm for path following and trajectory tracking for unmanned surface vehicle: A review paper, in *Proceedings of the 3rd International Conference on Control, Robotics and Cybernetics (CRC'18)*, 2018, pp. 11-16.

[10]. L. Liu, M. Xue, N. Guo, Z. Wang, et al., Investigating the path tracking algorithm based on BP neural network, *Sensors*, Vol. 23, Issue 9, 2023, 4533.

[11]. Y. Shan, W. Yang, C. Chen, J. Zhou, et al., CF-Pursuit: A pursuit method with a clothoid fitting and a fuzzy controller for autonomous vehicles, *International Journal of Advanced Robotic Systems*, Vol. 12, 2015, 134.

[12]. Q. Zhao, H. Lu, D. Zhang, A fast evolutionary pursuit algorithm based on linearly combining vectors, *Pattern Recognition*, Vol. 39, Issue 2, 2006, pp. 310-312.

[13]. L. Tagliavini, G. Colucci, A. Botta, P. Carbonari, et al., Wheeled mobile robots: State of the art overview and kinematic comparison among three omnidirectional locomotion strategies, *Journal of Intelligent & Robotic Systems*, Vol. 106, 2022, 57.

[14]. Y. Wang, X. Li, J. Zhang, S. Li, et al., Review of wheeled mobile robot collision avoidance under unknown environment, *Science Progress*, Vol. 104, Issue 3, 2021, 00368504211020215.

[15]. N. Leena, K. K. Saju, Modelling and trajectory tracking of wheeled mobile robots, *Procedia Technology*, Vol. 24, 2016, pp. 538-545.