Jurnal Teknologi

Full Paper

VELOCITY CONTROL OF A UNICYCLE TYPE OF MOBILE ROBOT USING OPTIMAL PID CONTROLLER

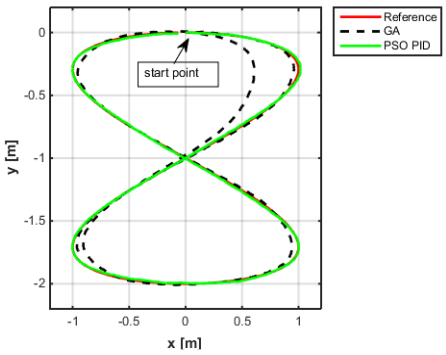
Norhayati A. Majid, Z. Mohamed*, Mohd Ariffanan Mohd Basri

Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia

Received
15 December 2015
Received in revised form
30 March 2016
Accepted
30 May 2016

*Corresponding author
zahar@fke.utm.my

Graphical abstract



Abstract

A unicycle model of control a mobile robot is a simplified modeling approach modified from the differential drive mobile robots. Instead of controlling the right speed, V_R and the left speed, V_L of the drive systems, the unicycle model is using u and ω as the controller parameters. Tracking is much easier in this model. In this paper, the dynamic of the robot parameter is controlled using two blocks of Proportional-Integral-Derivative (PID) controllers. The gains of the PID are firstly determined using particle swarm optimization (PSO) in offline mode. After the optimal gain is determined, the tracking of the robot's trajectory is performed online with optimal PID controller. The achieved results of the proposed scheme are compared with those of dynamic model optimized with genetic algorithm (GA) and manually tuned PID controller gains. In the algorithm, the control parameters are computed by minimizing the fitness function defined by using the integral absolute error (IAE) performance index. The simulation results obtained reveal advantages of the proposed PSO-PID dynamic controller for trajectory tracking of a unicycle type of mobile robot. A MATLAB-Simulink program is used to simulate the designed system and the results are graphically plotted. In addition, numerical simulations using 8-shape as a reference trajectory with several numbers of iterations are reported to show the validity of the proposed scheme.

Keywords: Unicycle type of mobile robot, tuning dynamic gains, PSO-PID controller

1.0 INTRODUCTION

A simplified control model called unicycle type of mobile robots has been used in many robotics applications. There are many proposed control laws in the mobile robot literature that can be used to control a unicycle type of mobile robots. One of the approaches used to design the basic control laws is based on kinematic and dynamic [1]. Most of the latest design has considered the dynamic part to improve the robot performance in the real condition. For example, Martin *et al.* [2] has considered using adaptive dynamic controller with feedback linearization. This control approach has good performance, however, it is dependent on accurate model parameters, i.e. when model parameters are unknown, adaptive control for adjusting these parameters is required. Some authors used backstepping techniques to design the adaptive control law [3]. Although the backstepping method can provide a systematic process, the controller parameters are obtained arbitrarily. On the other hand, other authors have used a feedback linearization approach and Lyapunov theories [4]. The saturation feedback controller for unicycle mobile robot proposed by Lee *et al.* [5], however, it is only applicable for a single controller. Thus, a kinematic controller, which controls the trajectory of the robot at an upper level, and an adaptive dynamic controller, which controls the velocities of the robot at a lower

level is necessary. The previous work done by Martin *et al.* [6] described the controller gains of the unicycle dynamic model adjusted using genetic algorithms. However, GA performance indicates slow convergence.

In this paper, an extension of the above work on velocity controller by using PID is developed. PID gains such as K_p K_i and K_d are determined by the controllers, however, the three adjustable controller parameters should be tuned appropriately. The existence of conventional parameter tuning techniques such as Ziegler-Nichols (ZN) and various artificial intelligence approaches such as Genetic Algorithm (GA), Artificial Neural Network (ANN), Ant Colony Optimization (ACO) and Differential Evolutions (DE) could help to tune the PID for optimal gain combinations for a better response of the system.

More recently, an optimization technique, Particle Swarm Optimization (PSO) has been introduced. A comparative study between PSO and GA has been carried out by Hassan *et al.* [7] and it was found that PSO gives better performance as it has good global searching ability and easier to be implemented than GA. However, as PSO is a new evolutionary computation technique, there are not many research have been done yet in implementing PSO technique in dynamic model of unicycle mobile robot. A good example of using PSO-PID is reported by Hashim *et al.* [8] utilized on precise positioning system in micro-EDM specific for biomedical application where the PI and PID controllers are used to control the DC motor precisely.

This paper will be focused on employing optimal PID controller into the dynamic model of the unicycle-like mobile robot that will be explained in Section 2. The method of tuning PID controller is discussed in Section 3. Next, Section 4 presents the simulation setup via MATLAB-SIMULINK. Section 5 highlights the results, of comparative studies between the proposed controller and the controller designed by the Martin *et al.* [6]. Lastly, the Section 6 concludes the results and the findings.

2.0 ROBOT MODEL

A unicycle mobile robot considered in this paper is a class of nonholonomic mobile robots. Therefore it also has nonholonomic constrains due to the wheel limitations. In this section, the dynamic model of the unicycle-like mobile robot proposed by Martin *et al.* [1] is reviewed. Figure 1 depicts the mobile robot, its parameters and variables of interest. $\boldsymbol{u}$ and $\boldsymbol{\omega}$ are the linear and angular velocities of the robot, respectively, $\boldsymbol{G}$ is the center of mass of the robot, $\boldsymbol{C}$ is the position of the castor wheel, $\boldsymbol{E}$ is the location of a tool onboard the robot, $\boldsymbol{h}$ is the point of interest with coordinates x and y in the XY plane, $\boldsymbol{\psi}$ is the robot orientation, and a is the distance between the point of interest and the central point of the virtual axis linking the traction wheels (point B).

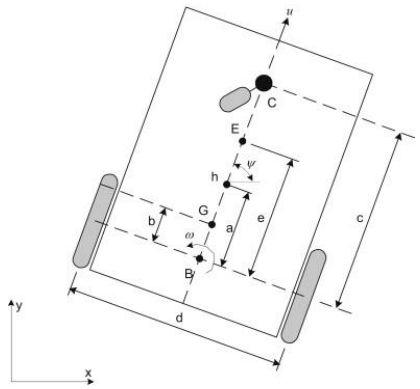


Figure 1 The differential drive (unicycle-like) mobile robot [2]

2.1 Kinematic Modeling

The design of the kinematic controller is based on the kinematic model of the robot, assuming that the disturbance term in Eq. (1) is a zero vector. From Martin *et al.* [2], the robot's kinematic model is given by

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\psi}\end{bmatrix}=\begin{bmatrix}cos\ \psi & -\mathrm{a}\sin\psi\\ \sin\psi & \mathrm{a}\cos\psi\\ 0 & 1\end{bmatrix}\begin{bmatrix}u\\ \omega\end{bmatrix}, \tag{1}$$

whose output are the coordinates of the point of interest, thus meaning $\mathbf{h}=[x \quad y]^T$. Hence

$$\dot{\mathbf{h}}=\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}=\begin{bmatrix}\cos\psi & -a\sin\psi\\ \sin\psi & a\cos\psi\end{bmatrix}\begin{bmatrix}u\\ \omega\end{bmatrix}=A\begin{bmatrix}u\\ \omega\end{bmatrix} \tag{2}$$

with $A=\begin{bmatrix}\cos\psi & -a\sin\psi\\ \sin\psi & a\cos\psi\end{bmatrix}$

whose inverse is

$$A^{-1}=\begin{bmatrix}\cos\psi & \sin\psi\\ -\frac{1}{a}\sin\psi & \frac{1}{a}\cos\psi\end{bmatrix}$$

Therefore, the inverse kinematics is given by

$$\begin{bmatrix}u\\ \omega\end{bmatrix}=\begin{bmatrix}\cos\psi & \sin\psi\\ -\frac{1}{a}\sin\psi & \frac{1}{a}\cos\psi\end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} \tag{3}$$

and the kinematic control law proposed to be applied to the robot is given by

$$\begin{bmatrix}u_{ref}^{c}\\ \omega_{ref}^{c}\end{bmatrix}=\begin{bmatrix}\cos\psi & \sin\psi\\ -\frac{1}{a}\sin\psi & \frac{1}{a}\cos\psi\end{bmatrix}\begin{bmatrix}\dot{x_d}+I_x\ tanh\left(\frac{k_x}{I_x}\tilde{x}\right)\\ \dot{y_d}+I_y\ tanh\left(\frac{k_y}{I_y}\tilde{y}\right)\end{bmatrix} \tag{4}$$

Instantaneous distance error can be calculated as

$$e(t)=\sqrt{\tilde{x}^2+\tilde{y}^2} \tag{5}$$

where, $\tilde{x}=x_d-x$, and $\tilde{y}=y_d-y$ are the current position errors in the axes X and Y, respectively, $k_x>0$ and $k_y>0$ are the gains of the controller, $I_x \in \Re$, and

$I_y \in \Re$ are saturation constants, and (x, y) and (x_d, y_d) are the current and the desired coordinates of the point of interest, respectively. The objective of such a controller is to generate the references of linear and angular velocities for the dynamic controller, as shown in Figure 2.

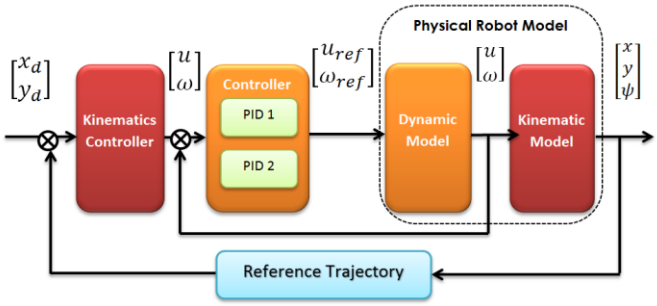


Figure 2 Overall PID control system

2.2 Dynamic Modeling

The complete mathematical model adopted from De La Cruz and Carelli [9], is written as

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\psi}\\ \dot{u}\\ \dot{\omega}\end{bmatrix} = \begin{bmatrix} u\cos\psi - a\omega\sin\psi \\ u\sin\psi + a\omega\cos\psi \\ \omega \\ \frac{\theta_3}{\theta_1}\omega^2 - \frac{\theta_4}{\theta_1} \\ -\frac{\theta_5}{\theta_2}u\omega - \frac{\theta_6}{\theta_2}\omega \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{\theta_1} & 0 \\ 0 & \frac{1}{\theta_2} \end{bmatrix} \begin{bmatrix} u_{ref} \\ \omega_{ref} \end{bmatrix} + \begin{bmatrix} \delta_x \\ \delta_y \\ 0 \\ \delta_u \\ \delta_\omega \end{bmatrix}, \quad (6)$$

where u_{ref} and ω_{ref} are the desired values of the linear and angular velocities, respectively, representing the input signals for the system. A vector of identified parameters and a vector of parametric uncertainties are associated with the above model of the mobile robot, which are,

$$\boldsymbol{\theta} = [\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4 \quad \theta_5 \quad \theta_6]^T \quad (7)$$

and

$$\boldsymbol{\delta} = [\delta_x \quad \delta_y \quad 0 \quad \delta_u \quad \delta_\omega]^T \quad (8)$$

where δ_x and δ_y are functions of the slip velocities and the robot orientation, δ_u and δ_ω are functions of physical parameters as mass, inertia, diameters of the wheel and tyre, parameters of the motors, forces on the wheels, etc., are considered as disturbances.

The parameters included in vector $\boldsymbol{\theta}$ are functions of some physical parameters of the robot, such as a mass, m; moment of inertia, I_z at point G; the electrical resistance, R_a of its motors; the electromotive constant, k_b of its motors; the torque constant, k_a of its motors; the friction coefficient, B_e; the moment of inertia I_e of each group rotor-reduction gear-wheel; the radius r of the wheels; the nominal radius, R_t of the tyres; and the distances b and d.

It is assumed that the robot have PID controllers to control the velocities of each dynamic parameter, with proportional gains $K_{PT} > 0$ and $K_{PR} > 0$, and derivative gains $K_{DT} > 0$ and $K_{DR} > 0$. It is also assumed that the motors associated to the driven wheels are DC motors with identical characteristics, with negligible inductance. The equations describing the parameters θ_i were firstly proposed by De La Cruz and Carelli [9] in their wheeled chair design, and later reproduced by Martin *et al.* [1] to compensate in his adaptive controller design. In detail, θ_i are

$$\theta_1 = \frac{\left[\frac{R_a}{k_a}(mR_t r + 2I_e) + 2rk_{DT}\right]}{(2rk_{PT})}$$

$$\theta_2 = \frac{\left[\frac{R_a}{k_a}\left(I_e d^2 + 2R_t r(I_z + mb^2)\right) + 2rdk_{DR}\right]}{(2rdk_{PR})}$$

$$\theta_3 = \frac{R_a}{k_a}\frac{mbR_t}{2k_{PT}}$$

$$\theta_4 = \frac{\frac{R_a}{k_a}\left(\frac{k_a k_b}{R_a} + B_e\right)}{(rk_{PT})} + 1$$

$$\theta_5 = \frac{R_a}{k_a}\frac{mbR_t}{dk_{PR}}$$

$$\theta_6 = \frac{R_a}{k_a}\left(\frac{k_a k_b}{R_a} + B_e\right)\frac{d}{2rk_{PR}} + 1$$

where $\theta_i > 0, i = 1, \ldots, 6.$ (9)

Parameters θ_3 and θ_5 will be null if and only if, the center of mass **G** is exactly in the central point of the virtual axis linking the traction wheels (point **B**), i.e. $b = 0$. In this paper it is assumed that $b \neq 0$. The robot's model presented in Eq. (6) is partitioned into a kinematic part and a dynamic part, as shown in Figure 2. Therefore, two controllers are implemented, based on feedback linearization, for both the kinematic and dynamic models of the robot. For this simulation setup, the value of the dynamic vector are set as $\theta_1 = 0.3088$; $\theta_2 = 0.3350$; $\theta_3 = 0.0007125$; $\theta_4 = 1.2484$; $\theta_5 = 0.005$; and $\theta_6 = 1.3207$. This is the dynamic vector parameters determined by Martin *et al.* [2] based on PIONEER 3-DX robot.

3.0 PID CONTROLLER

The parallel architecture of PID controller (after this is referred to as PID controller) such shown in Figure 3 sums up the error signals, *e(t)* by comparing the desired and actual linear and angular velocities. The error signals, *e(t)*are then being multiplied by PID gains, *Kp*, *Ki* and *Kd* to produce the input signal, *u(t)*. The 'tuning' or 'design' of PID controller is the adjustment process of the values *Kp*, *Ki* and *Kd*. There are two categories of tuning approaches, which are the conventional and the alternative approaches. The empirical and the analytical methods, widely used by control designers are considered as the conventional approaches. The alternative approaches are limited to

methods that employ the stochastic process in the tuning rules.

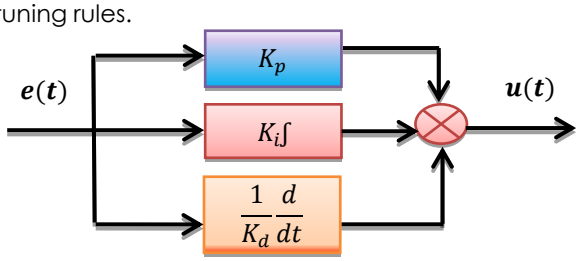


Figure 3 Parallel architecture of PID Controller

3.1 Conventional PID Tuning Method

Most of the conventional PID tuning methods are empirical tuning approaches while the analytical tuning approaches are limited to a few number reported. The most popular empirical PID tuning method is the classical Ziegler and Nichols (1942) method where the PID parameters are experimentally tuned in order to get the best outcome. To perform this method, the gains *Ki* and *Kd* are set to zero while the gain *Kp* is kept increasing until it reaches the ultimate gain value, *Ku*. *Ku* is determined when the output response is oscillating with constant amplitude (which is *Ku*) at the ultimate period, *Tu*.

3.2 PID Tuned with PSO

The basic PSO algorithm consists of three steps: generating particles positions and velocities, velocity update, and finally, position update. Here, a particle refers to a point in the design space that changes its position from one move (iteration) to another based on velocity updates. First, according to Hassan *et al.* [7], the positions, x_i^k, and velocities, v_i^k, of the initial swarm of particles are randomly generated using the upper and lower bounds on the design variables values, x_{min} and x_{max} as expressed in Eqs. (9) and (10).

$$x_i^0 = x_{min} + rand(x_{max} - x_{min}) \quad (9)$$

$$v_i^0 = x_{min} + rand(x_{max} - x_{min}) \quad (10)$$

The second step is to update the velocities of all particles at iteration $k+1$ using the objective or fitness values of particles. The fitness function value of a particle determines which particle has the best global value in the current swarm, $gbest$ and also determines the best position of each particle over iteration, $pbest$, The three values that effect the new search direction, namely, current motion, particle own memory, and swarm influence, are incorporated via a summation approach as shown in Eq. (11) with three weight factors, namely, inertia factor, w, self-confidence factor, c_1, and swarm confidence factor, c_2.

$$v_i^{k+1} = w \cdot v_i^k + c_1 \cdot rand \cdot (pbest - x_i^k) + c_2 \cdot rand \cdot (gbest - x_i^k) \quad (11)$$

The appropriate value range for c_1 and c_2 is between 1 and 2, however 2 is the most appropriate in many cases. The following inertia weight is used based on work by Lalitha *et al.* [10] can be written as

$$w = w_{max} - (w_{max} - w_{min})k/k_{max} \quad (12)$$

where k_{max}, k is the maximum number of iterations and the current number of iterations, respectively. Where, w_{min} and w_{max} are the minimum and maximum weights, respectively. Appropriate values for w_{min} and w_{max} are 0.4 and 0.9, respectively proposed by Eberhart and Shi [11] has been considered in this simulation setup. Position update is the last step in each iteration. The position of each particle is updated using its velocity vector as shown in Eq. (13) and depicted in Figure 4.

$$x_i^{k+1} = x_i^k + v_i^{k+1} \quad (13)$$

The three steps of velocity update, position update, and fitness calculations are repeated until a desired convergence criterion is met.

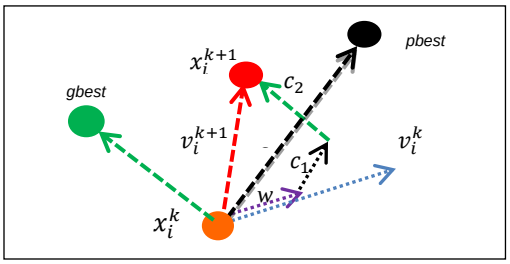


Figure 4 Velocity and position updates in PSO

In the previous section, the PID controller has been designed to control the dynamic parameter of the dynamic robots. The coefficients k_x, k_y are the kinematic control parameters and k_u, k_w are the velocity control parameters. The PID velocity controller used to maintain the velocities of the mobile robot at desired value as Serrano *et al.* [12] has experimented it on the PIONEER 3-AT robots. He suggested these values need to be positive to satisfy the stability criteria. In a conventional PID gain tuning method, these parameters are usually selected manually. It is also possible that the parameters are properly chosen, but it cannot be said that the optimal parameters are selected.

To overcome this drawback, this paper adopts the PSO for determining the optimal value of the PID dynamic control parameters. The PSO is utilized off line to determine the gain for the PID controllers. The performance of the controller varies according to adjusted parameters. As aforementioned, each PID block is comprised of one dynamic parameter. Thus, there are, in sum, three control parameters that need to be selected simultaneously for each PID representing the Kp, Ki and Kd. The controller parameters can be written as

$$[K_{pu}, K_{iu}, K_{du}, K_{p\omega}, K_{i\omega}, K_{d\omega}], \quad (14)$$

3.3 The Performance Criteria

In the present study, an integral absolute error (*IAE*) is utilized to assess the performance of the controller as it is widely adopted to evaluate the dynamic performance of the control system. Based on the calculation by Allaoua *et al.* [13], the integral of the tracking errors can be calculated as

$$IAE = \int_0^t |e(t)|dt \quad (15)$$

where $e(t)$ is the instantaneous error at each iterations.

The aim is to minimize this fitness function in order to improve the system response in terms of the steady-state errors. For fitness function calculation, the time-domain simulation of the unicycle mobile robot system model is carried out for the simulation period, t.

4.0 SIMULATION MODEL

This section discusses the simulation setup for the proposed methodology. In this study, the following values are used in simulation setup for PID controller parameter optimization [14]:

i. Dimension of the search space = 6 (i.e., $k_{i=1\ldots6}$);
ii. Population/swarm size = 15;
iii. The number of maximum iteration = 100;
iv. The self and swarm confident factor, c_1 and c_2 = 2;
v. The inertia weight factor w is set by [15], where $w_{max} = 0.9$ and $w_{min} = 0.4$;
vi. The searching ranges for the PID gains parameters are limited to (0, 200);
vii. The simulation time, t is equal to 250s;
viii. Optimization process is repeated 10 times;

4.1 Simulation Platform

In this section, the corresponding robot kinematic and dynamic parameters as explained in the earlier section are used in the MATLAB-Simulink simulation environment as shown in Figure 5. The PSO algorithm is written in MATLAB mfile.

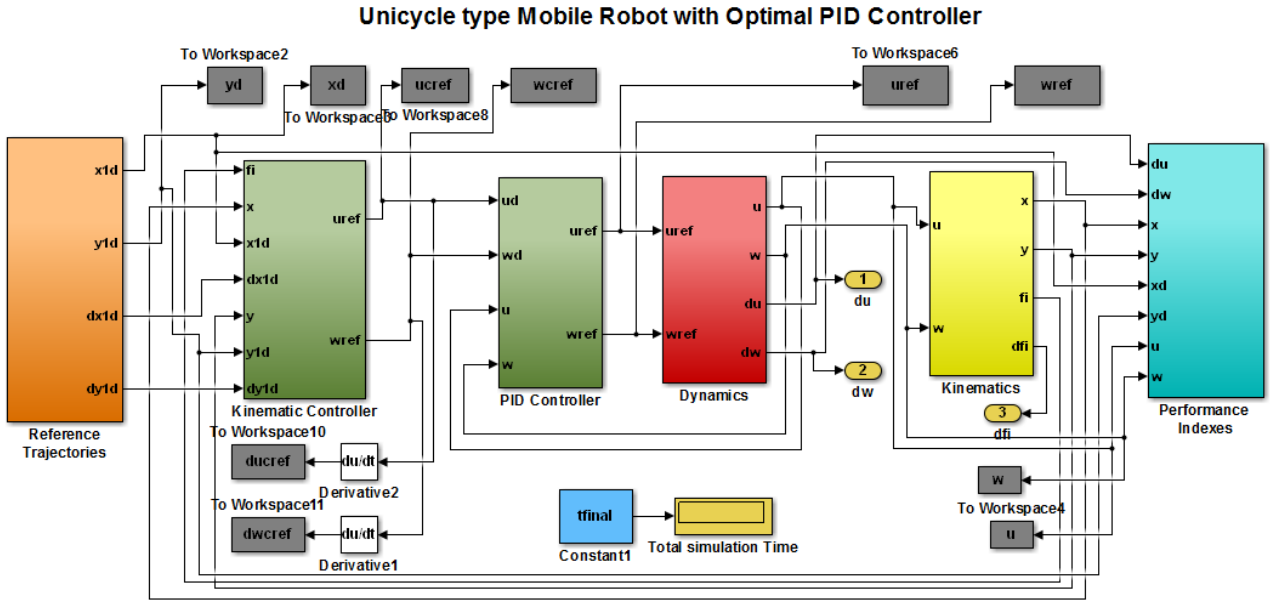


Figure 5 Simulation modeled using MATLAB-Simulink

4.2 The PID Tuning using PSO

In this section, PSO algorithm is used to optimize parameters of linear and angular velocities controller described in the previous section. The PSO creates a population of 100 swarms that contain the parameters necessary to minimize the objective function. The individuals of the PSO are coded as real values, and the values are shown as below:

$[K_{pu}, K_{iu}, K_{du}, K_{p\omega}, K_{i\omega}, K_{d\omega}]$= [66.74, 85.23, 53.17, 69.02, 71.75, 53.17]

After applying each best fitness into the PSO-PID velocity controller gains, the whole mobile robot system is simulated by taking into consideration the Pioneer 3-DX complete dynamic model including its speed and acceleration limitations. In this paper, the value for the gains for the kinematic controller, k_x and k_y are set as 1.0. Noise was added to the positions and velocities signals sent to the controllers. The period of

each simulation was set at $T = 250s$, in which the robot should follow an 8-shape trajectory (varying linear and angular speeds).

5.0 RESULTS AND DISCUSSION

From the simulations, the responses of the linear and velocity controller obtained using the proposed PSO-PID controller is shown in Figure 6. It can be seen that the linear and angular velocities are not much different with those of the reference signals. The rest of the results presented here are categorized into three aspects of verifications; i) smallest fitness function; ii) smaller distance trajectory error and iii) better performance with smaller IAE index.

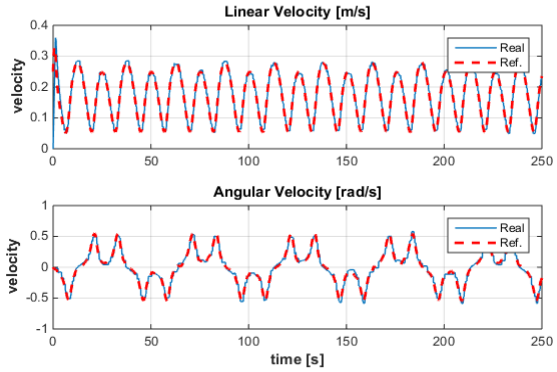


Figure 6 Linear and angular velocity responses

5.1 Verification 1- Smallest Fitness Function

The objective of utilizing PSO in tuning PID is to get the best optimal gain. The gain value is limited to between 0 and 200. As the number of iterations gets larger, the fitness value becomes smaller. The gain starts to stabilize starting at 50 iterations. The optimal value is reached at fitness 72. Although the iteration is continuing to 100, the fitness value has already reached optimal value at 72 iterations. Figure 7 illustrates this condition. As the MATLAB can capture gain values in each iteration, the one reported here is the fitness value at the point where the fitness is changing, or in other word, in the situation where the PSO found the global best for the two PID gains combination. This changing point is shown in Figure 8. Table 1 shows the fitness values of PID gains at the change points.

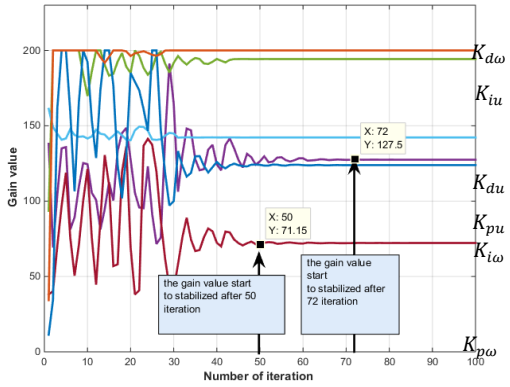


Figure 7 Value of the PID gains for the 2 PID block

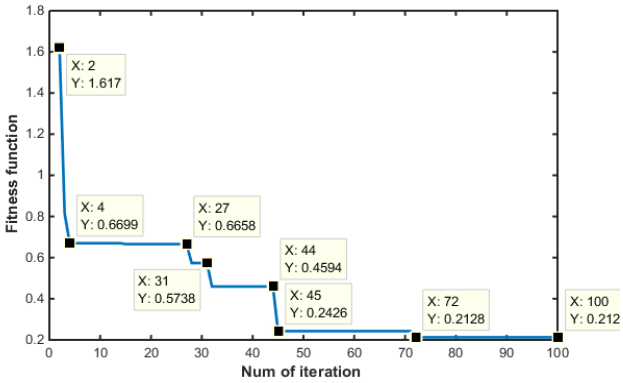


Figure 8 PID gains tuned using PSO for 100 iterations

Table 1 Fitness value of PID gains for velocity controller

No. of Iteration	Optimal gain parameters		Fitness value
	PID 1	PID 2	
2	$k_{pu} = 69.19,$ $k_{iu} = 200.0,$ $k_{du} = 149.15,$	$k_{p\omega} = 40.27,$ $k_{i\omega} = 34.53,$ $k_{d\omega} = 200.0,$	1.617
4	$k_{pu} = 135.05,$ $k_{iu} = 200.0,$ $k_{du} = 140.72,$	$k_{p\omega} = 96.92,$ $k_{i\omega} = 200.0,$ $k_{d\omega} = 200.0,$	0.6699
27	$k_{pu} = 75.86,$ $k_{iu} = 199.13,$ $k_{du} = 141.05,$	$k_{p\omega} = 87.22,$ $k_{i\omega} = 147.44,$ $k_{d\omega} = 197.56,$	0.6658
31	$k_{pu} = 106.23,$ $k_{iu} = 196.27,$ $k_{du} = 142.15,$	$k_{p\omega} = 58.18,$ $k_{i\omega} = 133.25,$ $k_{d\omega} = 200.0,$	0.5738
45	$k_{pu} = 127.47,$ $k_{iu} = 194.27,$ $k_{du} = 142.25,$	$k_{p\omega} = 72.18,$ $k_{i\omega} = 123.88,$ $k_{d\omega} = 200.0,$	0.2426
72	$k_{pu} = 127.50,$ $k_{iu} = 194.27,$ $k_{du} = 142.25,$	$k_{p\omega} = 72.22,$ $k_{i\omega} = 123.89,$ $k_{d\omega} = 200.0,$	**0.2128**
100	$k_{pu} = 127.50,$ $k_{iu} = 194.27,$ $k_{du} = 142.25,$	$k_{p\omega} = 72.22,$ $k_{i\omega} = 123.89,$ $k_{d\omega} = 200.0,$	0.2128

5.2 Verification 2- Smaller Distance Trajectory Errors

As described in earlier section, the objective of the design of the controller is to reduce the distance error. It is important as it will determine the ability of the controller to ensure the robot position to closely follow a desired trajectory. Figure 9 shows the result for robot trajectory compared with the reference (8-shape), the robot trajectory with GA and robot trajectory run with PSO with 10 iterations. From the graph, it shows that with the PSO-PID controller, the robot has managed to track the reference trajectory with very minimal error compared with GA controller.

The trajectory is related to the distance the robot compared with the reference. As the error is calculated using Eq.5 the result of the distance error is shown in Figure 10. It can be seen that the proposed PSO-PID controller is able to reduce the distance error much better compared with the GA. The distance error can be reduced much better using larger iterations. This can be seen closely in Figure 11.

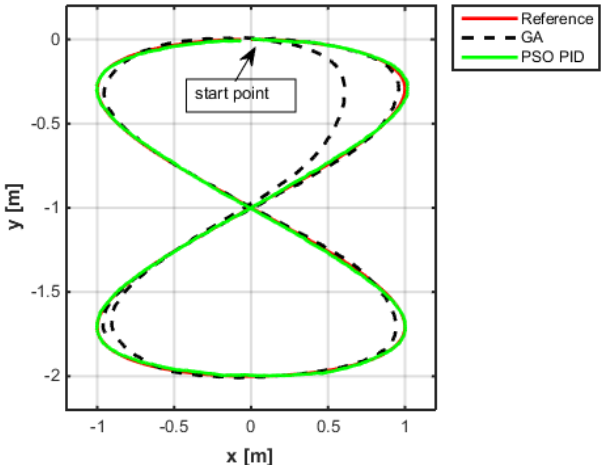


Figure 9 Robot Trajectory (8-shape reference trajectory)

5.3 Verification 3- Better Performances with Smaller IAE Index

As discussed in the earlier section, the performance criterion is determined based on the IAE and Energy indexes. Some of the resulting PID gains selected by the PSO, associated with the corresponding IAE and Energy indexes, are shown in Table 2. It can be seen that the smallest IAE value is achieved by the set of bigger gains for dynamic model. It can be seen that the PSO-PID gains also do not consume much energy compared with other approaches, moreover, the PSO-PID controller seems to have smaller IAE index, which indicates stable robot performance.

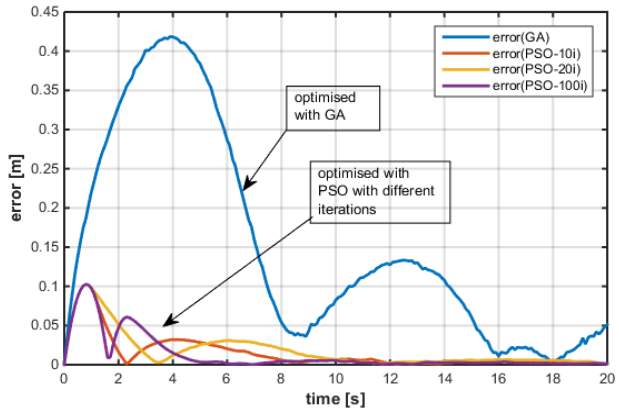


Figure 10 Distance errors compared using GA and PSO

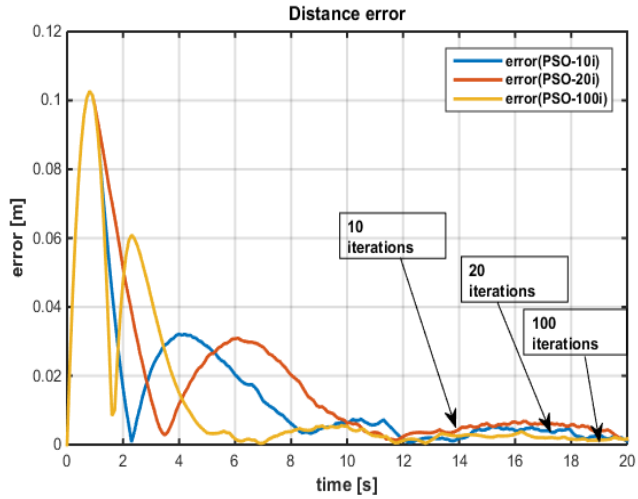


Figure 11 Distance error with different PSO iteration rate

Table 2 Gain selections associated with the corresponding IAE and energy indexes

Controller Approach	Gains parameters				Performance index	
	kinematic		dynamic			
	K_x	K_y	K_u	K_ω	*IAE*	*ENERGY*
Dynamic controller,[1]	1.0	1.0	4.0	4.0	4.815	27.54
Dynamic controller with GA, [2]	0.9713	0.9713	4.0	4.0	2.571	27.62
Proposed PSO-PID controller	K_x	K_y	***PID Gains for dynamic parameters***		*IAE*	*ENERGY*

			K_{pu}	K_{iu}	K_{du}	$K_{p\omega}$	$K_{i\omega}$	$K_{d\omega}$		
PSO-PID 10 iterations	1.0	1.0	190.19	107.22	127.521	52.16	80.03	183.99	1.641	30.15
PSO-PID 20 iterations	1.0	1.0	114.18	39.01	200	97.74	185.29	22.65	1.684	30.02
PSO-PID 100 iterations	1.0	1.0	55.70	51.63	33.28	49.33	65.39	52.89	**1.095**	**29.99**

6.0 CONCLUSION

Simulation was performed using MATLAB/Simulink software, employing the dynamic model of the unicycle-like mobile robot presented in section 2 and the PID control structure optimized using PSO proposed in section 3. The presented simulation results show that PSO can be successfully used to select controller gains that result in smaller tracking error and minimizing energy consumption. From the present study, the followings can be concluded:

(a) The two PID controllers optimized with PSO for control the dynamic model of mobile robot is developed.
(b) The validity of the PSO-PID controller is verified with dynamic controller which optimized with GA.
(c) The robot trajectory tracking perform better tracking with smaller distance error compared with the reference trajectory and the robot trajectory with the GA controller.
(d) The performance IAE index is better indicating stable controller performance with nearly same energy consumption.

A limitation of the proposed tuning methods is relies on the accuracy of the robot dynamic model. Therefore, the quality of the tuning results depends on the accuracy of the robot model.

Acknowledgement

The authors express gratitude for the MyBrain15 scholarship from the Malaysian Ministry of Education (MOE) to the first author, and Universiti Teknologi Malaysia for the research facilities.

References

(1) DeVon, D. Bretl, T. 2007. Kinematic and Dynamic Control Of A Wheeled Mobile Robot. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*, San Diego, CA. *29 October 2007*. 4065-4070.
(2) Martins, F. N., Celeste, W. C., Carelli, R., Sarcinelli-Filho, M., Bastos-Filho, T. F. 2008. An Adaptive Dynamic Controller For Autonomous Mobile Robot Trajectory Tracking. *Control Engineering Practice*. 16(11): 1354-1363.
(3) Huang, J. T. 2009. Adaptive Tracking Control Of High-Order Non-Holonomic Mobile Robot Systems. *Control Theory and Applications, IET*. 3(6): 681-690.
(4) Oriolo, G., De Luca, A., Vendittelli, M. 2002. WMR Control Via Dynamic Feedback Linearization: Design, Implementation, And Experimental Validation. *IEEE Transactions on Control Systems Technology*. 10(6): 835-852.
(5) Lee, T. C., Song, K. T., Lee, C. H., Teng, C. C. 2001. Tracking Control Of Unicycle Modeled Mobile Robots Using A Saturation Feedback Controller. *IEEE Trans. Control System Technology*. 9(2): 305-318.
(6) Martins, F. N., Almeida, G. M., IFES, C. S. 2012. Tuning a Velocity-based Dynamic Controller for Unicycle Mobile Robots with Genetic Algorithm. *Jornadas Argentinas de Robotica, JAR*. 12: 262-269.
(7) Hassan R, Cohanim B, De Weck O, Venter G. 2005. A Comparison Of Particle Swarm Optimization And The Genetic Algorithm. *Proceedings of the 1st AIAA Multidisciplinary Design Optimization Specialist Conference*. Austin, Texas. 18 April 2005. 18-21.
(8) Hashim, N. L. S., Yahya, A., Andromeda, T., Kadir, M. R. R. A., Mahmud, N., Samion, S. 2012. Simulation of PSO-PI Controller of DC Motor in Micro--EDM System for Biomedical Application. *Procedia Engineering*. 41: 805-811.
(9) De La Cruz, C., Carelli, R. 2006. Dynamic Modeling And Centralized Formation Control Of Mobile Robots. *32nd Annual Conference on IEEE Industrial Electronics (IECON)*, 2006. 6-10 November 2006. Paris. 3880-3885.
(10) Lalitha, M. P., Reddy, V. V., Usha, V. 2010. Optimal DG Placement For Minimum Real Power Loss In Radial Distribution Systems Using PSO. *Journal of Theoretical and Applied Information Technology*. 13(2): 107-116.
(11) Eberhart, R. C., Shi, Y. 2000. Comparing Inertia Weights And Constriction Factors In Particle Swarm Optimization. *Proceedings of the IEEE Congress on Evolutionary Computation*, La Jolla, CA. 16-19 July 2000. 84-88.
(12) Serrano, M. E., Godoy, S. A., Mut, V. A., Ortiz, O. A. and Scaglia, G. J. 2015. A Nonlinear Trajectory Tracking Controller For Mobile Robots With Velocity Limitation Via Parameters Regulation. *Robotica*. 1-20.
(13) Allaoua, B., Gasbaoui, B., Mebarki, B. 2009. Setting up PID DC Motor Speed Control Alteration Parameters Using Particle Swarm Optimization Strategy. *Leonardo Electronic Journal of Practices and Technologies*. 14: 19-32.
(14) Basri, M, A., Danapalasingam, K. A., Husain, A. R. 2014. Design and Optimization of Backstepping Controller for An Underactuated Autonomous Quadrotor Unmanned Aerial Vehicle. *Transactions of FAMENA*. 38(3): 27-44.