VOL. 21, NO. 3, 2022, 69-75
https://elektrika.utm.my
ISSN 0128-4428

ELEKTRIKA
Journal of Electrical Engineering

Modelling, Simulation and Navigation of a Two-Wheel Mobile Robot using Pure Pursuit Controller

Amr A. A. Abdeltawab[*], **Sophan Wahyudi Nawawi, Noor Erlia Nasha Samsuria**, and **Navein A/L Sirkunan**

School of Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia.

[*]Corresponding author: amro2500@gmail.com

Abstract: Recently, the field of mobile robotics have witnessed an unprecedented expansion due to their vast applications that range from surveillance and construction to planetary exploration and emergency rescue operations. Their tasks became more and more complex that their autonomy became essential in the majority of the applications which means that devising controllers that suit different scenarios and situations is of importance. In this project, we aim to control the velocity of a modelled 2-wheel differential drive robot by using a PSO optimized PID controller. In addition to speed control, in order for a robot to move to specific waypoints by itself, a path planning algorithm has to be implemented. Pure pursuit, a simple yet efficient path tracking method for nonholonomic ground vehicles is utilized.

Keywords: Autonomous Navigation, Pure Pursuit, PID speed control

Article History: received 27 September 2022; accepted 15 December 2022; published 22 December 2022.

1. INTRODUCTION

First introduced in 1955, Automatic Guided Vehicles since that time has evolved to fulfill many different applications. Since that time, the guiding system which is the core part of any AGV has evolved along various stages of mechanical, optical, inductive, inertial, and laser guidance resulting in today's vision-based system. AGVs hence evolved to become Autonomous Mobile Robots or AMRs. AGVs can only move in specific path (typically defined by magnetic tape to be followed) while AMRs can freely navigate to in any point in a collision-free zone on a given map. This ability of AMRs caused an increase in production by reducing the time of inactivity [1]. AMRs also have the ability to adapt to new environments which by itself contributes to lowering the economic risk that the nature of AGVs pose [2].

When dealing with a simulated robot just as presented in this paper, accurate modelling is an important key in achieving realistic results. The robot to be simulated in this project is a 2-wheel non-holonomic differential drive robot with a caster wheel. A differential drive mobile robot is a common type that has an application area in the robotic research often [3]. Although the mathematical model for this robot can be simple, the existence of non-holonomic constraints presents a difficult challenge to overcome when it comes to designing the control system as typically, time-variant control systems are utilized. Based on the literature concerned with differential drive mobile robots, two methods of modelling are adopted, either Lagrangian approach or the Newton-Euler approach [4].

Along the years, many controllers have been developed for various dynamic systems. Up until now, Proportional Integral Derivative (PID) controller is being used in a multitude of industrial applications. PID controllers deal with most of the practical issues such as actuator saturation and integrator wind up. Their practicality and ease of design has made them quite popular that about 90% of industrial controllers are PID based according to [5]. As mentioned earlier, controlling non-holonomic systems typically requires time-variant controllers which means that in the case of a PID controller, there has to be a method that adjust the gains accordingly when needed. There are multiple algorithms to accomplish that, the most widely used of which is Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). PSO is famous for its simple and easy to implement concept which means that computationally, PSO is very efficient [6]. Compared to the rest of heuristic techniques, PSO has a more flexible with a balanced mechanism to enhance the local and the global explorations [7].

An autonomous vehicle's objective is to be able to self-drive itself by utilizing its sensory profile which may contain GPS, IMU, cameras, sensors etc. In order that accomplish that, the robot has to not only be able to detect its environment but also estimate its position by the help of these sensors. Once the robot knows both, it can be possible then to navigate itself with global and local planner which provide the robot with a path along which it needs to control itself to follow [8]. Path-following operations along with calculating minimal lateral distance as well as the heading between the vehicle and defined path is achieved by a path-tracking controller [2].

The path-tracking controller is responsible for generating steering and speed commands for the robot in order for it to follow the generated path by the planner based on the tracking error measured. For path tracking techniques for autonomous ground vehicles – which as mentioned earlier have to deal with non-holonomic constraints – are based on nonlinear control theory. Examples include predictive and adaptive control (Adaptive PID [9]), as well as fuzzy control. Despite the accuracy of these controllers, certain issues are inherent. PID controllers comes with issues in optimization of parameters and overshot in tracking. As for fuzzy controllers, they typically need more information. An alternative approach is to depend on the geometric considerations between the current position of the robot and the path it needs to follow [10]. Pure-pursuit algorithm is a very common – due its simplicity – and effective geometric method. Pure-pursuit algorithm calculates the robot's current position and a set point along the path. This point is chosen at a specified look-ahead distance, which is the chord length of this arc. Pure-pursuit algorithm tuning is fairly simple as it depends on the look-ahead distance which – in addition to its computational simplicity as well as the absence of any derivative terms – makes it a fairly simple and easy path-tracking algorithm.

2. MATHEMATICAL MODELLING

Modelling is the process of expressing real-life problems in mathematical terms. It is an important stage as it enables controller design and makes it possible to tune and test various controllers. For robotics, dynamic and kinematic models need to be implemented to express the movement of a robot. Derivation of the dynamic equations is usually achieved by adopting Lagrangian approach or the Newton-Euler approach [11]. The following subsections, describe the derivation of a two-wheel differential drive robot's mathematical model using Newton-Euler approach. The equations are derived and converted into the s-domain as they are to be used for modelling on Simulink.

2.1 Robot Dynamic Model

The first step in Newton-Euler dynamic modeling is to draw the free body diagram of the system which helps in analyze the forces acting on it. This is a very crucial step as the accuracy of the model derived depends upon whether all forces are considered or not.

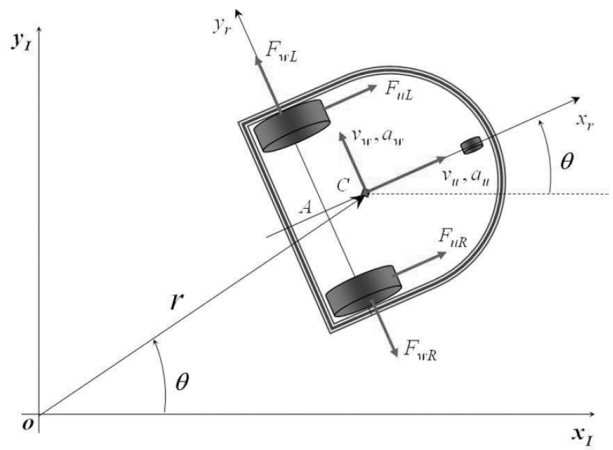

Figure 1. Robot FBD [4].

The free body diagram of the two-wheel differential drive robot (with a caster wheel) is shown in Figure 1. Using the robot local frame $\{x_r, y_r\}$, the following notations are introduced. (v_u, v_w) represents the velocity of the vehicle center of mass C in the local frame; v_u is the longitudinal velocity and v_w is the lateral velocity; (a_u, a_w) represent the acceleration of the vehicle's center of mass C; $\left(F_{u_g}, F_{u_L}\right)$ are the longitudinal forces exerted on the vehicle by the left and right wheels; $\left(F_{w_g}, F_{w_L}\right)$ are the lateral forces exerted on the vehicle by the left and right wheels; θ is the orientation of the robot; ω is the angular velocity; and m is the mass of the robot [4].

By assuming the robot to be a rigid body, we can derive its radial and tangential velocity as well as acceleration from the differential equations created by correlating the polar position vector with the robot's inertial frame.

$$v_u = \dot{r} \tag{1}$$

$$v_w = r\dot{\theta} \tag{2}$$

$$a_u = \ddot{r} - r\dot{\theta}^2 \tag{3}$$

$$a_w = 2\dot{r}\dot{\theta} + r\,\ddot{\theta} \tag{4}$$

Then by using Newton's second law of motion in the robot's frame of reference, the relationship between the forces and torques are derived. The robot exhibits two types of motion, which are translations in radial and tangential directions, and rotation around its Z-axis at C. Assuming M is the total mass of the robot, and J as the moment of inertial with respect to C. By also assuming the absence of any slipping as any sliding, the dynamic equations for translational velocity as well as rotational acceleration can be represented by

$$\dot{v}_u = d\dot{\theta}^2 + \frac{1}{M}(F_{uL} + F_{uR}) \tag{5}$$

$$\ddot{\theta} = \frac{L}{Md^2+J}(F_{uL} + F_{uR}) - \frac{Mdv_u}{Md^2+J}\dot{\theta} \tag{6}$$

By using the Lagrangian approach, actuator torques can be considered as well in (5) and (6).

$$(Md^2 + J)\,\ddot{\theta} + Mdv_u\dot{\theta} = \frac{L}{R}\left(\tau_r - \tau_l\right) \tag{7}$$

$$M\dot{v}_u - Md\dot{\theta}^2 = \frac{1}{R}\left(\tau_r + \tau_l\right) \tag{8}$$

By rearranging the above equations, the angular acceleration and the translational velocity can then be expressed by the following two equations.

$$\ddot{\theta} = \frac{L(\tau_r - \tau_l)}{R\,(Md^2+J)} - \frac{MdV\dot{\theta}}{Md^2+J} \tag{9}$$

$$\dot{v}_u = d\dot{\theta}^2 + \frac{1}{MR}\left(\tau_r + \tau_l\right) \tag{10}$$

2.2 Actuator Model

Servo DC motors are the typical actuators for a ground differential drive mobile robot. The armature voltage v_a is used as the input that controls the motor while keeping other conditions constant. For the armature circuit the following equations are utilized.

$$v_a = R_a i_a + L_a \frac{di_a}{dt} + e_a \quad (11)$$

$$e_a = K_b \omega_m \quad (12)$$

$$\tau_m = K_t i_a \quad (13)$$

$$\tau = N\tau_m \quad (14)$$

Here, i_a is the armature current, (R_a, L_a) is the resistance and inductance of the armature winding respectively, e_a is the back emf, ω_m is the rotor angular speed , τ_m is the motor torque, $(K_t,\ K_b)$ are the torque constant and back emf constant respectively, N is the gear ratio, and is τ the output torque applied to the wheel. By using LaPlace transform on (1), we get

$$V_a(s) = s.\ L_a I_a(s) + R_a I_a(s) + E_a(s) \quad (15)$$

Then by substituting (12) into (15) we get

$$V_a = s.\ L_a I_a + R_a I_a + K_b \omega_m \quad (16)$$

By rearranging (16), and substituting (13) into (14), yields the following equations.

$$I_a = \frac{v_a - K_b \omega_m}{R_a + s.\ L_a} \quad (18)$$

$$\tau = N K_t i_a \quad (19)$$

2.3 Wheel Kinematics

The linear velocity of the differential drive mobile robot in the Robot Frame is the average of the linear velocities of the two wheels represented by the equation below.

$$V = \frac{V_r + V_l}{2} = \frac{R\dot{\omega}_r + R\dot{\omega}_l}{2} \quad (20)$$

Which means that the angular velocity of the robots is

$$\omega = \frac{V_r - V_l}{2L} \quad (21)$$

Thus, we can drive the angular acceleration of each wheel by substituting (21) into (20), which will yield the equations below.

$$\dot{\omega}_r = \frac{V + \omega L}{R} \quad (22)$$

$$\dot{\omega}_l = \frac{V - \omega L}{R} \quad (23)$$

3. SPEED CONTROLLER DESIGN

We would typically want the robot to maintain a specific linear velocity (given by the commands from the path-tracking algorithm) by controlling the voltage input. To accomplish this a controller is needed be designed. PID is selected for its simplicity and ease of implementation yet, an efficient method of optimization for adjusting the gains has to be adopted. Hence, Particle Swarm Optimization is selected in order to accomplish that.

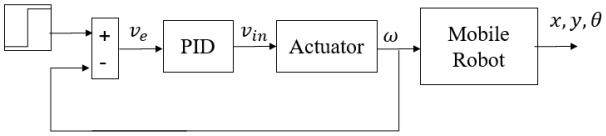

Figure 2. PID controller block diagram.

Particle Swarm Optimization (PSO) is a technique used to explore the search space of a given problem to find the settings or parameters required to maximize a particular objective – hence the optimization part. PSO is sometimes considered as an evolutionary computation technique. The method has been found to be robust in solving problems featuring nonlinearity and non-differentiability, multiple optima, and high dimensionality through adaptation. In this paper the PSO algorithm is used to find the optimal parameters for the PID controller used for the control of the velocity of the two-wheel differential drive mobile robot. PSO is used to optimize the three gains of the PID controller Kp, Ki, and Kd which means that the search space of three dimensions and thus the particles have to fly in 3D space.

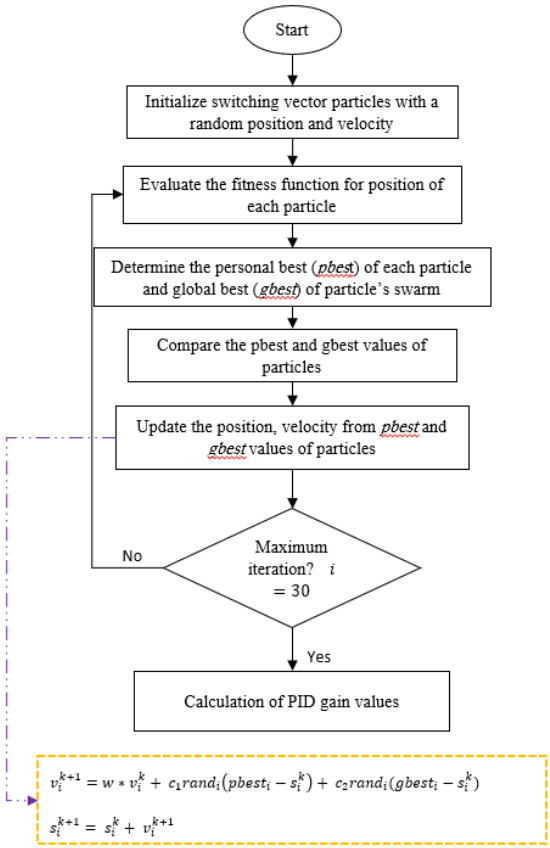

Figure 3. PSO Algorithm Block Diagram

In PSO, for a d-variables optimization problem, a flock of particles are put into the d-dimensional search space with randomly chosen velocities and positions knowing their best values so far (P_{best}) and their position in the d-dimensional space. The velocity of each particle, adjusted according to its own flying experience and the other particle's flying experience. The index of best particle among all of the particles in the group is g_{best} [12]. The modified velocity and position of each particle can be calculated using the current velocity and the distance from P_{best} to g_{best}. By testing and using different values for population size and number of iterations, the values shown in Table I below are selected. The resulted PID gains are shown in Table II. It is worth noting that Mean Square Error evaluation criterion was selected.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(e - \hat{e})^2 \qquad (24)$$

Table I. Values selected for the PSO

Var.	Description	Value
N	No. of Particles	35
t_{max}	Max. No. of Iterations	20
w_{imax}	Max. inertia weight	0.9
wi_{min}	Min. inertia weight	0.4
c_1 ; c_2	Acceleration Coeff.	1.2
J	Fitness function	MSE

Table II. PID gains calculated by PSO

PID Gain	Value
Kp	9.9918
Ki	1.0192
Kd	0.0427

4. SIMULINK MODELLING & PURE-PURSUIT CONTROLLER IMPELEMENTATION

In this section, modelling is carried out using MATLAB and Simulink software (2020a). The implementation of the robot model is carried out based on the previously derived dynamic equations. The implementation of the pure pursuit controller is also carried out using Simulink. Simulink provided several powerful kits that are used in this project. Among which many helped with the results and analysis discussed in section 5.

4.1 Robot Kinematic Model

The first step was to model the robot on Simulink using the dynamic equations derived earlier by using a block diagram. The model should incorporate the wheel kinematics, the actuator model, as well as the robot entire kinematics. The robot Simulink block diagram is shown in Figure 4 below.

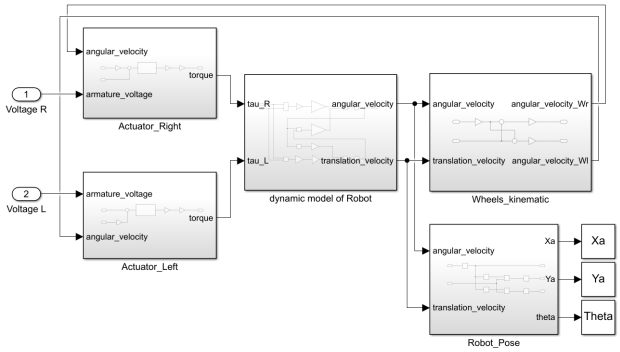

Figure 4. Robot Kinematic Simulink Model

The robot model shown above had many constants that needed to be assumed in order carry out simulation. The assumed parameters (taken from [3]) are shown in Table 3. The model is verified by inputting a step input as voltage to one of the wheels and graphing the pose on xy-plane. The robot showed the expected rotation.

Table III. Robot Parameters

Parameter	Value
La	0.088
Ra	1.01
Kb	12.939
Kt	12.939
N	53
J	0.0732
M	27
d	0.05
R	0.0975
L	0.0164

4.2 Pure Pursuit Controller

Next was implementing the PID and the Pure pursuit controllers. For the PID controller, the gains by PSO were directly used. As for the pure pursuit controller, the Simulink block "Pure Pursuit Controller" as used. It takes two inputs, the post of the robot as well as the waypoints to which the robot has to navigate. Bear in mind that mapping as well as the sensory profile of the robot were not incorporated in this project, so we assumed a static map and we assume that the robot has accurate estimation about its current position. The pure pursuit controller was then to track the path to the waypoints by controlling the robot linear and angular velocities. The complete model is shown in Figure 5. The parameters needed for the pure pursuit controller can be found in Table 4.

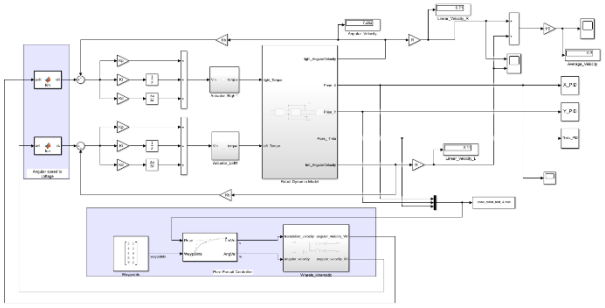

Figure 5. Entire Robot Simulink Model

Table IV. Values selected for the PSO

Var.	Description	Value
Desired Linear Velocity, (ms^{-1})	Desired Linear Velocity for the robot to move to a designated waypoint	0.3
Desired Angular Velocity, $(rads^{-1})$	Desired Angular Velocity for the robot to move to a designated waypoint	2.5
R, (m)	Radius of the wheel	0.0975
L, (m)	Distance between the midpoint and centers of two wheels	0.164
Lookahead distance, (m)	Distance of how far along the path the robot should look from current location to compute angular velocity	0.1

5. SIMULATION AND RESULTS

Although many DC servo motors (which are used for the majority of ground mobile robots) has their embedded speed controller, the PID controller designed seemed to completely smoothen out fluctuations noticed in the actuators' behavior. Additionally, the response time drastically improved which can be of a great importance especially for advanced navigation tasks where the robot is assumed to stream its pose in real-time. Figure 6 shows the difference between the output in actuator velocity for a 12V input for both wheels with and without the controller.

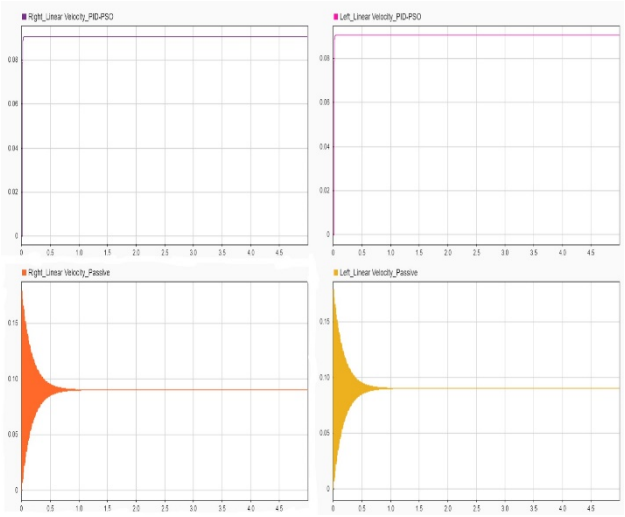

Figure 6. Linear velocity for both wheels with and without the controller

The same kind of improvement (shown in Figure 7) can be seen when the waypoints were entered into the system and the robot started to follow them accordingly. No noisy signal observed; a good control of DC motor is maintained and secured

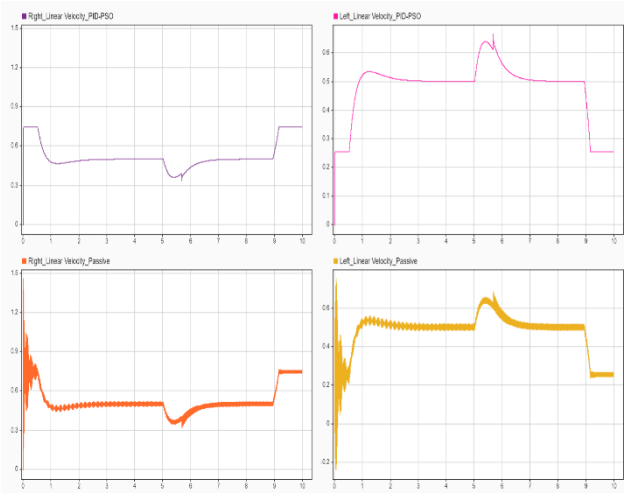

Figure 7. Controller effect on linear velocity while navigating.

For the pure-pursuit simulation, the "robot visualizer" tool in Simulink was utilized. As shown in Figure 8, the robot visualizer takes in the current po and the desired waypoints and drive the robot within an environment of design as desired.

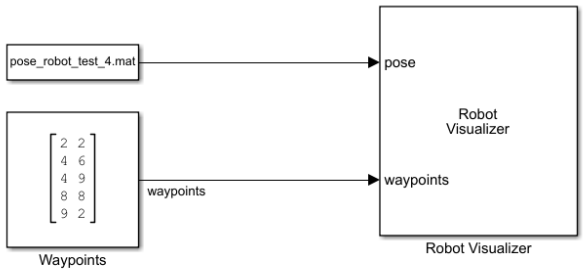

Figure 8. Robot Visualizer block diagram

The flowchart shown in Figure 9 resembles the entire navigation process for the robot. First, the robot has its initial post, upon which it will use the pure pursuit to set out the commands for the actuator to move towards the first waypoint in the matrix provided. Then, it will update the pose accordingly until all waypoints are achieved.

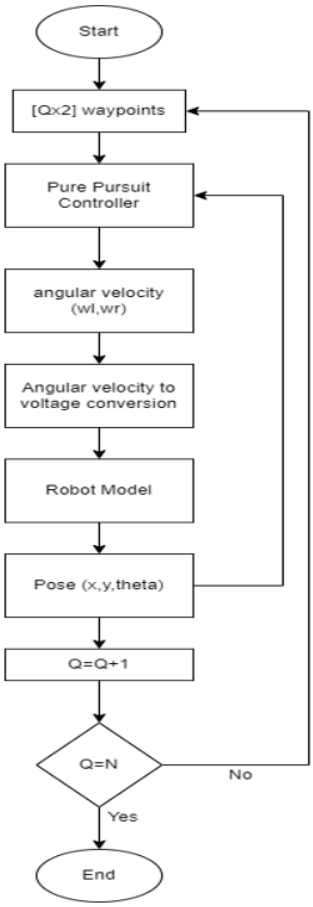

Figure 9. Flowchart of the mobile robot navigation

The robot visualizer (shown in Figure 10) helps visualize the robot as well as its look-ahead / foresight distance which is crucial to the efficiency of the trajectory tracking. If it is large the robot will move along a small arc and the path will not have large oscillation, which although might sound great, will significantly lower the tracking accuracy. Similarly a small foresight distance will cause the robot to be too sensitive to the curvature of the path and the robot will experience sudden and acute rotation [13].

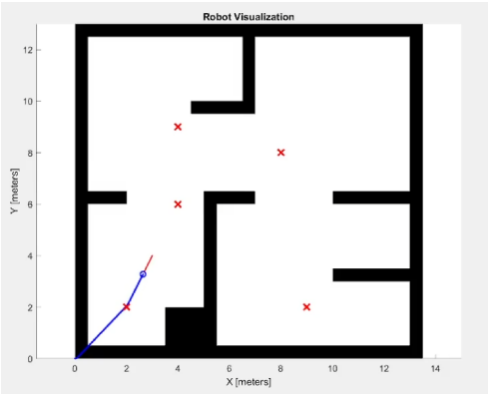

Figure 10. Robot visualizer output

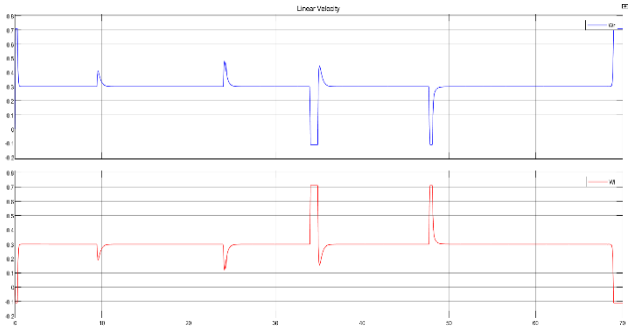

Figure 11. Linear Velocity of both wheels along the test

Studies as early as 2001 [14] had suggested that since too large or too small foresight distance will affect the accuracy of the robot's trajectory tracking, the actual pose and speed of the robot had to be taken into account instead of simply considering a single factor. In [13], and improved pure pursuit trajectory tracking approach was introduced by using fuzzy controllers to determine the foresight dynamically.

6. CONCLUSION

Mobile robotics have a growing attention due to its ability to revolutionize many applications, ranging from surveillance and construction to planetary exploration and emergency rescue operations. As autonomy became essential in the majority of the applications, designing different control systems and improving them became vital to many of the robot's tasks' efficiency. In this project A dynamic model of a two-wheel robot has been implemented and validated. A PSO optimized PID controller has been implemented for the robot model after the tuning process was carried out it accordingly. Point to point navigation has been done using a pure pursuit controller to move the robot autonomously to the desired waypoints. The speed control implemented using the PID controller proved to drastically improve the linear velocity output as a smooth and fast response is achieved. An improvement for this project can include the simulation of multiple sensors and fusing them accordingly in order achieve the most accurate position estimation possible. Pure pursuit path tracking although simple in application, can be improved and tuned by adding another layer of optimizers concerned with the lookahead or the foresight distance.

REFERENCES

[1] G. Fragapane, R. de Koster, F. Sgarbossa, and J. O. Strandhagen, "Planning and control of autonomous mobile robots for intralogistics: Literature review and research agenda," *Eur. J. Oper. Res.*, vol. 294, no. 2, pp. 405–426, 2021, doi: 10.1016/j.ejor.2021.01.019.

[2] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*, Second. MIT Press, 2011.

[3] M. KALYONCU and F. DEMİRBAŞ, "Differential Drive Mobile Robot Trajectory Tracking With Using Pid and Kinematic Based Backstepping Controller," *Selcuk Univ. J. Eng. ,Science Technol.*, vol. 5, no. 1, pp. 1–15, 2017, doi: 10.15317/scitech.2017.65.

[4] R. D. Ahmad Abu Hatab, "Dynamic Modelling of

Differential-Drive Mobile Robots using Lagrange and Newton-Euler Methodologies: A Unified Framework," *Adv. Robot. Autom.*, vol. 02, no. 02, 2013, doi: 10.4172/2168-9695.1000107.

[5] L. A. Aguirre, "The historical development of texts for teaching classical control of linear systems," *Annu. Rev. Control*, vol. 39, no. October, pp. 1–11, 2015, doi: 10.1016/j.arcontrol.2015.03.002.

[6] M. A. Abido, "Optimal design of power-system stabilizers using particle swarm optimization," *IEEE Trans. Energy Convers.*, vol. 17, no. 3, pp. 406–413, 2002, doi: 10.1109/TEC.2002.801992.

[7] T. Y.Abdalla and A. A. A, "PSO-based Optimum Design of PID Controller for Mobile Robot Trajectory Tracking," *Int. J. Comput. Appl.*, vol. 47, no. 23, pp. 30–35, 2012, doi: 10.5120/7497-0601.

[8] N. Buniyamin, W. W. Ngah, W. a J. Wan Ngah, N. Sariff, and Z. Mohamad, "A simple local path planning algorithm for autonomous mobile robots," *Int. J. Syst. Appl. Eng. Dev.*, vol. 5, no. 2, pp. 151–159, 2011, [Online]. Available: http://www.naun.org/main/UPress/saed/19-671.pdf%5Cnfiles/1177/19-671.pdf.

[9] P. Zhao, J. Chen, Y. Song, X. Tao, T. Xu, and T. Mei, "Design of a control system for an autonomous vehicle based on adaptive-PID," *Int. J. Adv. Robot. Syst.*, vol. 9, pp. 1–11, 2012, doi: 10.5772/51314.

[10] M. Samuel, M. Hussein, and M. Binti, "A Review of some Pure-Pursuit based Path Tracking Techniques for Control of Autonomous Vehicle," *Int. J. Comput. Appl.*, vol. 135, no. 1, pp. 35–38, 2016, doi: 10.5120/ijca2016908314.

[11] A. M. Bloch, *Nonholonomic Mechanics and Control*, vol. 24. New York, NY: Springer New York, 2003.

[12] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, 1995, vol. 4, pp. 1942–1948, doi: 10.1109/ICNN.1995.488968.

[13] H. Wang, X. Chen, Y. Chen, B. Li, and Z. Miao, "Trajectory tracking and speed control of cleaning vehicle based on improved pure pursuit algorithm," *Chinese Control Conf. CCC*, vol. 2019-July, pp. 4348–4353, 2019, doi: 10.23919/ChiCC.2019.8865255.

[14] C. S. Tseng, B. Sen Chen, and H. J. Uang, "Fuzzy tracking control design for nonlinear dynamic systems via T-S fuzzy model," *IEEE Trans. Fuzzy Syst.*, vol. 9, no. 3, pp. 381–392, 2001, doi: 10.1109/91.928735.