

Tracking of a PID Driven Differential Drive Mobile Robot

U. Dinesh Kumar*, M. Nisha and N. Mathivanan

Assistant Professor, Department of Electronics, St. Joseph's College, Trichy, Tamilnadu, India

Guest Lecturer, USIC, Madurai Kamaraj University, Madurai, Tamilnadu, India

Visiting Faculty, Department of Instrumentation & Control Engineering, National Institute of Technology, Trichy, Tamilnadu, India.

Phone Number: +91-9524121838

***Corresponding Author's E-mail:** udinkumar87@gmail.com

Abstract

A PID Controller has been designed and incorporated into the differential drive mobile robot. The mobile robot is built around an ARM7 based microcontroller LPC2129. It includes an odometry unit attached to the rear wheels and ZigBee based RF transceivers. The position and the orientation of mobile robot are estimated using the odometry unit. As a ZigBee based RF transceivers are integrated on mobile robot and remote PC an online tracking and control system is established. In several mobile robotic applications the control systems implemented are Open Loop Control System(OLCS). These OLCS based systems faces uncertainty errors on their trajectory. To overcome such errors a Closed Loop Control System(CLCS) driven robot is discussed in this paper. A firmware including a Proportional-Integral-Derivative (PID) control algorithm is developed. This enables the online velocity tuning mechanism for the robots to drive in user defined trajectory. The PID control algorithm is developed for reducing the initial inertia error. Inertial errors affects the robot's programmed velocity which intrun causes the robot to deviate from the user defined trajectory. The PID based CLCS periodically checks and corrects the individual wheel speed online to place the robot in trajectory. A LabVIEW application program is developed to compute, track the position and orientation of robot online. Experimental tests were conducted to demonstrate the working of the PID control system and the results are presented.

Keywords: PID, OLCS, CLCS, ARM7, ZigBee, LabVIEW.

1. Introduction

PID based control systems are widely used in industries because of its simplicity as it does not require plant modeling and system designing. They are structurally simple and exhibit robust performance over a wide range of operating conditions. In the absence of the complete knowledge of the process these types of controllers are the most efficient of choices. The three main parameters involved are Proportional (P), Integral (I) and Derivative (D) constants. The proportional part is responsible for following the desired set-point, while the integral and derivative part account for the accumulation of past errors and the rate of change of error in the process respectively. Tuning of a PID controller refers to the tuning of its various parameters (P, I and D) to achieve an optimized value of

Article History:

Received Date: Apr. 09, 2017

Accepted Date: Oct. 08, 2017

Available Online: Jan. 04, 2018

the desired response. The basic requirements of the output will be the stability, desired rise time, peak time and overshoot. Different processes have different requirements of these parameters which can be achieved by meaningful tuning of the PID parameters. If the system can be taken offline, the tuning method involves analysis of the step input response of the system to obtain different PID parameters. But in most of the industrial applications, the system must be online and tuning is achieved manually which requires very experienced personnel and there is always uncertainty due to human error. While this method is good for online calculations and it involves some trial-and-error.

PID controller based mobile robots

Mobile robots are designed with integration of both hardware and software for various real world applications. Working with the applications the robots are about to roam around the global frame and create different trajectories to execute the user defined tasks. The way robot has to make motion is pre-programmed by the user in some applications with the integrated embedded system components present with the robot design. Among such situations there are possibilities for the robot to deviate from the programmed trajectory due to the intervention of some non-systematic parameters mostly those errors are introduced in the robots trajectory. This creates the need of tracking robots in the real world applications. From the results it is clear that the robot's velocity is not maintained due to uneven floor conditions and due to some non - systematic errors. This problem exists because the control algorithm adopted in the previous system is basically an OLCS[1]. The mobile robots are supposed to be monitored periodically with the help of sensors with a reference point to what is programmed controlled by the controller at every instant as its trajectory is being tracked.

Abdalla et al (2012) used a differential drive mobile robot equipped with an odometry unit to predict the robots motion for every instant. Further a tracking control of differential-drive wheeled mobile robots using a back stepping-like feedback linearization is also implemented [2-4]. C. S. Mala and S. Ramachandran (2014) designed PID controller based direction control for agriculture robots to monitor the automatic steering and speed control [5]. Gohiya and Chandra Shekhar (2013) found that slight variations in the velocity will create a turning effect in the robot and create a new trajectory. This can be avoided by implementing control loop tuning techniques which are traditionally available and used in various industrial applications. A closed loop feedback algorithm based on digital PID controller has been developed for precise position and speed control of mobile robot [6].

Localization and closed loop motion control of a differential drive mobile robot which is capable of navigating to a desired goal location in an obstacle free static indoor environment with Lyapunov criterion control algorithm was implemented by S k malu and J.Majumdar (2014) [7]. Stephen Armah et al (2014) implemented a MATLAB robot simulator on two wheeled ground mobile robot, Dr Robot X80SV. Also four wheeled robots and robot manipulators trajectory tracking is compared with different autonomous navigation control algorithms based on PID .The general relationships of the PID controller design on the robotic dynamics and the planned trajectory are analyzed and simulations of its closed loop dynamics indicating its effectiveness in fast and accurate trajectory tracking for non linear motion control for industrial robots [8-14].

A robust digital PID controller is proposed by Mummadi, V. (2011) for the H-bridge soft-switching boost converter (HSBC) to ensure load voltage regulation as well as to give robust performance with step loads and source rejection [15]. Park, B. S et al (2010) proposed a simple adaptive control approach for path tracking of uncertain non holonomic mobile robots incorporating actuator dynamics [16,17].

This paper presents the design and implementation of PID control algorithm for controlling the movement of a mobile robot built around LPC2129 microcontroller. The program is developed and fused into the EPROM by which the robot is controlled and tracked with its programmed trajectory. Different trials were conducted with various PID gains to achieve best suitable gain for the robot.

Control loop tuning

Adjusting the controller gain K_C , the reset time T_I , and the derivative time T_D is referred as controller tuning, objective being to get the characteristics of the controller “in tune with” the characteristics of the process.

Trial and error tuning procedure

Today, most controllers are still tuned using the traditional trial and error tuning procedures (sometimes derisively referred to as “knob twiddling”). This approach involves a repeated sequence of the following:

1. Obtain the response of the loop to some change, usually in the set point.
2. Assess the loop behavior relative to its desired behavior.
3. Adjust the tuning coefficients.

The most convenient response to obtain is usually to a step change in the set point, but the response to other changes can be used if desired. Step 1 tunes a proportional - only controller, steps 1 and 2 tune a proportional - integral (PI) controller, and all three steps tune a PID controller.

2. PID control algorithm

The PID algorithm involves three separate constant parameters, and is accordingly sometimes called three-term control: the proportional, the integral and derivative values, denoted P , I , and D . These values can be interpreted in terms of time: P depends on the *present* error, I on the accumulation of *past* errors, and D is a prediction of *future* errors, based on current rate of change.

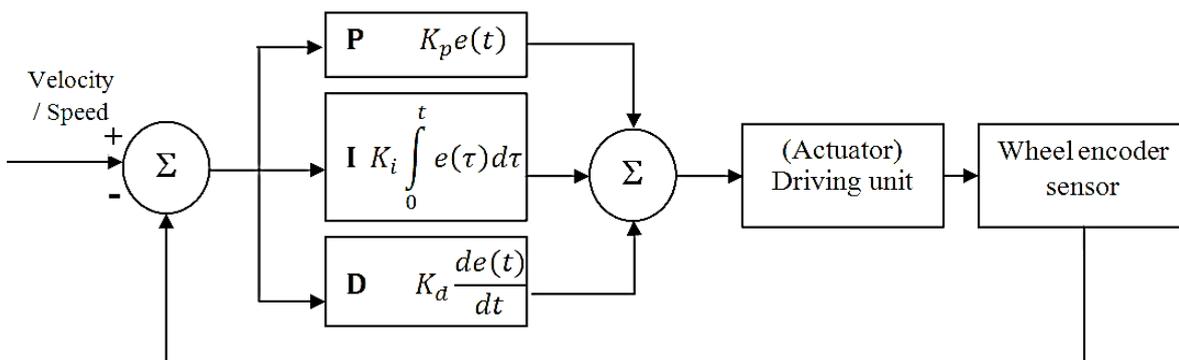


Figure 1: Functional Block diagram of PID controller.

The weighted sum of these three actions is used to adjust the process variable via a control element such as PWM signal supplied to the motor, etc.

The Equation of PID algorithm can be given as

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \tag{1}$$

Where

K_p : Proportional gain, a tuning parameter

K_i : Integral gain, a tuning parameter

K_d : Derivative gain, a tuning parameter

e : Error = Set Point – Process Variable

t : Time or instantaneous time (the present)

Applying Laplace transform to equation (3.1)

$$U(s) = K_P + \frac{K_i}{s} + sK_d \tag{2}$$

Where,

$$\begin{aligned}
 K_p &= K \\
 K_i &= K/dt \\
 K_d &= K * dt
 \end{aligned}$$

For discrete time PID algorithm implementation, z-transform of PID equation is applied,

$$U(z) = \left[K_p + \frac{K_i}{1-z^{-1}} + K_d(1 - z^{-1}) \right] E(z) \tag{3}$$

By rearranging the equation,

$$U(z) = \left[\frac{(K_p+K_i+K_d)+(-K_p-2K_d)z^{-1}+K_dz^{-2}}{1-z^{-1}} \right] E(z) \tag{4}$$

Equation (3.4) can be rewritten as

$$U(z) - z^{-1}U(z) = [K_1 + K_2z^{-1} + K_3z^{-2}]E(z) \tag{5}$$

The above equation if converted to difference equation as,

$$u[n] = u[n - 1] + K_1e[n] + K_2e[n - 1] + K_3e[n - 2] \tag{6}$$

Where

$$K_1 = K_p + K_i + K_d \tag{7}$$

$$K_2 = -K_p - 2K_d \tag{8}$$

$$K_3 = K_d \tag{9}$$

3. Design of mobile robot

Hardware Description

The functional block diagram of the PID controller is shown in Figure 1. The measurement system is built around an odometry unit, a microcontroller (NXP LPC2129), driving unit and a XBee-PRO series-2 ZigBee transceiver with the remote position tracking unit is shown in Figure 2.

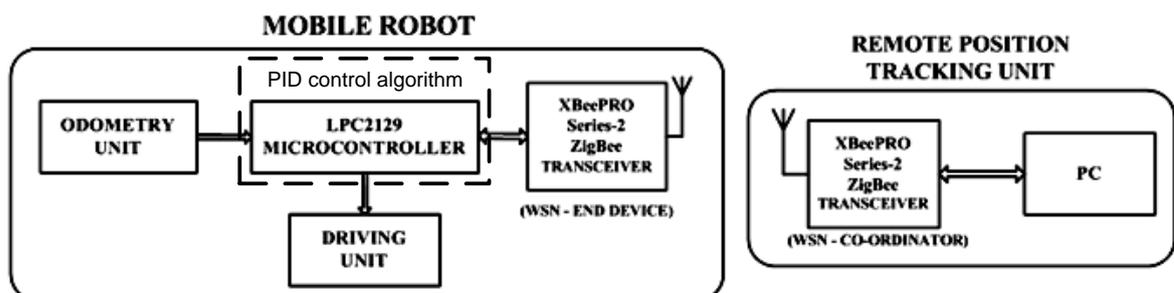


Figure 2: Block diagram of mobile robot and remote tracking unit.

Firmware Description

The firmware for the embedded program is developed on Keil μ Vision4 IDE. The application programming interface (API) for remote tracking unit is developed on LabVIEW.

Embedded firmware:

The embedded program performs the following functions. The flow chart of the embedded program is shown in Figure 4.

Main program:

- i). Initializes the on-chip peripherals UART1, Timer0, EINT 2 & EINT 3, PWM 3, PWM 5 and GPIO pins. Configures GPIO pins for Left and Right wheel motors direction control pins. Configures Vectored Interrupt Controller for interrupts from three sources (Timers, right and left wheel slotted sensors). Configures UART1 for baud rate 38400 bits/s.
- ii). Configures external interrupts EINT2 and EINT3 to rising edge sensitivity.
- iii). Initializes the Timer0 to generate interrupts at 0.5 s intervals
- iv). Initializes two PWM channels of PWM3 and PWM5 to generate two different PWM outputs at two different duty cycles. Configures PWM3 for 93.3% and PWM5 for 73.3% duty cycle.
- v). configures software counters to count time intervals and pulses from right and left wheel slotted sensors (odometry unit).

Function1: UART1 interrupt service routine: Measurement system generates an interrupt when a command is received from the remote tracking unit.

- i). If the received command is "START", it enables Timer0, EINT 2 & EINT 3, PWM 3 and PWM 5.
 - a. Get the measurement data from left wheel slotted sensor ' C_{Ln} ', right wheel slotted sensor ' C_{Rn} ' and time data ' n ' from Timer0.
 - b. Check status
- ii). If Timeflag is set, it transmits the data packet to remote tracking unit via UART1. A 15-bytes packet is generated with the quantized data n , C_{Ln} and C_{Rn} by EINT2, EINT3 and Timer0 ISR routines. The format of the data packet is shown in Figure 3.

Data -1 Time count data n XXX	Data- 2 Space character X	Data -3 Lcount data C_{Ln} XXXXX	Data- 4 Space character X	Data -5 Rcount data C_{Rn} XXXXX
--	--	---	--	---

Figure 3: Format of the data packet.

- i). If Timeflag is reset, it checks the status and go to step (b).
- ii). If the received command is "STOP", it disables Timer0, EINT 2& EINT 3, PWM 3& PWM 5.

Function2: EINT2 interrupt service routine:

- i). Increment the count value of left wheel slotted sensor ' C_{Ln} ' by 1.
- ii). Stores the measurement data in a buffer.

Function3: EINT3 interrupt service routine:

- i). Increment the count value of right wheel slotted sensor ' C_{Rn} ' by 1.
- ii). Stores the measurement data in a buffer.

Function4: Timer0 interrupt service routine:

- i). Increment the Time count value from the interrupt on Timer0 ' n ' by 1.
- ii). Stores the measurement data in a buffer.

Function5: PID interrupt service routine:

- i). Get left wheel count value ' C_{Ln} ' and right wheel count value ' C_{Rn} ' from the buffer.
- ii). Check if set point left wheel count ' S_{Ln} ' and left wheel count value ' C_{Ln} ', set point right wheel count ' S_{Rn} ' and right wheel count value ' C_{Rn} ' is equal or not. If it is not equal compute following manipulation.

$$E_{Ln} = SP_L - C_{Ln} \quad (10)$$

$$E_{Rn} = SP_R - C_{Rn} \quad (11)$$

This is the proportional direct error that got from subtracting the set point count value and feedback count value of slot sensors.

- iii) The PID gain constants are calculated using following equations, where KP_L, KI_L, KD_L and KP_R, KI_R, KD_R are set of constant values as programmed

$$KP_L = K \quad (12)$$

$$KI_L = K/dt \quad (13)$$

$$KD_L = K * dt \quad (14)$$

$$K1_L = KP_L + KI_L + KD_L \quad (15)$$

$$K2_L = KP_L - (2 * KD_L) \quad (16)$$

$$K3_L = KD_L \quad (17)$$

$$KP_R = K \quad (18)$$

$$KI_R = K/dt \quad (19)$$

$$KD_R = K * dt \quad (20)$$

$$K1_R = KP_R + KI_R + KD_R \tag{21}$$

$$K2_R = KP_R - (2 * KD_R) \tag{22}$$

$$K3_R = KD_R \tag{23}$$

The error values are updated for every 0.5 s time interval using the following statement,

$$EL_{n-2} = EL_{n-1} \tag{24}$$

$$EL_{n-1} = EL_n \tag{25}$$

$$ER_{n-2} = ER_{n-1} \tag{26}$$

$$ER_{n-1} = ER_n \tag{27}$$

Now all three errors are multiplied by their gain constants which are set manually with motor characteristics and testing on motor. Now by adding all three errors the actual error quantity is found. This enables a smooth speed control over motor.

$$Uval_{Ln} = Uval_{Ln-1} + (K1_L * EL_n) + (K2_L * EL_{n-1}) + (K3_L * EL_{n-2}) \tag{28}$$

$$Uval_{Rn} = Uval_{Rn-1} + (K1_R * ER_n) + (K2_R * ER_{n-1}) + (K3_R * ER_{n-2}) \tag{29}$$

The above manipulated value of PID output added with PWM signals generated by the microcontroller. Hence by varying this variable the system could tune the PWM signal generated. This inturn tunes the velocity of the mobile robot.

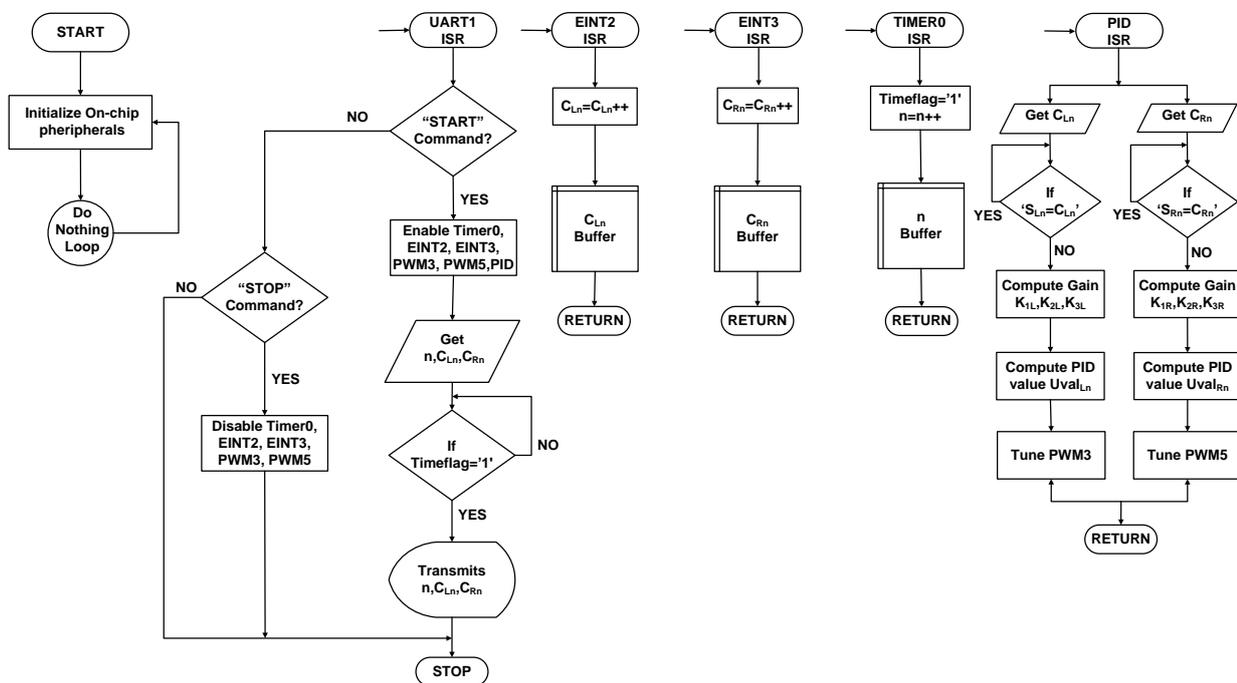


Figure 4: Flowchart of embedded firmware.

LabVIEW application program

A LabVIEW program has been developed to track the mobile robot estimated position at every instant of time. In remote tracking unit, the XBee PRO series-2 ZigBee transceiver receives the time count, left wheel count and right wheel count values. LabVIEW program running on remote PC Instantly computes the estimated position of mobile robot using kinematic algorithm of differential drive. The estimated position is displayed on X-Y graph indicator in real-time. Figure 5 shows the front panel and Figure 6 shows the block diagram of the program.

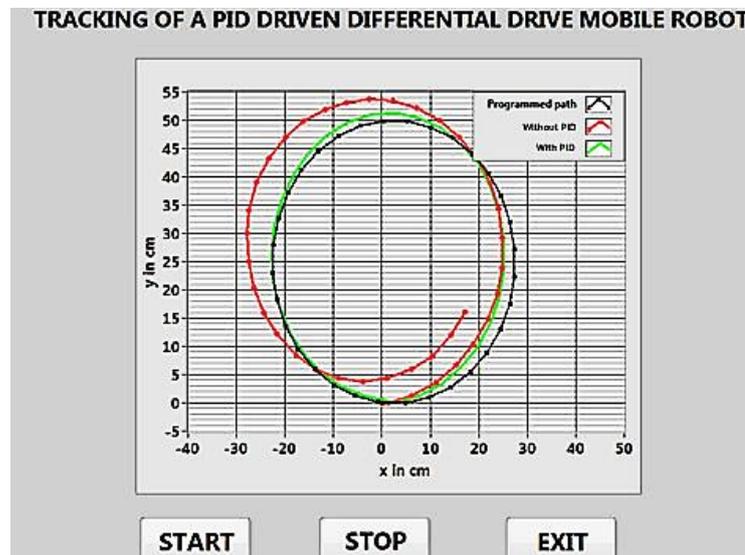


Figure 5: Front panel of the LabVIEW program.

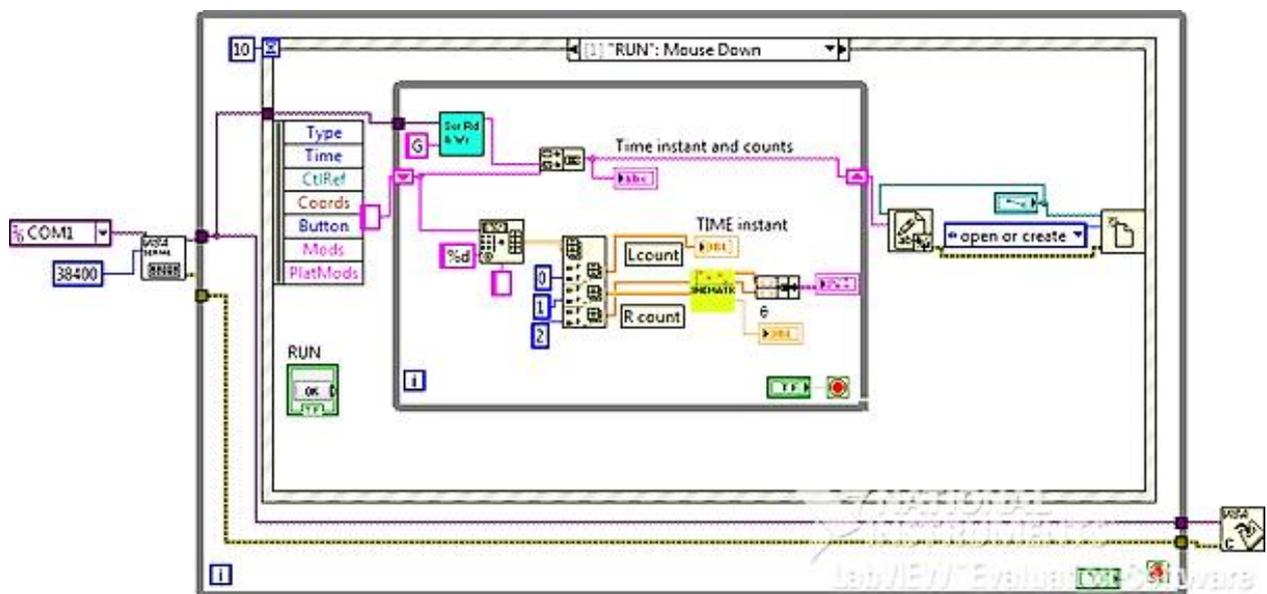


Figure 6: Block diagram of LabVIEW program.

Measurements with the Developed System

Measurements of the estimation of position and linear velocity of the mobile robot driven using PID controller made with LabVIEW program are presented. The improvements in performance of the mobile robot with the implementation of a PID controller based CLCS using Trial and Error tuning technique over the OLCS is discussed further.

The measurement system of odometry unit is mounted on a differential drive mobile robot and the robot is driven in circular trajectory. LPC2129, an ARM 7 based microcontroller, is used for implementing a real time digital PID control algorithm. The test is carried out to study the response of the measurement system for different velocities as given below.

- i). The mobile robot right wheel and left wheel velocities have been programmed by 6.74 and 3.02 cm/0.5s, for these different wheel velocities make the circular trajectory.
- ii). The mobile robot programmed average velocity is 4.88 cm/0.5s and maintained at the same velocity for about 10 s.
- iii). The right and left wheel linear velocities of the robot at every instant as computed and it's displayed as shown in Figure 7.
- iv). The PID gain parameters K_p , K_i and K_d for left wheel in 4 different trials are set with values given in Table 1.

Table 1:Left wheel PID gain parameters.

Trial No.	Lkp	Lki	Lkd
Trial-1	200	1800	350
Trial-2	200	2000	600
Trial-3	200	1800	400
Trial-4	200	1800	100

The Figure 7 shows left wheel velocity for applying varies PID gains using trial and error tuning procedure.

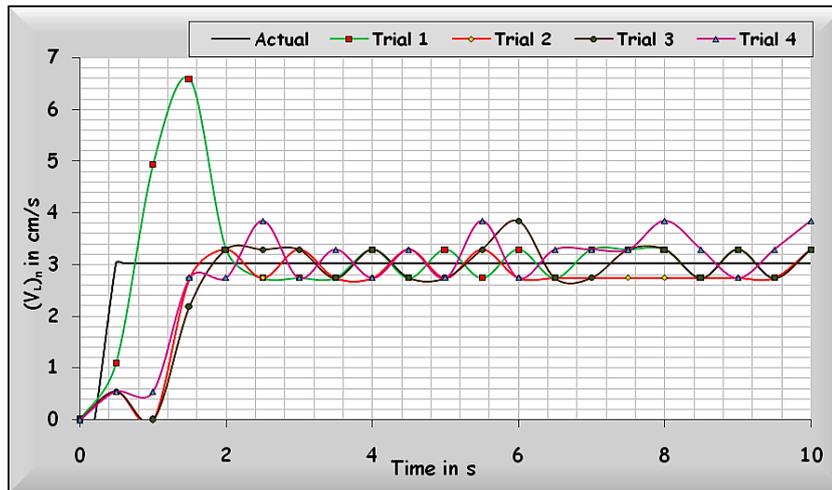


Figure 7: Left wheel's velocity (V_L) profiles with different PID gain parameters

- i). The PID gain parameters K_p , K_i and K_d for right wheel in 4 different trails are set with values given in Table 2.

Table 2: Right wheel PID gain parameters.

Trial No.	Rkp	Rki	Rkd
Trial-1	200	1900	400
Trial-2	200	2000	900
Trial-3	300	1900	600
Trial-4	200	2000	950

The Figure 8 shows Right wheel velocity for applying varies PID gains using trial and error tuning procedure.

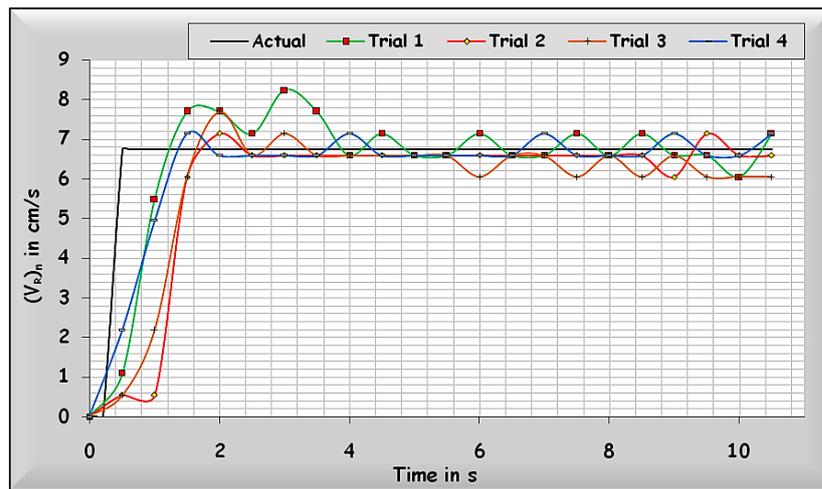


Figure 8: Right wheel's velocity (V_R) profile for different PID gains

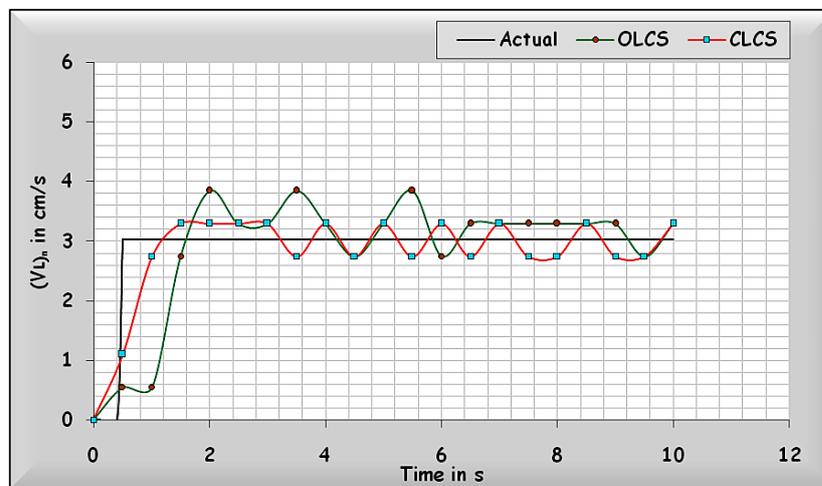


Figure 9: Left wheels' estimated linear velocity vs. actual velocity

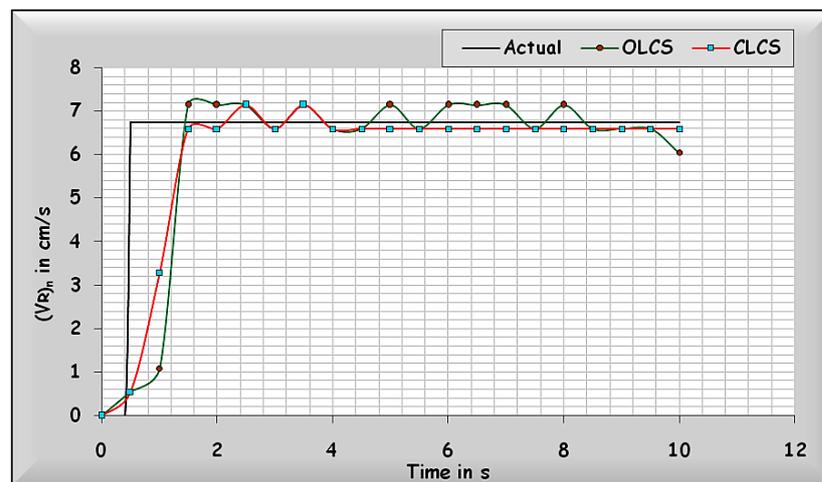


Figure 10: Right wheel's estimated linear velocity vs. actual velocity

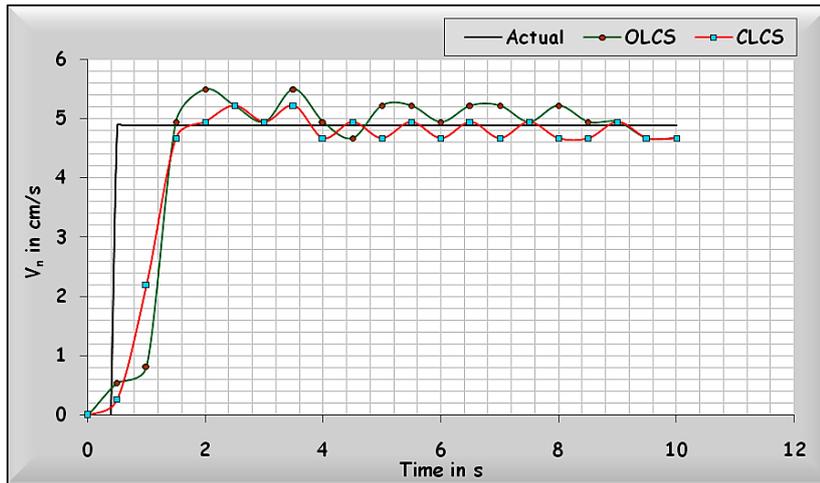


Figure 11: Mobile robots' estimated linear velocity vs. actual velocity

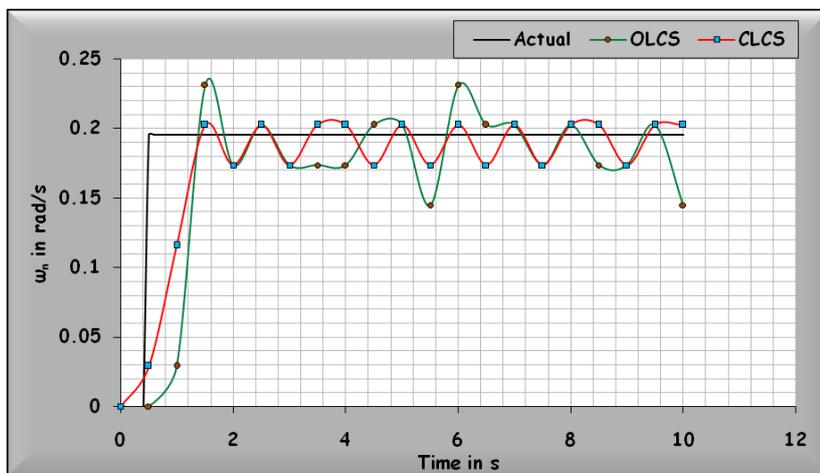


Figure 12: Mobile robots' estimated angular velocity

Results and discussion

Errors in velocities

From the above 3 graphs, it is observed that the behaviour of the CLCS based PID driven mobile robots tracking is satisfactory. It is noticeable the initial inertia error is been gradually reduced in output of CLCS. As the estimated velocities of CLCS and actual velocities are less deviated than the OLCS in graphs are presented. The Mean Square Error (MSE) is calculated between the actual applied velocity and measured signals of odometry linear velocity. MSE of Left wheel velocity, Right wheel velocity and Average velocity has been observed from the estimated $(V_L)_n$, $(V_R)_n$ and V_n and the values are tabulated in Table 3.

Table 3: MSE of linear velocity

Left Wheel Velocity $(V_L)_n$ cm/s		Right Wheel Velocity $(V_R)_n$ cm/s		Average Velocity V_n cm/s	
OLCS	CLCS	OLCS	CLCS	OLCS	CLCS
0.73	0.68	3.44	2.42	1.75	1.38

Table 4: RMSE's of linear velocity

Left Wheel Velocity $(V_L)_n$ cm/s		Right Wheel Velocity $(V_R)_n$ cm/s		Average Velocity V_n cm/s	
OLCS	CLCS	OLCS	CLCS	OLCS	CLCS
0.85	0.82	1.85	1.55	1.32	1.17

From the measurement the following Root Mean Square Error is calculated with the signals of odometry linear velocity. RMSE of Left wheel velocity, Right wheel velocity and Average velocity has been observed from the estimated $(V_L)_n$, $(V_R)_n$, and V_n and the values are tabulated in Table 4.

Table 5 MSE and RMSE of angular velocity

MSE of Angular velocity ($\dot{\theta}$ or ω) rad/s		RMSE of Angular velocity ($\dot{\theta}$ or ω) rad/s	
OLCS	CLCS	OLCS	CLCS
0.003867	0.001899	0.06219	0.04357

MSE and RMSE of Angular velocity has been observed from the estimated ($\dot{\theta}$ or ω) and tabulated.

Experimental setup for position tracking on trajectory

The robot is driven with PID control algorithm on a circular trajectory and tracked from LabVIEW program. This is presented in comparison with the actual trajectory, OLCS and CLCS trajectory tracked results in a single plot shown in Figure 13.

1. Actual Trajectory

A kinematic equation of differential drive mobile robot has been designed and simulated as actual trajectory to compare with the online orientation tracking of robot, (x, y, θ) at 0.5 s interval. The x and y values were plotted on X-Y graph. The actual trajectory for the robot is shown in Figure 13.

2. OLCS- Estimated path using odometry

The robot is driven in a circular path. Tests are conducted to evaluate the performance of OLCS. The Linear velocity is computed using the LabVIEW program. The results are shown in the Figure 13. The X – Y plane graduations were drawn on a flat, smooth floor. The robot was positioned at origin and oriented along the x-axis. The circular path planned for the robot was accomplished by initializing the PWM module to generate PWM outputs at 93.3% and 73.3 % duty cycles. It was found that for these duty cycles the wheels rotate at 49.06rpm and 21.98 rpm speeds and move at 4.88cm/0.5s linear velocities respectively.

3. CLCS- Estimated path using odometry with PID

The CLCS of PID control loop trial and error tuning method is implemented in LPC2129 embedded program. The robot is driven in a circular path. The robot was allowed to travel for more than a minute. The online tracking made with LabVIEW program is displayed on the X-Y graph on the monitor screen at every 0.5s interval is shown in the Figure 13.

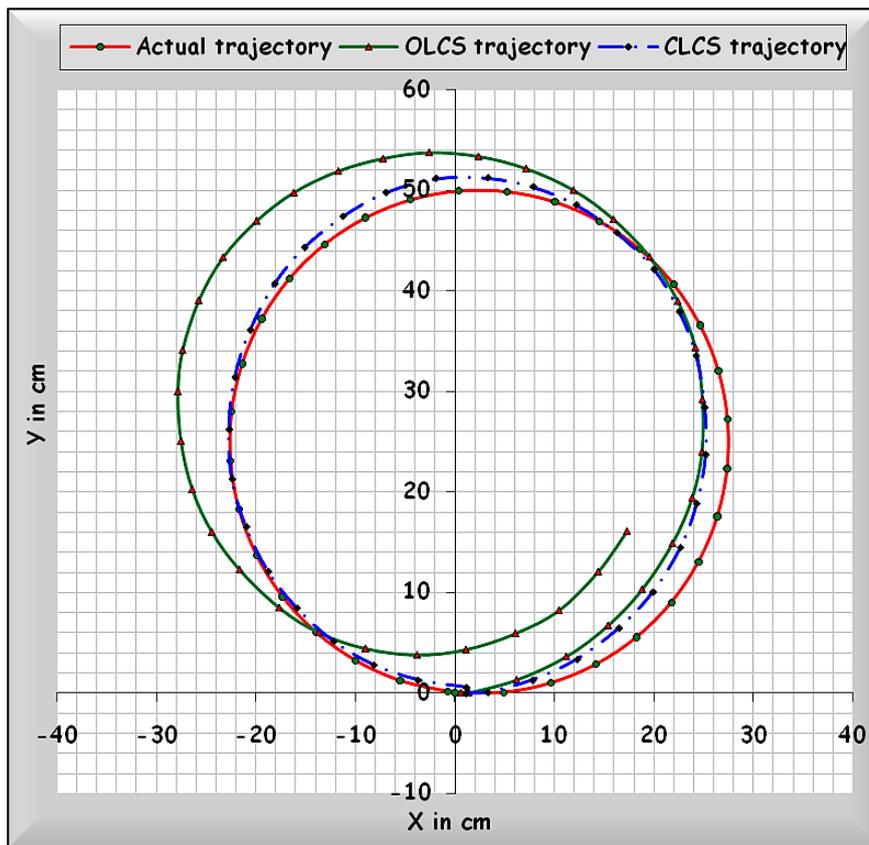


Figure 13: Position estimation on a circular trajectory

Table 5: Position co-ordinate at t = 5 s

Position Co-ordinates (X, Y, θ)					
X in cm		Y in cm		θ in rad	
X_{actual}	26.47	Y_{actual}	31.98	θ_{actual}	0.87
X_{OLCS}	24.81	Y_{OLCS}	29.17	θ_{OLCS}	0.67
X_{CLCS}	25.10	Y_{CLCS}	28.36	θ_{CLCS}	0.73

Conclusion

The PID controller based CLCS applied systems response is comparatively better than OLCS based position tracking. PID controller gain parameters online tuning techniques falls on two main variants such as manual and automatic. Though the trial and error method based implementation is simple, it consumes lot of time and requires several trails to choose the appropriate gains. For every trial the controller parameters need to be modified in the embedded program. Further, this method of tuning does not depend upon any kind of numerical formulas to calculate the gain. Whereas the Auto-tuning techniques works with a defined formulas derived by Ziegler and Nicholas [18-20]. The wheel slippage is another chance of missing the wheel rotation counts by which the errors gets involved in the tracking of mobile robot with its position estimation. To overcome this issue in future, multiple interial sensors can be integrated with the measurement system. Thus the mobile robot can be incorporated with an auto-tuning controller and sensor fusion algorithms.

References

- [1]. U. Dinesh kumar, M. Nisha and N. Mathivanan, (2014). Design of Differential Drive Mobile Robot and Remote Tracking of Robot using Odometry and WSN Nodes, *International Journal of Mechatronics, Electrical and Computer Technology*, 4(11), 384- 399.
- [2]. T. Y. Abdalla and A.A. Abdulkarem, (2012). PSO-based Optimum Design of PID Controller for Mobile Robot Trajectory Tracking. *International Journal of Computer Applications*, 47(23), 30-35.
- [3]. D. Chwa, (2010). Tracking control of differential-drive wheeled mobile robots using a backstepping-like feedback linearization. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 40(6), 1285-1295.
- [4]. A. A. Mahfouz, A.A. Aly and F. A. Salem, (2013). Mechatronics design of a mobile robot system. *International Journal of Intelligent Systems and Applications*, 5(3), 23.
- [5]. S. Mala and C. Ramachandran, (2014). Design of PID Controller for Direction Control of Robotic Vehicle for Agricultural Implements. *Global Journal of Researches In Engineering*, 14(3).
- [6]. S. K. Malu, and J. Majumdar, (2014). Kinematics, Localization and Control of Differential Drive Mobile Robot. *Global Journal of Research and Engineering-GJRE-H*, 14(1).
- [7]. Stephen Armah, Sun Yi and Taher Abu-Lebdeh.,(2014).Implementation of Autonomous Navigation Algorithms On Two-Wheeled Ground Mobile Robot. *American Journal of Engineering and Applied Sciences*. Volume 7, Issue 1 Page 149-164.
- [8]. F. A. Salem, (2013). Refined models and control solutions for mechatronics design of mobile robotic platforms. *Estonian Journal of Engineering*, 19(3), 212-238.
- [9]. Q. Xu, J. Kan, S. Chen and S. Yan, (2014). Fuzzy PID based trajectory tracking control of mobile robots and its simulation in simulink. *International Journal of Control and Automation*, 7(8), 233-244.
- [10]. C. Urrea and J. Muñoz,. (2015). Path Tracking of Mobile Robot in Crops.*Journal of Intelligent & Robotic Systems*, 80(2), 193-205.
- [11]. A. Salehi, F. Piltan, M. Mousavi, A. Khajeh, and M.R. Rashidian, (2013). Intelligent Robust Feed-forward Fuzzy Feedback Linearization Estimation of PID Control with Application to Continuum Robot. *International Journal of Information Engineering and Electronic Business*, 5(1), 1.
- [12]. H. Zhang, (2010). PID Controller Design for A Nonlinear Motion Control Based on Modeling the Dynamics of Adept 550 Robot. *International Journal of Industrial Engineering and Management*, 1(1).
- [13]. V. Mummadi, (2011). Design of robust digital PID controller for H-bridge soft-switching boost converter. *Industrial Electronics, IEEE Transactions on*, 58(7), 2883-2897.
- [14]. E. M. Jafarov, M. A. Parlakçi, and Y. I Stefanopoulos, (2005). A new variable structure PID-controller design for robot manipulators. *Control Systems Technology, IEEE Transactions on*, 13(1), 122-130.
- [15]. B. S. Park, S. J. Yoo, J. B. Park and Y. H. Choi, (2010). A simple adaptive control approach for trajectory tracking of electrically driven nonholonomic mobile robots. *Control Systems Technology, IEEE Transactions on*, 18(5), 1199-1206.
- [16]. K. Shojaei, A. M. Shahri, A. Tarakameh and B. Tabibian, (2011). Adaptive trajectory tracking control of a differential drive wheeled mobile robot.*Robotica*, 29(3), 391-402.
- [17]. Smith, Cecil L. *Practical process control: tuning and troubleshooting*. John Wiley & Sons, 2009.
- [18]. C. D. Johnson, (1993). *Process control instrumentation technology*. Prentice Hall PTR.
- [19]. S. K. Singh, (2004). *Computer-Aided Process Control*. PHI Learning Pvt. Ltd.
- [20]. P. Cominos and N. Munro, (2002, January). PID controllers: recent tuning methods and design to specification. In *Control Theory and Applications, IEE Proceedings-* (Vol. 149, No. 1, pp. 46-53). IET.

Author(s)



U. Dinesh Kumar received his M.Sc, degree in Electronics and Instrumentation from Madurai Kamaraj University, India, and B.Sc Electronics and Communication from Madurai Kamaraj University, India. He is currently a Assistant Professor with the Department of Electronics, St. Joseph's College, Tirchirappalli, Tamil Nadu, India. Areas of his research interests include Robotics, Instrumentation, real-time embedded and distributed systems, intelligent systems, condition monitoring, and mechatronics.

email: udinkumar87@gmail.com

Phone Number: +91-9524121838



M. Nisha pursuing her M. Phil degree in Electronics and Communication from Alagappa University, Tamil Nadu, India, received her M.Sc, degree in Electronics and Instrumentation from Madurai Kamaraj University, India, and B.Sc Electronics and Communication from Madurai Kamaraj University, India. she is currently a Guest Lecturer with the University Science Instrumentation Centre(USIC), Tamil Nadu, India. Areas of her research interests include Instrumentation, Digital Image processing and Digital Signal processing, real-time embedded and environment monitoring, Robotics and mechatronics.

email: braveynisha@gmail.com

Phone Number: +91-8760368259



Retd. Prof. Dr. N. Mathivanan received his Doctoral degree in Instrumentation from Madurai Kamaraj University, Tamil Nadu, India, in the year of 1998. M.Tech, degree in Instrumentation Technology from Indian Institute of Science, Bengaluru, India, and M.Sc Applied Physics from National Institute of Technology, Tirchy, India. He is currently a Visiting Professor with the Department of Instrumentation & Control Engineering, NIT, Trichy, India. He have served as Professor and Director of University Science Instrumentation Centre (USIC), Madurai Kamaraj University, Tamil Nadu, India. His research interests are Design of Embedded Systems for Automobile Automation, Building Automation, Environmental Parameters monitoring, Design of Computer based Instrumentation for Lab applications, Design of data acquisition systems compatible to several standard interfaces, Robotics and mechatronics. The author of two popular books :

1. "Microprocessors, PC Hardware and Interfacing", N. Mathivanan, PHI Learning (2003), ISBN: 978-81-203-2317-9, No. of Pages: 519, (XI Print – 2015).
2. "PC Based Instrumentation: Concepts and Practice", N . Mathivanan, PHI Learning (2007), ISBN: 978-81-203-3076-4, No. of Pages: 683, (V Print – 2015).

email: mathivanan@nitt.edu, nmvanan@yahoo.com

Phone Number: +91 9443829142