JOURNAL OF MECHANICS OF CONTINUA AND MATHEMATICAL SCIENCES
www.journalimcms.org

ISSN (Online) : 2454 -7190 Vol.-15, No.-1, January (2020) pp 73-95 ISSN (Print) 0973-8975

OPTIMUM PAATH TRACKING AND CONTROL FOR A WHEELED MOBILE ROBOT (WMR)

Kawther K Younus[1], Nabil H Hadi[2]

[1]University of Baghdad College of Engineering / Mechanical Engineering
[2]University of Baghdad College of Engineering / Aeronautical

E-mail : [1]k.yones0903@coeng.uobaghdad.edu.iq

https://doi.org/10.26782/jmcms.2020.01.00006

Abstract

This work studies the trajectory tracking of a non-holonomic WMR. A type of back stepping method in conjunction with Lyapunov method were used for deriving two controllers. But, in non-linear systems controllers may not be enough to reach a good performance. Different cases of trajectory where studied such as (straight line, circular, elliptical, sinusoidal, and infinity shape trajectory) to examine the WMR control system utilizing MATLAB (R2018a)/Simulink to simulate the results. In addition, particle swarm optimization technique was utilized to determine the controllers' parameters by implementing the summation absolute compound error for the position (x, y), the orientation β, the linear and angular velocity (v_c,ω_c), and the energy. Results showed a very good matching between simulation and the desired trajectory where all errors converge to zero.

Keywords: Mobile robot, Nonholonomic, DDWMR, Optimum, PSO, control

Table 1: Nomenclature

Table 1.a

Subscripts	
g	Global frame
r (up)	Robot frame
R	Pertaining to the right wheel
L	Pertaining to the left wheel
o	Pertaining to the center point between the driving wheels
c(down)	Pertaining to the center of mass
w(down)	Wheel
d	Desired
opt.	Optimized
tun.	tuned

Table 1.b

Abbreviations	
Abbreviation	**Definition**
WMR	Wheeled mobile robot
PSO	particle swarm optimization
SACE	Summation absolute compound error

Table 1.c

symbols		
Symbol	**Description**	**units**
OXY Z	Main or global axis	
oxy z	Robot or local axis	
$\boldsymbol{v}$	Liner velocity	m/s
$\boldsymbol{\omega}$	Angular velocity	rad/s
$\boldsymbol{\beta}$	Angular position of the robot	rad
$\dot{\boldsymbol{\gamma}}$	Angular speed of the wheels	rad/s^2
2L	Distance between wheels	m
b	Distance from wheel's axis to center of mass	m
R	Wheels radius	m
M(q)	Inertia matrix	Kg
$\boldsymbol{A^T(q)}$	The nonholonomic constraint matrix associated with the kinematic constraints	
λ	The Lagrange multiplier vector	
$\boldsymbol{q_i}$	The generalized coordinates	
k	Gain	
m	The mass of the platform	Kg
I	The inertia of the platform about its rotational axis	kg.m^2

I. Introduction

Wheeled mobile robots usually the most interesting kinds of mobile robots. The principal reasons for interesting applications of this systems are Fast maneuver capability, energy efficiency, and simple control approaches. WMR has attracted great attention in unlimited fields such as medicine, service, domestic needs, industry, mining transportation, aerospace, entertainment, and in dangerous areas applications.

Motion control is the strategy were the MR reaches the desired location.Controllers have been designed to help the WMR complete its goal successfullyand to ensure efficient performance.
On the other hand, optimization technique could be utilized to ensure efficient performance. Particle Swarm Optimization is a strong technique derived from the manner and motion of insects' swarms, such as bees, ants, and termites; a school of fish; or birds flock. It performs the social communication model to find the problem solution.

The wild applications of WMR in different fields attracted many researchers such as Saleh A. L. et al, (2018) presented the kinematic and dynamic typical model of wheeled mobile robot. Two Fuzzy Neural Petri Net (FNPN) controllers in addition to the particle swarm method (PSO) were used to solve the trajectory tracking problem for the azimuth and the velocity. These two controllers offer the accuracy and the required performance to the robot motion that makes the robot capable to work in dangerous environment. Simulation results showed that the suggested controllers is very efficient in target tracking problem with different paths and has suitable dynamic performance[XII].Aouf A. et al (2018)presented the kinematic model of Khepera III mobile robot and investigated the novel evolutionary techniquebased on Teaching-Learning-Based Optimization (TLBO) to use it as an alternative method for the navigation problem solution. This technique was simulated and compared with other methods such as invasive weed optimization (IWO), particle swarm optimization (PSO), and biogeography-based optimization (BBO). Simulation results demonstrated that the proposed algorithm is an efficient alternative method in solving the WMR navigation problem [I].

Pandey A. and Parhi D. R. (2017)introduced a singleton type one fuzzy logic system (T1-SFLS) as a controller in addition to a Fuzzy-WDO hybrid for the WMR collision avoidance and navigation in an unknown environment. The Wind Driven Optimization (WDO) algorithm is utilized for optimizing and tuning the parameters of the fuzzy controller. Khepera-III WMR was used in real-time experiments. Simulation and experimental results showed that Fuzzy-WDO algorithm gives good agreement in comparison with the T1-SFLS controller in the case of WMR navigation.[XI]. Maghenem M. et al (2017)Considered the typical kinematic and dynamic model of WMR. The tracking problem of the robot was investigated by using a simple time-varying controller. In this control method a δ-persistently exciting controller was considered as the kinematic controller which control the robot velocity. In addition the inertia of the robot was considered as unknown. Simulation results showed that the controller guarantees the velocity tracking errors convergence [VIII].

Nurmaini S. et al (2017)presented the kinematic model of the differential drive WMR and investigated the robot control when it move from a starting position to a target position. The linear feedback control law was suggested with pole placement method to achieve the desired trajectory. By utilizing the scheme, the performance of the controller revealed regularity and fast position errors convergence is gained

[IX].Leenaa N. and Sajub K.K. (2016) modeled the kinematics and typical actuator dynamics of a nonholonomic differential drive WMR. Trajectory tracking problem was solved by designing two controllers which are the outer controller (kinematic controller) and the inner loop controller (a simple P controller). The trajectory tracking control scheme was performed in MATLAB Simulink. Results revealed accurate trajectory tracking for the control algorithm and the model [VII].

Han P. et al (2016) studied the robot path planning in a known static environment. The ant colony optimization with the influence of critical obstacle (ACOIC) approach was used and compared with the traditional ant colony optimization (ACO) approach where the ACOIC has the ability to choose the way toward the more desirable direction instead of considering all directions with the same weight as in traditional ACO. Experimental results proved that the suggested method (ACOIC) is more efficient than (ACO) in locating the shortest path [V].Hadi N. H. (2005) Developed a control model of a 2-WMR with PID fuzzy logic control systemin addition to a sliding suppression algorithm. Results showed state correction and the WMR moved according to the desired trajectory with more quality improvement in trajectory program [IV].

The contribution of this work includes:

- ❖ Design of a control law to track the desired trajectory using the backstepping method
- ❖ Investigating the optimized minimum error percentage by finding the control gain
- ❖ Building of aMATLABSimulink model(using MATLAB R2018a) to simulate the results.

II. Modeling of the Differential-Drive Wheeled Mobile Robot

II.i. Kinematic model

The position and orientation of WMR could be described by two coordinate systems (frames) which are initial (global) coordinate system and robot (local) coordinate system. The global frame is fixed in the plane of the robot movement and denoted as $\{x^g, y^g, \beta\}$ or $\{X, Y, \beta\}$, while the robot frame is attached to the robot and moves with it, this coordinate denoted as $\{x^r, y^r, \beta\}$ or $\{x, y, \beta\}$ as shown in the WMR model in Figure 1. Now according to assumption of pure rolling and the non-holonomic condition the kinematic model can be written in terms of the linear and angular velocities in the global coordinates as follows:

$$\dot{q}^g = \begin{bmatrix} \dot{x}_o^g \\ \dot{y}_o^g \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} \cos\beta & 0 \\ \sin\beta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \text{[XIII], [XIV]} \qquad (1)$$

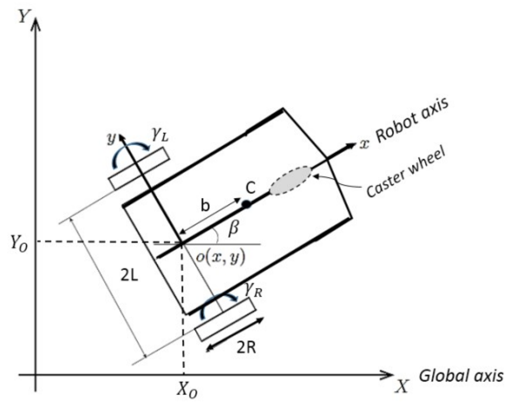


Figure 1: Two-wheeled DDMR

II.ii. Dynamic Modeling

The dynamic equation of motion for non-holonomic system could be investigated by assuming there is no friction and no unknown disturbance, on the other hand due to the robot's movement on the horizontal plane the gravitational vector and the potential energy will be equal to zero, so the equation of motion becomes:

$$M(q)\ddot{q} = B(q)\,\tau + A^T(q)\,\lambda \tag{2}$$

Now by using the Lagrange dynamic approach and after simplifications and eliminating the Lagrange multiplier the dynamic equations referred to the center of rotation (o) of the robot areas follows:

$$\dot{v}= \frac{\tau_1}{m},\ \dot{\omega} = \frac{\tau_2}{m} \tag{3.a,b}$$

Where:

$$\tau_1= \frac{\tau_R + \tau_L}{R}\ ,\ \tau_2 = \frac{2L(\tau_R - \tau_L)}{R} \tag{4.a, b}$$

Note that these dynamic equations are the equations used to design the motor controller depending on the torque.

III. Differential drive WMR control

In this study the WMR controlled by two controllers which are kinematic and dynamic controllers. The linear and angular velocities resulting from the kinematic controller will be the input velocities for the dynamic controller that generates the torques of the wheels. The procedure of this control is classified as a backstepping control.

III.i. Kinematic controller

According to Figure 2 where the desired trajectory satisfy the nonholonomic and the kinematic equations, the global errors $\breve{q} = [\breve{x} \quad \breve{y} \quad \breve{\beta}]^T$ represent the difference between the desired and actual position or orientation where $\breve{x}$ =$x_d - x$, $\breve{y}$ =$y_d - y$, $\breve{\beta}$ =$\beta_d - \beta$,q_d=$[x_d \quad y_d \quad \beta_d]^T$,and q=$[x \quad y \quad \beta]^T$. then by using the transformation matrix we get the errors in the local frame as follows:

$$\begin{bmatrix} \breve{x}_r \\ \breve{y}_r \\ \breve{\beta}_r \end{bmatrix} = \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \breve{x} \\ \breve{y} \\ \breve{\beta} \end{bmatrix} \tag{5}$$

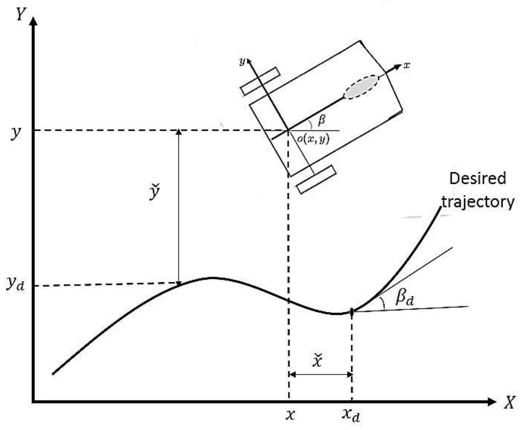


Figure 2: Robot trajectorytracking model

Then by differentiating Equation (5) we get the dynamic error$\dot{\breve{q}}_r$=$[\dot{\breve{x}}_r \quad \dot{\breve{y}}_r \quad \dot{\breve{\beta}}_r]^T$ as follows:

$$\dot{\breve{x}}_r = v_d \cos\breve{\beta}_r - v + \breve{y}_r \qquad \text{[II] (6)}$$

$$\dot{\breve{y}}_r = v_d \sin\breve{\beta}_r - \breve{x}_r\omega \qquad \text{[II] (7)}$$

$$\dot{\breve{\beta}}_r = \omega_d - \omega \qquad \text{[II] (8)}$$

Now to make the error convergence to zero, the structure of the controller will be determined utilizingLyapunov's second method, starting by choosing the following candidate function

$$\text{V}\,(\breve{q}_r) = \frac{1}{2}(\breve{x}_r^2 + \breve{y}_r^2) + (1 - \cos\breve{\beta}_r) \tag{9}$$

This candidate function should satisfy the following Lyapunov function properties [XIV]:

1- V ($\breve{q}_r$) this function and its derivative is continuous
2- V (0) = 0
3- V ($\breve{q}_r$)> 0 for all $\breve{q}_r \neq 0$
4- $\frac{\mathrm{dV}\,(\breve{q}_r)}{dt} < 0$ for $\breve{q}_r \neq 0$

Since Equation (9) satisfies the first three properties we should check the fourth one by taking the time derivative of (V) to get the following:

$$\dot{V}\,(\breve{q}_r) = (v_d \cos\breve{\beta}_r - v)\breve{x}_r + (\omega_d - \omega + v_d \breve{y}_r) \qquad (10)$$

The error dynamics will be asymptotically stable by using the following control law [III], [XV]:

$$v_c = k_x\,\breve{x}_r + v_d \cos\breve{\beta}_r$$

$$\omega_c = k_\beta \sin\breve{\beta}_r + v_d \breve{y}_r + \omega_d \qquad (11.a,b)$$

Where k_x andk_β are positive gains, it is clear that this velocity control inputs makes $\dot{V}\,(\breve{q}_r) < 0$ as follows:

$$\dot{V}\,(\breve{q}_r) = -(k_x\,\breve{x}_r^2 + k_\beta \sin^2\breve{\beta}_r) \qquad (12)$$

III.ii. Dynamic controller

The velocity error equation will be m$\dot{\breve{v}}_c$+$k_a\,\breve{v}_c$= 0, I$\dot{\breve{\omega}}_c$ +$k_b\,\breve{\omega}_c$= 0 if the chosen input torques are as follows:

$$\tau_1 = \mathrm{m}\,\dot{v}_c + k_a\,\breve{v}_c,\ \tau_2 = \mathrm{I}\,\dot{\omega}_c + k_b\,\breve{\omega}_c \qquad (13.\ a,\ b)$$

Where the velocity errors are$\breve{v}_c = v_c - v$, $\breve{\omega}_c = \omega_c - v$and the velocity dynamic errorsare$\dot{\breve{v}}_c = \dot{v}_c - v$, $\dot{\breve{\omega}}_c = \dot{\omega}_c - \omega$, and the gains k_a and $k_b > 0$ which means that the velocity errors ($\breve{v}_c$,$\breve{\omega}_c$) are stable and converge to zero.

IV. Particle Swarm Optimization (PSO)

PSO is a computational stochastic optimization method utilized for solving optimization problems. In this adopted algorithm the term particle referees to each individual solution. Multiple flying particles are applied in a specific searching space and each one has an initial velocity. The main idea of PSO is to quick every particle in the best self-position direction (pbspd) and the gained global best position (gbp) is quicken at every time step by a random weight. Each particle in the swarm saves a previous information and information of neighboring particles then by utilizing the best previous position, neighborhood position, and velocity the particle velocity could be determined and used to update the particle position as demonstrating in the following formulas.

$v_{t+1} = w * v_t + c_1 * \text{rand}\,(0,1) * (\text{psbpd} - x_t) + c_2 *$ rand $(0,1) * (gbp - x_t)$ [VI] (14)

$x_{t+1} = x_t + v_{t+1}$ [VI] (15)

Where the inertial weight w and the acceleration coefficients $c_1$and $c_2$taken as unity. Now if$(fitness_{x_t})$is better than $(fitness_{\text{psbpd}})$then $(fitness_{\text{psbpd}})$ =$(fitness_{x_t})$, and (psbpd) = (x_t). At last the looping will converge to $(fitness_{\text{gpb}})$ =the better$(fitness_{\text{psbpd}})$. The pso flow chart describe its technique as shown in Figure 3.

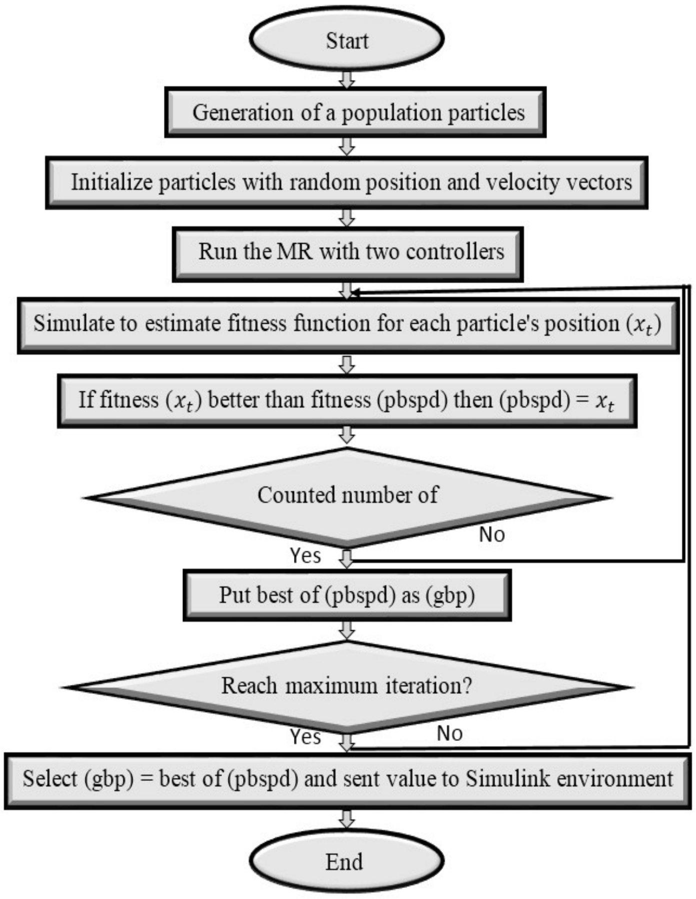

Start
Generation of a population particles
Initialize particles with random position and velocity vectors
Run the MR with two controllers
Simulate to estimate fitness function for each particle's position (x_t)
If fitness (x_t) better than fitness (pbspd) then (pbspd) = x_t
Counted number of
No
Yes
Put best of (pbspd) as (gbp)
Reach maximum iteration?
Yes
No
Select (gbp) = best of (pbspd) and sent value to Simulink environment
End

Figure 3: PSO flow chart

The particle swarm optimization algorithm approach is utilized as M file that linked to the Simulink model where the controllers' parameters are evaluated and sent to the controllers GUI. The PSO initial parameters are listed in Table 2.

Table 2: PSO initial parameters

parameter	**value**
number of particles	50
number of dimensions	7
maximum iteration	50
C1=C2	2
Objective function	Summation absolute compound error (SACE)

The initial values of the controllers' parameter are generated in the PSO program and sent to the simulation model Figure4. Which is running automatically and evaluating the objective function that is send to the PSO program for improving the value of its parameters. Finally when the iterations finished, the optimized control parametersK_x,K_β,K_a, and $K_b$have been determined as shown in Table 3, and Figure. 5 shows how the objective function varies with the number of iteration.

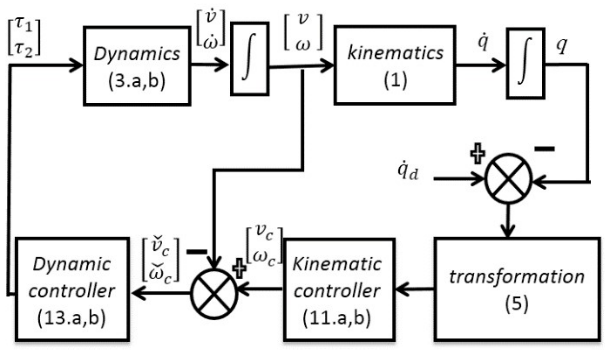

$\begin{bmatrix}\tau_1\\\tau_2\end{bmatrix}$
Dynamics (3.a,b)
$\begin{bmatrix}\dot{v}\\\dot{\omega}\end{bmatrix}$
$\int$
$\begin{bmatrix}v\\\omega\end{bmatrix}$
kinematics (1)
$\dot{q}$
$\int$
q
$\dot{q}_d$
+
−
transformation (5)
Kinematic controller (11.a,b)
$\begin{bmatrix}v_c\\\omega_c\end{bmatrix}$
+
−
$\begin{bmatrix}\breve{v}_c\\\breve{\omega}_c\end{bmatrix}$
Dynamic controller (13.a,b)

Figure 4: The entire robot trajectory tracking

Table 3: The control parameters obtained using PSO

Trajectory ⇨	circular	line	Elliptical	Sinusoidal	Infinity
Gain ⇩					
K_x	19	2.4842	9.0401	3.4188	28.9172
K_β	1.1781	1.5326	0.8025	0.516	0.0963
K_a	2.9157	5	2.1977	5.3248	16.1077
K_b	1.2437	0.5685	9.1951	0.5	0.8327

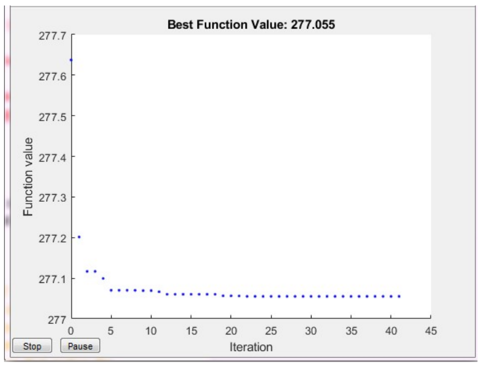


Figure 5:The objective function versus number of iteration

V. Simulation results:

a. Trajectory tracking optimization results:

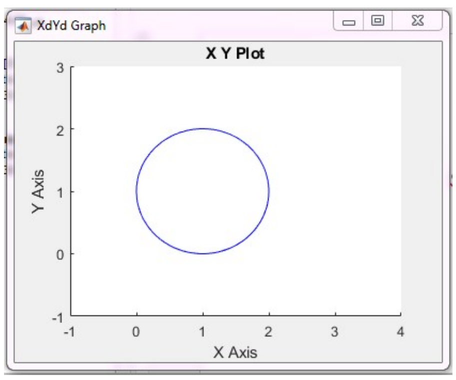
The trajectory tracking of the DDWMR was simulated using MATLAB/SIMULINk. The simulation included five trajectories which are straight line, circular, elliptical, sinusoidal, and infinity shape trajectory to verify the controllers' performance. The robot model parameter values are presented in Table 4.

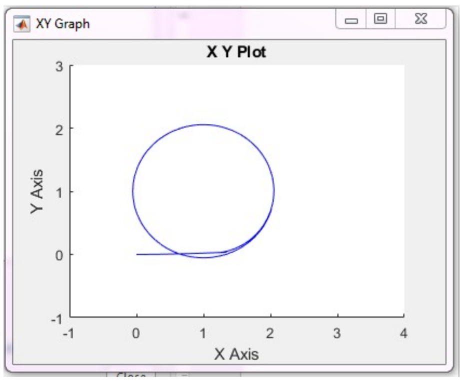
Table 4: The robot model parameter values

symbol	value	unit
M	0.5	kg
I	0.00095	Kg.m^2
L	0.05	m
R	0.0325	m

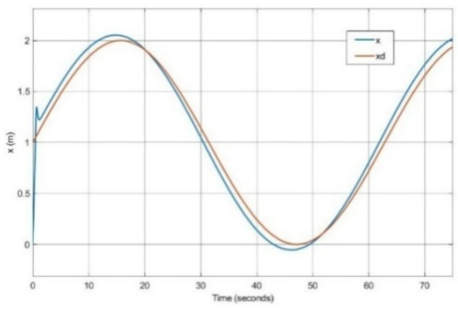
V.i.a. Circular trajectory:

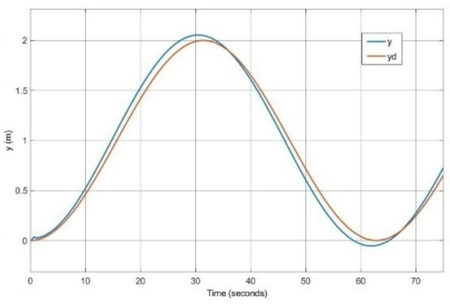
It is required that the robot follow the desired trajectory described by x_d= 1+sin$\omega_d\, t$ and y_d=1− cos $\omega_d t$ as shown in Figure 6.a.Figure 6.b shows the simulation result of factual circular trajectory. Figure 6.c shows the desired and factual values of(x). Figure 6.d shows the desired and factual values of(y). Figure 7.e shows the position and orientation errors. Figure 7.f shows the velocity error.



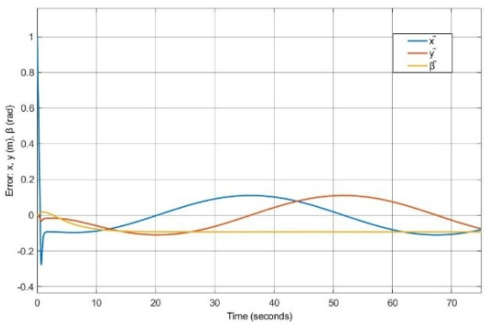


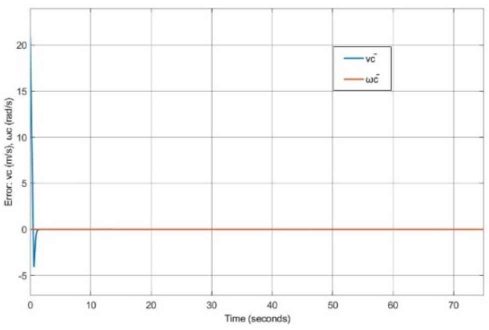
(a) : Desired circular trajectory (b):Factual circular trajectory.





(c): Desired and factual values of(x). (d): Desired and factual values of(y)



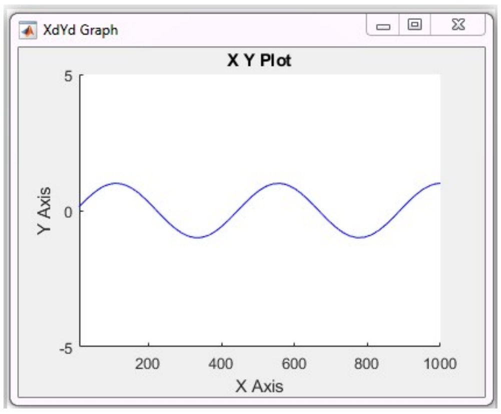


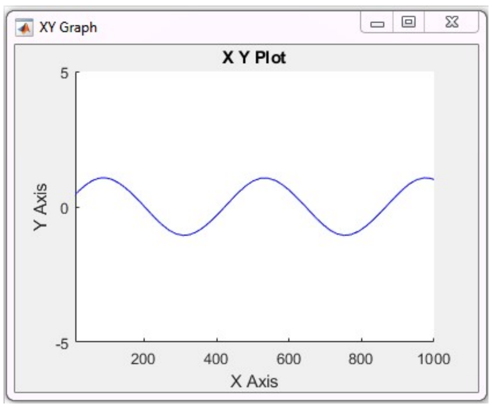
(e): Position and orientation errors(f): Velocity error

Figure 6(a), (b), (c), (d), (e), and (f): Circular trajectory.

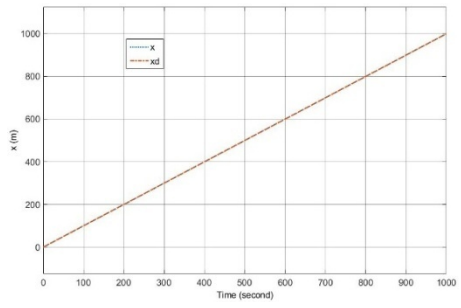
V.i.b. Sinusoidal trajectory:

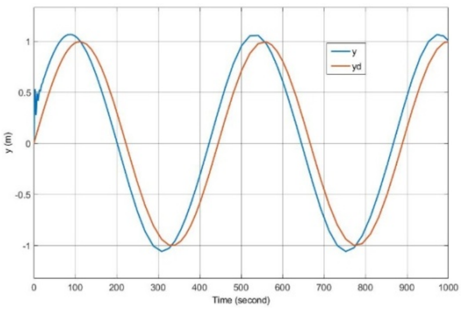
It is required that the robot follow the desired trajectory described by x_d=sin$\frac{\pi}{2}\omega_d\, t$ and y_d=sin$(\frac{\pi}{2}\omega_d\, t+\frac{\pi}{2})$ as shown in Figure 7.a. Figure 7.b shows the simulation result of factual sinusoidal trajectory. Figure 7.c shows the desired and factual values of(x). Figure 7.d shows the desired and factual values of(y). Figure 7.e shows the position and orientation errors. Figure 7.f shows the velocity error.



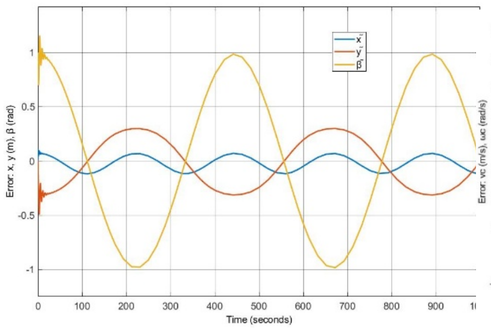


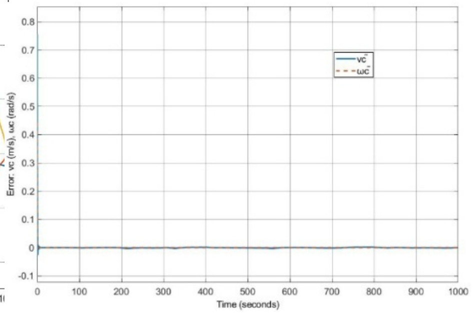
(a):Desired sinusoidal trajectory (b):Factual sinusoidal trajectory





(c): Desired and factual values of(x) (d): Desired and factual values of(y)



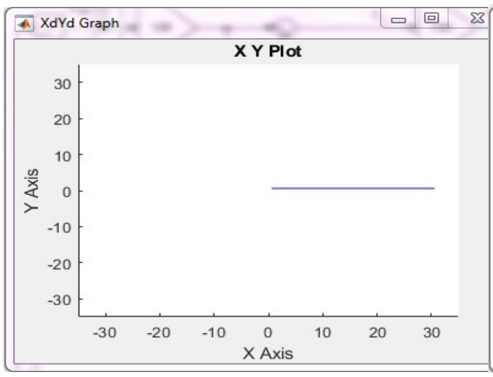


(e): Position and orientation errors (f): Velocity error

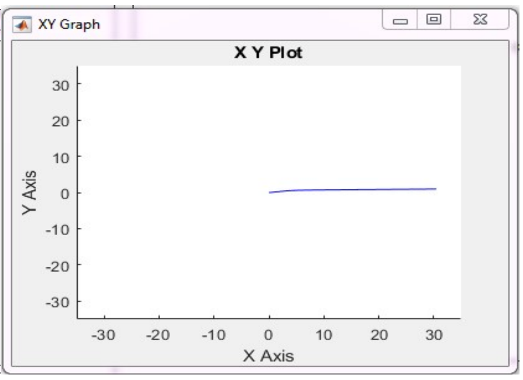
Figure 7(a), (b), (c), (d), (e), and (f): Sinusoidal trajectory.

V.i.c. Straight line trajectory:

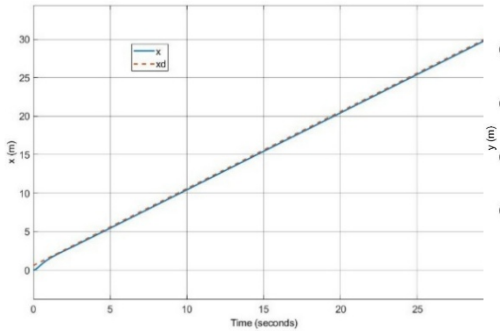
It is required that the robot follow the desired trajectory described by x_d=t+3 and y_d=3 as shown in figure 8.a. Figure 8.b shows the simulation result of factual line trajectory. Figure 8.c shows the desired and factual values of(x). Figure 8.d shows the desired and factual values of(y). Figure 8.e shows the position and orientation errors. Figure 8.f shows velocity error.

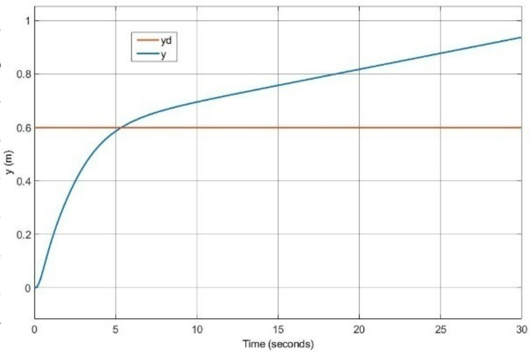


(a): Desired line trajectory

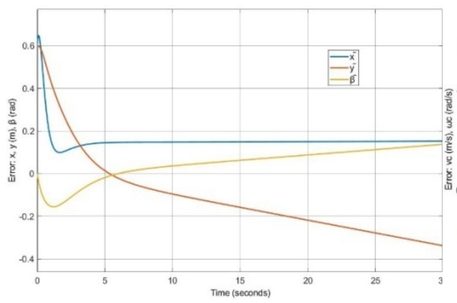


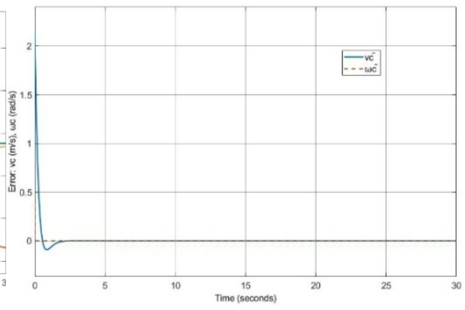
(b): Factual line trajectory





(c): Desired and factual values of(x)(d): Desired and factual values of(y)



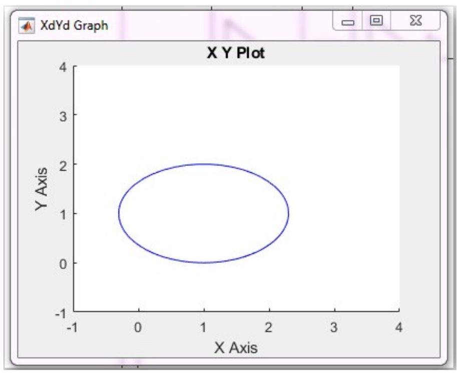


(e): Position and orientation errors (f): Velocity error

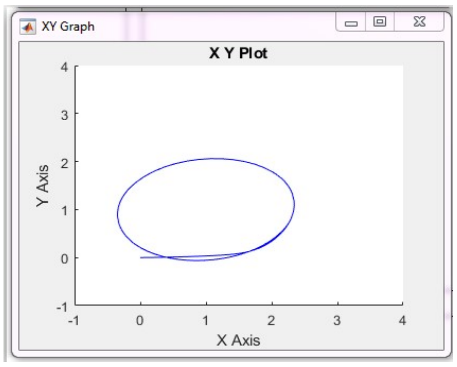
Figure 8(a), (b), (c), (d), (e), and (f):straight line trajectory.

V.i.d. Elliptical trajectory:

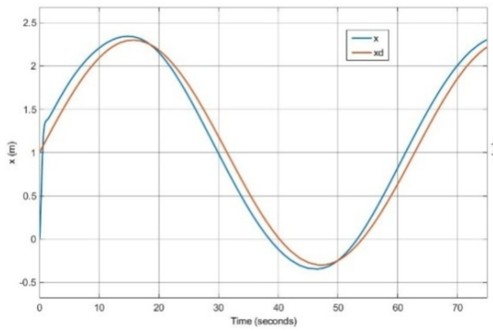
It is required that the robot follow the desired trajectory described by x_d= 1+1.3 sin$\omega_d\, t$ and y_d=1− cos $\omega_d t$ as shown in figure 9.a. Figure 9.b shows the simulation result of factual elliptical trajectory. Figure 9.c shows the desired and factual values of(x). Figure 9.d shows the desired and factual values of(y). Figure 8.e shows the position and orientation errors. Figure 9.f shows the velocity error.



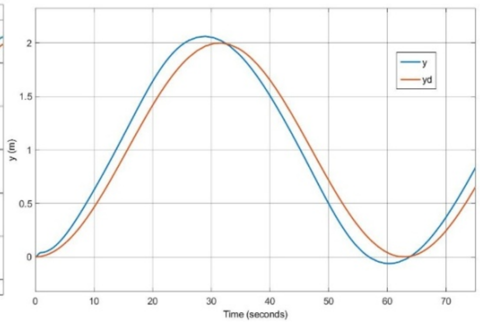
(a): Desired elliptical trajectory



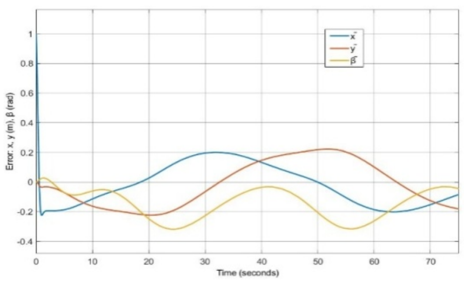
(b): Factual elliptical trajectory



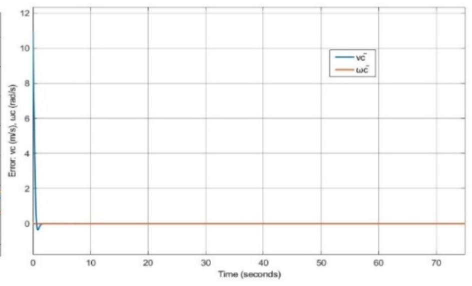
(c): Desired and factual values of(x)



(d): Desired and factual values of(y)



(e): Position and orientation errors

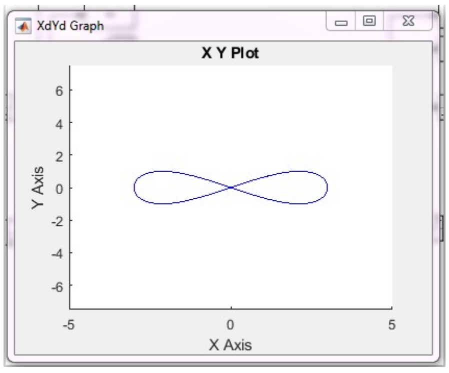


(f): Velocity error

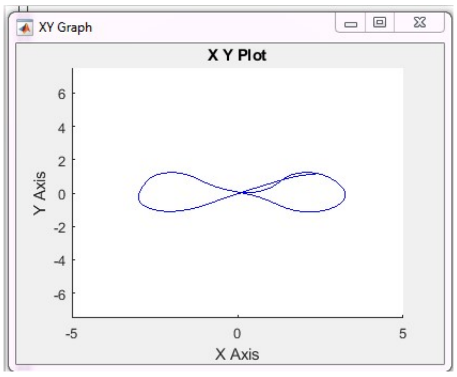
Figure 9(a), (b), (c), (d), (e), and (f):Ellipticaloptimum trajectory.

V.i.e. Infinity trajectory:

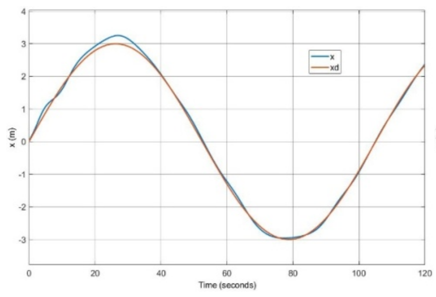
It is required that the robot follow the desired trajectory described by x_d= $3\sin\frac{2\pi}{105}t$ and y_d=$\sin\frac{4\pi}{105}t$ as shown in Figure 10.a. Figure 10.b shows the simulation result of factual infinity trajectory. Figure 10.c shows the desired and factual values of(x). Figure 10.d shows the desired and factual values of(y). Figure 10.e shows the position and orientation errors. Figure 10.f shows the velocity error.



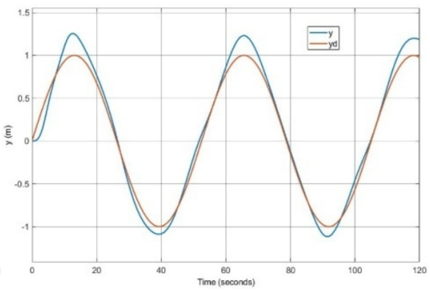
(a): Desired infinity trajectory



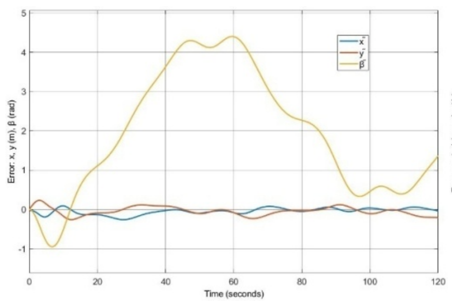
(b): Factual infinity trajectory



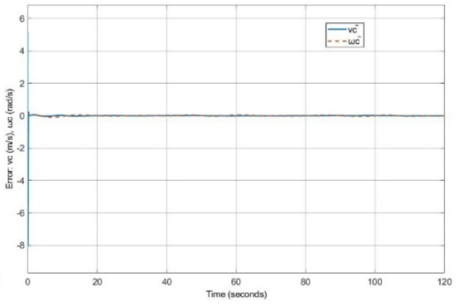
(c): Desired and factual values of(x)



(d): Desired and factual values of(y)



(e): Position and orientation errors



(f): Velocity error

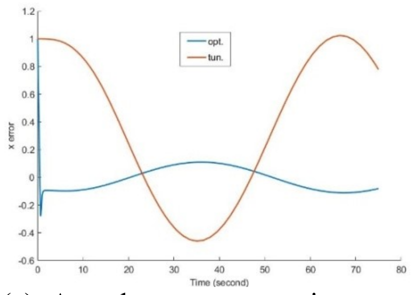
Figure 10(a), (b), (c), (d), (e), and (f):Infinity optimum trajectory.

V.ii Summation absolute minimization function results:

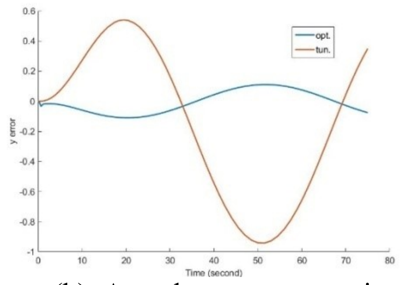
Particle swarm optimization method based on summation absolute function for minimum errors used to minimize the compound error as much as could and compared with the case of tuned parameters from [XVI] to show the modification percent. The following results show the comparison between the errors in the case of tuned control parameters and the case of optimized control parameters for each trajectory tracking. In addition Table 5 shows the mean absolute error for each variable in the tuned and optimized case and it's also shows the optimization percent for each variable.

V.ii.a. Circular trajectory:

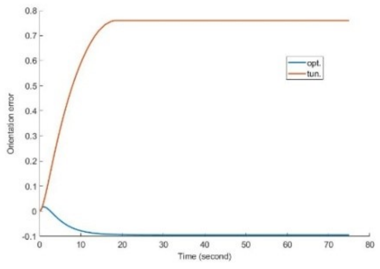
The following figures show the error comparison for the circular trajectory variables where Figure 11.a shows the actual x error comparison, Figure 11.b shows the actual y error comparison, Figure 11.c shows the actual orientation error comparison, Figure 11.d shows the actual linear velocity error comparison, Figure 11.e shows the actual angular velocity error comparison, and Figure 11.f shows the energy comparison.

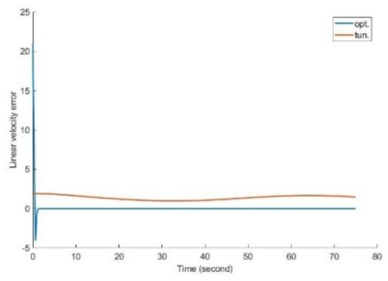


(a): Actual x error comparison

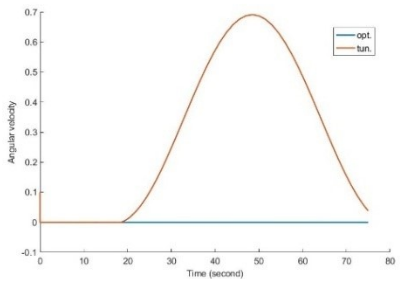


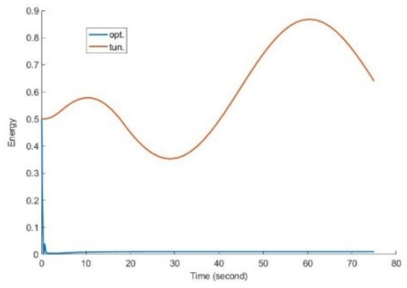
(b): Actual y error comparison





(c): Actual orientation error comparison(d): Actual linear velocity error comparison



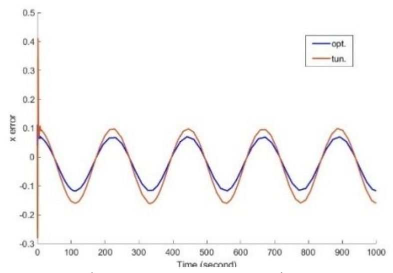


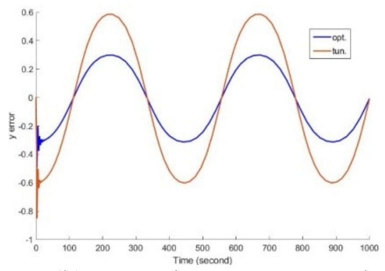
(e): Actual angular velocity error comparison (f): Energy comparison

Figure 11(a), (b), (c), (d), (e), and (f): Circular trajectory comparison

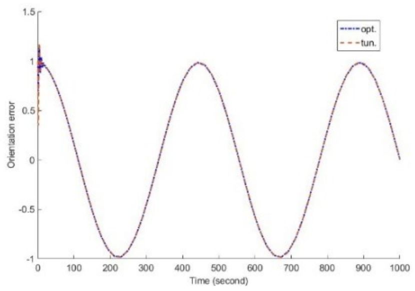
V.ii.b. Sinusoidal trajectory:

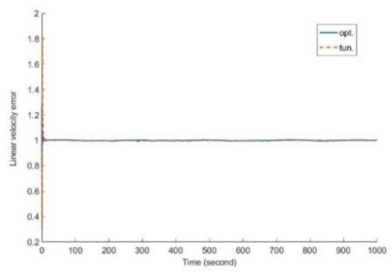
The following figures show the error comparison for the sinusoidal trajectory variables where Figure 12.a shows the actual x error comparison, Figure 12.b shows the actual y error comparison, Figure 12.c shows the actual orientation error comparison, Figure 12.d shows the actual linear velocity error comparison, Figure 12.e shows the actual angular velocity error comparison, and Figure 12.f shows the energy comparison.



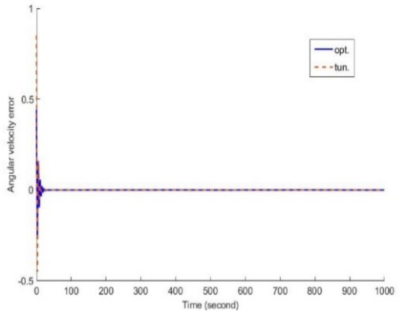


(a): Actual x error comparison (b): Actual y error comparison

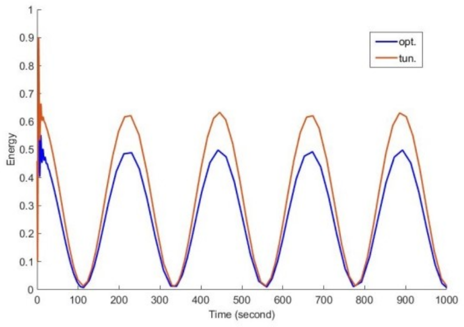




(c): Actual orientation error comparison (d): Actual linear velocity error comparison



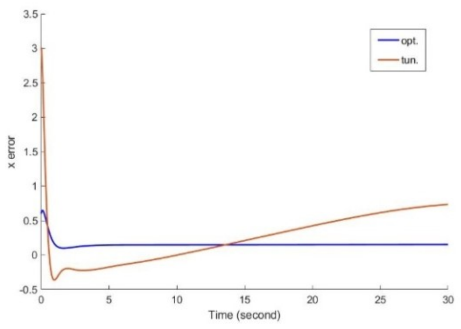
(e): Actual angular velocity error comparison



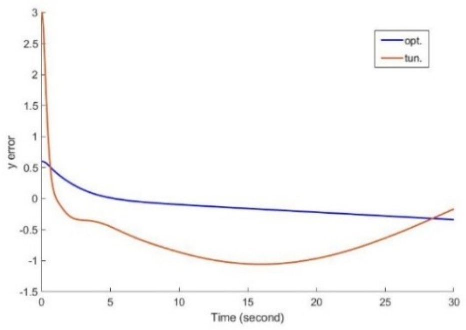
(f): Energy comparison

Figure 12(a), (b), (c), (d), (e), and (f): Sinusoidal trajectory comparison

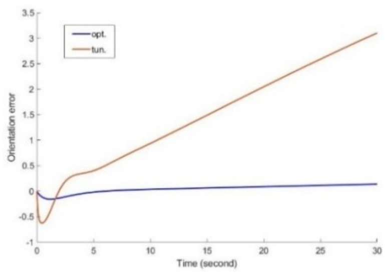
V.ii.c. Straight line trajectory:
The following figures show the error comparison for the line trajectory variables where Figure 13.a shows the actual x error comparison, Figure 13.b shows the actual y error comparison, Figure 13.c shows the actual orientation error comparison, Figure 13.d shows the actual linear velocity error comparison, Figure 13.e shows the actual angular velocity error comparison, and Figure 13.f shows the energy comparison.

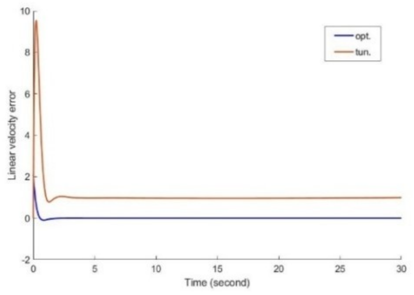


(a): Actual x error comparison

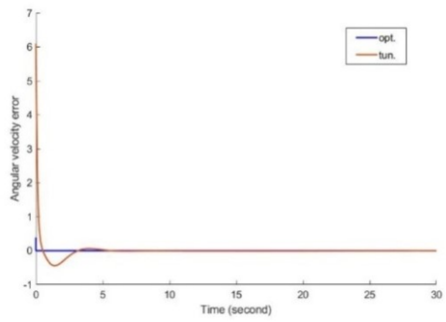


(b): Actual y error comparison

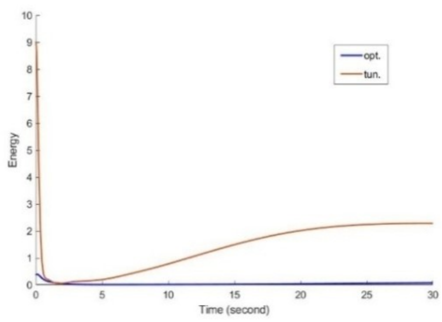




(c): Actual orientation error comparison (d): Actual linear velocity error comparison



(e): Actual angular velocity error comparison

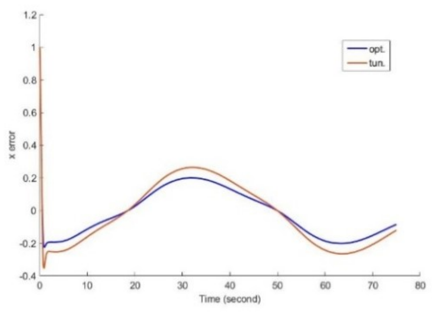


(f): Energy comparison

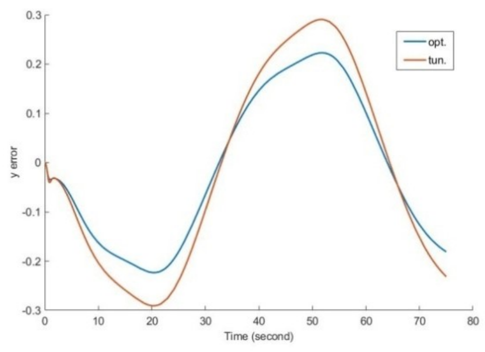
Figure 13(a), (b), (c), (d), (e), and (f): Straight line trajectory comparison

V.ii.d. Elliptical trajectory:

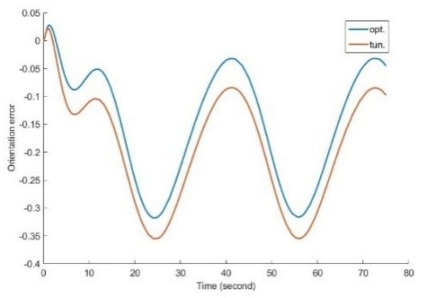
The following figures show the error comparison for the elliptical trajectory variables where Figure 14.a shows the actual x error comparison, Figure 14.b shows the actual y error comparison, Figure 14.c shows the actual orientation error comparison, Figure 14.d shows the actual linear velocity error comparison, Figure 14.e shows the actual angular velocity error comparison, and Figure 14.f shows the energy comparison.

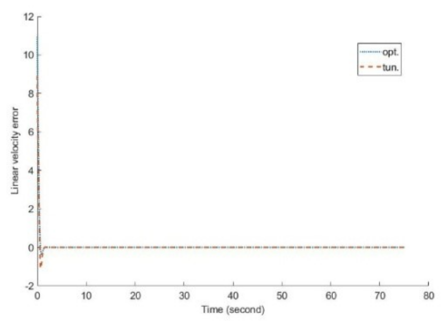


(a): Actual x error comparison

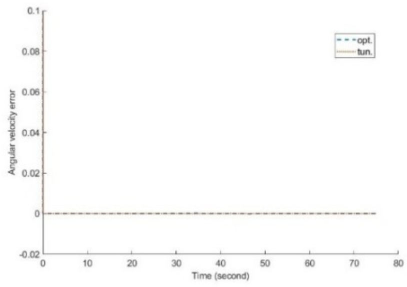


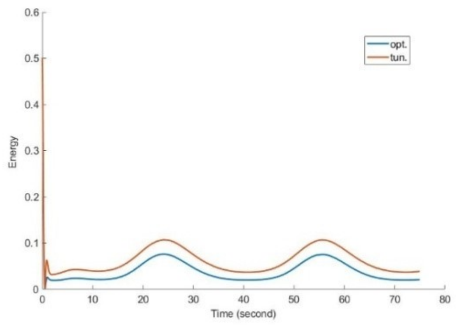
(b): Actual y error comparison





(c): Actual orientation error comparison (d): Actual linear velocity error comparison



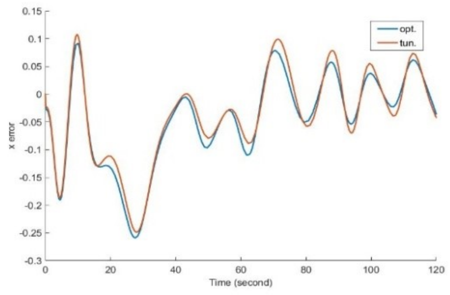


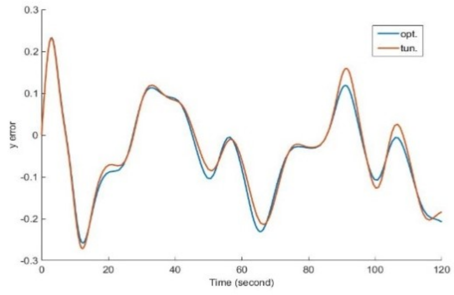
(e): Actual angular velocity error comparison (f): Energy comparison

Figure 14(a), (b), (c), (d), (e), and (f): Elliptical trajectory comparison

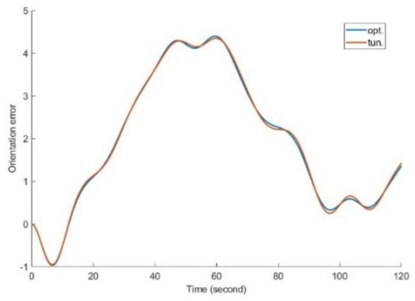
V.ii.e. Infinity trajectory:

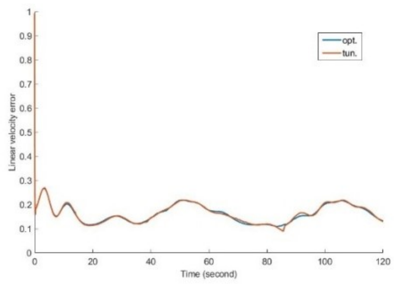
The following figures show the error comparison for the infinity trajectory variables where Figure 15.a shows the actual x error comparison, Figure 15.b shows the actual y error comparison, Figure 15.c shows the actual orientation error comparison, Figure 15.d shows the actual linear velocity error comparison, Figure 15.e shows the actual angular velocity error comparison, and Figure 15.f shows the energy comparison.



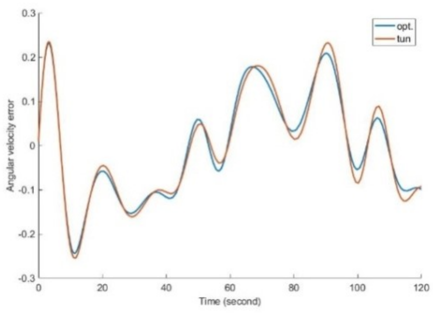


(a): Actual x error comparison (b): Actual y error comparison

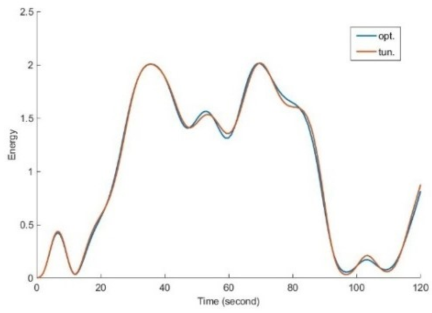




(c): Actual orientation error comparison (d): Actual linear velocity error comparison



(e): Actual angular velocity error comparison



(f): Energy comparison

Figure 15(a), (b), (c), (d), (e), and (f): Infinity trajectory comparison

Table 5: The mean absolute errors of tuned and optimized cases with optimization percent

Mean absolute error	Circular	Sinusoidal	Straight line	Elliptical	Infinity
x opt.	0.1963	0.0199	0.3856	0.2294	0.0320
x tun.	0.6625	0.0498	1.4712	0.2366	0.0257
%	70.369	60.040	73.790	3.0431	19.687
y opt.	0.0132	0.1357	0.3027	0.0146	0.0199
y tun.	0.0741	0.2092	1.2190	0.0043	0.0188
%	82.186	35.133	75.168	70.547	55.27
βopt.	0.0572	0.7055	0.0043	0.0884	1.0497
βtun.	0.4046	0.6086	0.4402	0.1153	1.1071
%	85.862	13.734	99.023	23.330	51.847
v opt.	4.1609	0.9593	0.9452	2.9177	0.3847
v tun.	1.6190	1.0131	1.7304	2.1335	0.3269
%	61.284	5.3104	45.376	26.877	15.024
ωopt.	0.0048	0.1303	0.0724	0.0059	0.0159
ωtun.	0.1001	0.2214	2.4443	0.0054	0.0117
%	95.204	41.147	97.038	8.4745	26.415
Energy opt.	0.1101	0.3800	0.2121	0.1577	0.5518
Energy tun.	0.5290	0.3909	4.6780	0.1640	0.6334
%	79.187	2.7884	95.466	3.8414	12.882

VI. Discussion

It's clear that the suggested controllers accurate the robot trajectory tracking especially with the use of the optimized control parameters. All errors value in the five cases of trajectory tracking are converge zero except the orientation error that appeared in the infinity trajectory due tothe presence of tipping points. It is clear from Table 5that the optimization method made large modification where the maximum optimization percent for position y reached to 82.186% in the circular trajectory and all others maximum optimization percent were appeared in theline trajectory which are reached to 73.79%, 99.023%, 45.376%, 97.038%, and 95.466% for the position (x), orientation error(β), linear velocity (v), angular velocity (ω), and the energy respectively.Figure 15 showed that the optimized results are close to the tuned results due to the optimized gains values which were close to the tuned values.

VII. Conclusion

In this work, backstepping-based controllers have been designed in conjunction with Lyapunov approach to control the robot tracking. MATLAB/Simulink were used to simulate the WMR trajectory tracking for five cases of trajectory. The parameters of the kinematic and dynamic controllers wereoptimized using the particle swarm approach. Simulation results showed excellentperformance and matching between desired and actual trajectory and ensured that the errors converged to zero. Also PSO method were very efficient in minimizing the compound error.

References

I. A. Sofi, B. R. Phanikumar, "An experimental investigation on flexural behaviour of fibre-reinforced pond ash-modified concrete", Ain Shams Engineering Journal. vol. 6, pp: 1133-1142, 2015.

II. Fareh R, Saad M, Khadraoui S, and Rabie T 2016. Lyapunov-based tracking control for nonholonomic wheeled mobile robot. J. International Journal of Electrical and Computer Engineering. V10. PP1042-1047.

III. García-Sánchez J, et al 2017 Mexico. Tracking control for mobile robots considering the dynamics of all their subsystems: experimental implementation.P1-18.

IV. Hadi N 2005 Iraq. Fuzzy control of mobile robot in slippery environment. J. Al-Khwarizmi Engineering Journal. V1.pp41-51.Bakharev T., "Durability of geopolymer materials in sodium and magnesium sulfate solutions", Cement and Concrete Research. vol. 35, no. 6, pp: 1233-1246, 2005

V. Han P. et al 2016 USA. "Path planning for a mobile robot using ant colony optimization and the influence of critical obstacle". C. International Conference on Industrial Engineering and Operations Management Detroit.

VI. Klancar G. et al. 2017 UK, USA. B. "WHEELED MOBILE ROBOTICS"

VII. Leenaa N. and Sajub K.K. 2016 India. "Modelling and trajectory tracking of wheeled mobile robots". J. Procedia Technology. V24. PP538 – 545.

VIII. Maghenem M. et al 2017 France. "Global tracking-stabilization control of mobile robots with parametric uncertainty". J. International Federation of Automatic Control. V50. PP4114–4119.

IX. Nurmaini S. et al 2017. "Differential-Drive Mobile Robot Control Design based-on Linear Feedback Control Law". C. IAES International Conference on Electrical Engineering, Computer Science and Informatics. S755. PP1-7

X. Obaid M, Husain A, Al-Kubati A 2016. Robust backstepping tracking control of mobile Robot Based on Nonlinear Disturbance Observer. J. International Journal of Electrical and computer engineering (IJECE) .V6. PP901-908.

XI. Pandey A. and Parhi D. R. 2017 India. "Optimum path planning of mobile robot in unknown static and dynamic environments using Fuzzy-Wind Driven Optimization algorithm". J. Defence Technology. V13. PP47-58.

XII. Saleh A. L. et al, 2018 Iraq. "Design fuzzy neural petri net controller for trajectory tracking control of mobile robot". J. International Journal of Engineering & Technology. V7. PP1-13

XIII. Shih C, and Lin L 2017 Taiwan. Trajectory planning and tracking control of differential drive mobile robot in a picture drawing application.P1-15.

XIV. Tawfik M, Abdulwahb E, and Swadi S 2014 Iraq. Trajectory tracking control for a wheeled mobile robot using fractional order PID controller. J. Al-Khwarizmi Engineering Journal. V10. PP39- 52

XV. Tzafestas S 2014 Greece. B. Introduction to mobile robot control.

XVI. Younus K. K. and Hadi N. H. 2019 Iraq. "Path tracking and backstepping control for a wheeled mobile robot (WMR) in a slipping environment". C. 3,d International Conference on Engineering Sciences: ICES