ISSN 2278-3091
Volume 8, No.5, September - October 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse12852019.pdf
https://doi.org/10.30534/ijatcse/2019/12852019

Genetic Algorithm Optimization and Implementation of Velocity Control PI Controller for Cart Follower Application

A.A.M Zahir[1], S.S.N Alhady[2*], W.A.F.W Othman[3], Zhiling Low[4], A.A.A Wahab[5],
[1]School of Electrical & Electronic Engineering, Universiti Sains Malaysia, Malaysia, afiq.mzahir@gmail.com
[2*] School of Electrical & Electronic Engineering, Universiti Sains Malaysia, Malaysia, sahal@usm.my
[3] School of Electrical & Electronic Engineering, Universiti Sains Malaysia, Malaysia, wafw_othman@usm.my
[4] School of Electrical & Electronic Engineering, Universiti Sains Malaysia, Malaysia, zhilingusm@gmail.com
[5] School of Electrical & Electronic Engineering, Universiti Sains Malaysia, Malaysia, aeizaal@usm.my

ABSTRACT

One of the main important aspect in designing a cart follower for wheelchair user is regulating the velocity of the cart itself so that it follows the wheelchair users. Two of the most important performance parameters are overshoot percentage and settling time. Small overshoot percentage is important so that there is no sudden rise in velocity that could reduce the lifespan of hardwares. Faster settling time is needed as the cart need to adapt with various speed of wheelchair as the cart need the shortest time to achieve steady state condition. Genetic Algorithm tuning method by using ITSE error criterion is used in optimizing the PI controller. In real application, GA yields the best performance in settling time and overshoot percentage, 31.96% and 13.63% better than AMIGO, the second best in these performance parameters. GA produce 140.135% better rise time than AMIGO. Eventhough SIMC and ZN is better in terms of rise time, oscillatory responses and huge overshoot percentage which is 31.73% and 180.75% respectively make both of the tuning methods are unfit in this application. Therefore, GA tuning methods is determined to be the best to be used in the application of velocity control PI controller for cart follower.

Key words : Brushed DC Motor, Cart Follower, Genetic Algorithm Optimization, PI Controller;

1. INTRODUCTION

PID controller is popular due to its simplicity and fulfilling performance [1]. This controller has been used to improve the transient response and the steady state error of the system [1]. The cart follower system is designed for the wheelchair user that faced difficulties when carrying their luggage. DC motor which has been used widely in homes or industries and usually driven by direct current had been used to drive the cart follower. Thus, in term of the safety and performance purpose of the cart follower, the velocity control of the brushed DC motor has become a focused aspect of this paper.

The objective of this paper is to compare the performance of the cart follower after implementing different PID tuning methods in real life application. The cart follower is designed to fulfill wheelchair users needs to carry luggage. The prototype is developed to achieve the average speed of of wheelchair which is 0.5m/s. The velocity control PI controller is designed based on the requirements of the cart that need to catch up with the speed of cart under various unexpected disturbance such as road surface friction, payload and road inclination angle. The velocity controller needs to yields minimum settling time, in order to adapt with various speed of wheelchairs. The controller also need to produce minimum overshoot so that there is no unnecessary spike in velocity that will cause the luggage to be toppled down.

PID tuning methods involved are Genetic Algorithm Optimization, Ziegler-Nichols (ZN), Chien-Hrones-Reswick (CHR), Skogestad Internal Model Control (SIMC) and Approximate M - constrained Integral Gain Optimization (AMIGO). These PID tuning methods can be used to control the velocity of the brushed DC Motor [2].

The earliest ZN tuning method is known to introduce by Ziegler, Nichols and Rochester in 1942 [3]. The proportional term, KP is usually contributing to the overall control factor of the system, the integral term, KI is usually contributed to steady-state error improvement and the derivative term, KD contributed to transient response improvement [4].

CHR tuning method is a modified from Ziegler-Nichols tuning method and proposed by Chien, Hrones and Reswick in 1952 to have better control to the overshoot criteria [2]. The other famous tuning methods which have been widely used are SIMC and AMIGO tuning methods. These tuning methods can be used for improving the disturbance rejection properties [5].

SIMC yields huge amount of overshoot percentage and produce oscillatory response in second order systems of Interacting Spherical Tank System [6]. AMIGO and CHR produce big percentage of overshoot and longer settling time compare with GA and PSO [7].

Metaheuristic tuning methods are widely being used recently to tune the controller [8-12]. There are various type of metaheuristic tuning methods such as Genetic Algorithm,

Particle Swarm Optimisation, Neural Network, Fuzzy, and Bacterial Foraging.

The PID tuning methods are simulated in MATLAB. The performance criteria such as rise time, overshoot, settling time and steady-state error are determined from the closed-loop response. However, since the derivative term is usually very sensitive to noise in practical, the noisy factor has led the PI controller to be used [13]. The simplicity and easy to tune characteristics of PI control has made it famous to be used in motor speed control [13]. The performance of the implementation of different PI tuning methods are being compared and the the fittest tuning methods is evaluated based on the step response and performance parameters evaluation.

2. METHODOLOGY

2.1 Cart Follower's Design and Specification

The cart follower is designed to follow wheelchair user by using colour tracking technique where Pixy CMUcam5 is used as a sensor. The cart follower is equipped with 3 front and rear ultrasonic sensor as an obstacle avoidance mechanism. The cart follower is expected to be able to withstand 70 kg payload with velocity of 0.5m/s. The robot is driven by brushed DC motor and the velocity is controlled by using PID controller that received feedback velocity from a rotary encoder.

3D drawing of cart follower is designed by using SolidWork software. Figure 1 depicts isometric view of the cart follower drawn by using CAD software. The Ackermann configuration is adopted to be used in the cart follower application, where the front tires is used for cornering and rear tires is fixed in angle. Figure 2 shows ortographic drawing of cart follower where the cart follower has the dimension of 800(L) x 646 (W).

As shown in the bill of materials as in figure 3, the cart followers electronics component includes 6 unit of ultrasonic sensors, 1 unit of colour tracking CMUcam 5 camera, 1 unit of servo motor for connering, 1 unit of rear driven transaxle motor, 1 unit of motor driver, and Arduino Mega 2560.

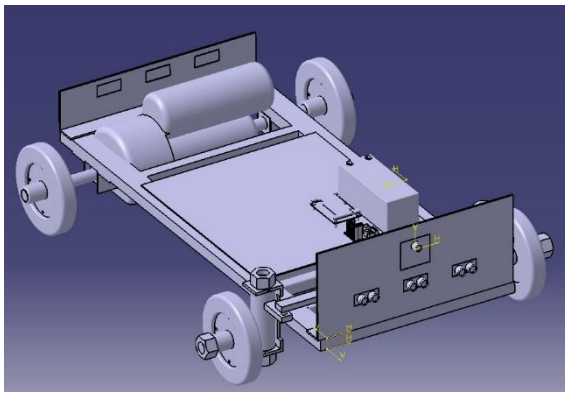


Figure 1: 3D Drawing of Cart Follower

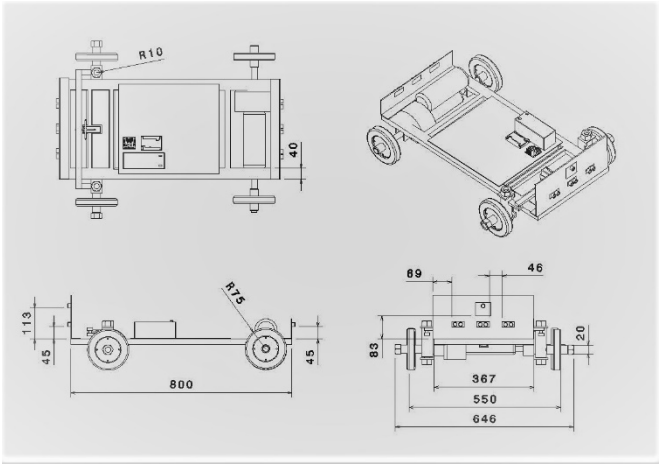
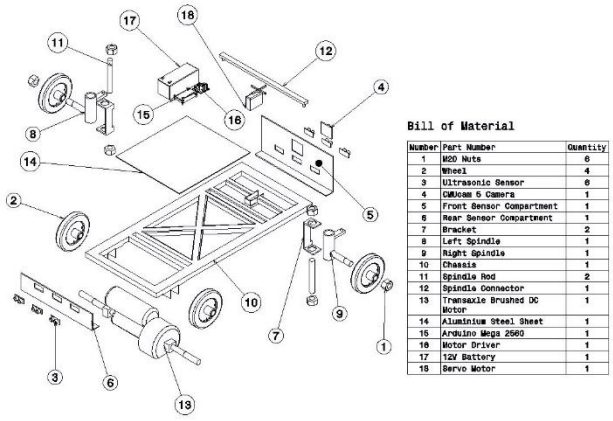


Figure 2: Ortographic Drawing of Cart Follower



Bill of Material

Number	Part Number	Quantity
1	M20 Nuts	6
2	Wheel	4
3	Ultrasonic Sensor	6
4	CMUcam 5 Camera	1
5	Front Sensor Compartment	1
6	Rear Sensor Compartment	1
7	Bracket	2
8	Left Spindle	1
9	Right Spindle	1
10	Chassis	1
11	Spindle Rod	2
12	Spindle Connector	1
13	Transaxle Brushed DC Motor	1
14	Aluminium Steel Sheet	1
15	Arduino Mega 2560	1
16	Motor Driver	1
17	12V Battery	1
18	Servo Motor	1

Figure 3: Bill of Materials of Cart Follower

The main focusing area for this paper is on designing PI velocity controller to drive the transaxle motor. The motor used in the cart follower is PPSM63L-01 as shown in figure 4. The brushed DC motor could yield the output rpm of 4700 rpm and 10.95Nm torque. The plant identification is done by conducting a bump test where the velocity datas is collected by using a rotary encoder over a specific time. The velocity datas is then inserted in MATLAB to obtain the plant transfer function in frequency domain. This is based on authors work in [14] that elaborate on system identification method and kinematics equations of cart follower. The transfer function of the motor is as follow :

$$G(s) = \frac{10.01}{s^2 + 2.553s + 10.91} \quad (1)$$

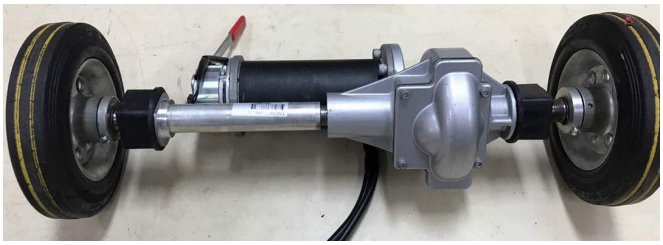


Figure 4: PPSM63L-01 Transaxle Brushed DC Motor

2.2 Controller Selection

PID and PI controller is being considered to be used in the application. PID controller includes derivatives component that could minimize rise time and could act as a damper to the system, however it could amplify noise that could yields excessive output from the controller. PI controller in the other hand is simpler to tune but lacks derivative component that could stabilize the plant and reducing the time for the controller in minimizing the error. Therefore, both PID and PI controller is evaluated in real application in order to observe the controller performance under additional disturbance, that is surface friction and chassis's weight.

The type of controller used is important so that it satisfied the application's requirement of cart follower which involves unexpected availability of disturbance such as surface friction, payload weight, and road inclination angle. PID and PI controller tuned by using AMIGO method is compared in real life application in order to evaluate the suitableness of the controller in the cart follower application. AMIGO is chosen as a fixed tuning method to evaluate both controller as it is simple to tune as it is a non-metaheuristic tuning method that involves no training or optimization process. Firstly, the plant is tuned by using PI and PID controller by using AMIGO tuning. The step response for the new plant including the controller is evaluated. The gains tuned is then inserted into the hardware, coded in Arduino Mega 2560. The velocity of the cart moving on indoor floor that have fixed surface friction, is then recorded for 15 seconds for both PID and PI controller by using a rotary encoder. The performance of both controller is then compared by using the step response plotted from the velocity data. The performance parameters that is being evaluated is rise time, overshoot, settling time and the overall step response graph, in order to observe if there is any fluctuations in the velocity reading.

Based on the experiment data, PID controller produce fluctuate readings, therefore PI controller is chosen to be used. Details explanation could be seen in results and discussions section.

2.3 Genetic Algorithm Optimization of PI Controller

Figure 5 represent the block diagram of the system. PI controller acts as compensator to the plant. Genetic algorithm optimized KP and KI values through offline training. The controller calculate the input needed for the plant to operate. The output of the plant is velocity that is recorded by a rotary encoder. The error is calculated by substracting the current velocity reading by target velocities.

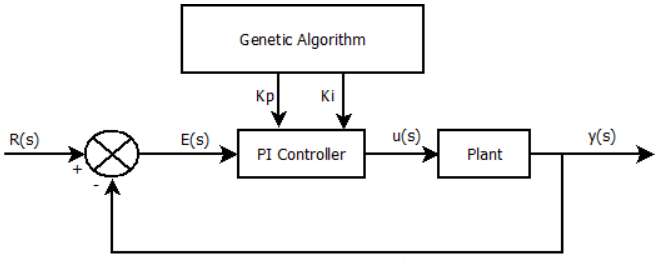


Figure 5: Block Diagram of the Process

PI controller consists of proportional gain, KP and integral gain, KI. Therefore GA optimization is done to find the most optimum values of KP and KI. The upper and lower boundaries values for both proportional and integral gains is set to [2 5] and [0 0] respectively. The rational of those values is based on the KP and KI values obtained from classical formulae tuning methods, Ziegler-Nichols,SIMC, CHR, and AMIGO.

Selection of boundaries' values is important in order to reduce optimization time. The initial population is set to 50 populations due to small number of output parameters. The crossover fraction and elite count selected is 0.8 and 2.5 respectively. Table 1 describe the properties used in GA tuning.

Table 1: GA Properties

Parameters	Details
Population Size	50
Population Type	Double Vector
Lower Bound	[0 0]
Upper Bound	[2 5]
Initial Range	[-10 10]
Selection Function	Stochastic Uniform
Elite Count	2.5
Crossover Fraction	0.8
Stopping criteria	Average change of objective function values for last 50 generations < 0.000001
Maximum Generation	300

Selection of objective function is important in order to obtain good optimum values from metaheuristic tuning methods. Integral Time Squared Error, ITSE error criterion is selected to be used as the objective function in GA optimization of PI controller. GA evaluates the population based on ITSE error criterion, the smaller number of cost function value, the smaller the error, which define the fitness of the solution. Error signal, e(t) is integrated over time to achieve minimum error. The formulation of objective function is as in equation 2.

$$f(K_P,K_I)=\int_0^\tau te(t)^2dt \qquad (2)$$

GA process started by the creation of initial population. The initial population is then evaluated to check whether it meets stopping criterion, which is the values of cost function values is stagnant for 50 generations. If the simulation failed to achieve desired stagnant generations, the simulation stopped at 300 generations. After the evaluation of populations, if the stopping criterion is not met, the fittest chromosomes are selected undergoes crossover and mutation process. The crossover and mutation process produced new offsprings and the new population is reevaluated if it satisfies stopping criterion. The process is represented in figure 6. Authors' previous work in [15] had described the GA process in obtaining optimum values by using different error criteria by using simulation.

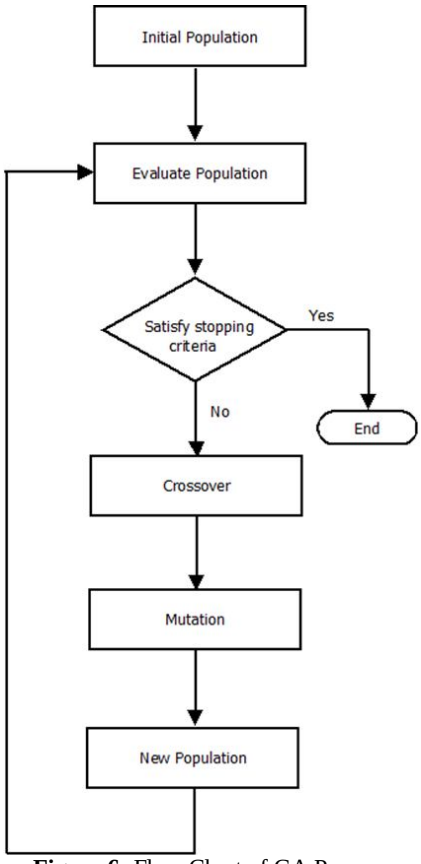


Figure 6: Flow Chart of GA Process

2.4 PI Controller Implementation on Cart Follower Application

Figure 7 depicts overall flow of the program. The flow started with rotary encoder reading the velocity of the cart. The controller evaluate whether the target velocity is met. At stationary, the reading of the encoder would be 0 m/s, Arduino Mega 2560 compute required PWB based on the gains trained offline. The controller sent signal to MD30C motor driver that regulate the speed of the motor in terms of Pulse Width Modulation (PWM). The brushed DC motor rotated and resulted in the cart movement. The rotary encoder reads the velocity at the sampling rate of 250ms. The cart maintained its speed if the targeted velocity is met. Figure 8 shows hardware implementation pictures of cart follower.

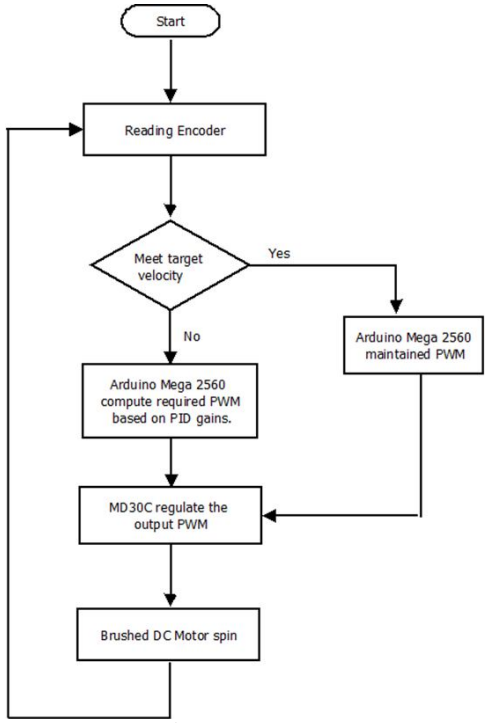


Figure 7: Flow Chart of PI Controller Implementation on Cart Follower

Figure 8: Prototype of Cart Follower

3. RESULTS & DISCUSSIONS

3.1 Controller Selection

Table 2 and table 3 show value of gains obtained from SIMC and AMIGO tuning for controller validation and selection. These gains is inserted into coding and the real application performance is evaluated to determine which controller is more suitable in this cart follower application.

Table 2: Proportional, integral and derivative terms for SIMC and AMIGO PID tuning methods

Tuning Methods	K_P	K_I	K_D
SIMC	1.7183	4.4184	0.0895
AMIGO	1.5189	6.1682	0.0846

Table 3: Proportional and Integral term for SIMC and AMIGO PI tuning methods

Tuning Methods	K_P	K_I
SIMC	1.4442	4.4180
AMIGO	0.5994	2.0727

Figure 9 and figure 10 depicts the implementation performance comparison between PID and PI controller for SIMC and AMIGO tuning method. Both tuning methods indicates that PI controller is more suitable in the real application due to PID controller shows fluctuate readings of velocity. The first interval of 6 seconds in step response graph for both tuning methods for PID controller indicates oscillation and fluctuations in readings. The readings for SIMC PID tuning shows fluctuations and the error is 100% at t=1s and t=4s. For AMIGO PID tuning the error reach until 80% difference from target velocity at t=2.2s. The steady state region for both SIMC and AMIGO PID shows bigger fluctuations compare with PI tuning methods.

Contrary, SIMC and AMIGO PI tuning methods show better response compare with PID controller. The fluctuations for PID tuning occurs due to the availability of disturbance such as road surface friction and chassis's weight. The derivatives component that is very sensitive to noise results to the fluctuations in velocity readings. Therefore, based on the step response analysis, PI controller is chosen to be evaluated and compared further by using GA optimization, CHR and ZN for the application purpose.

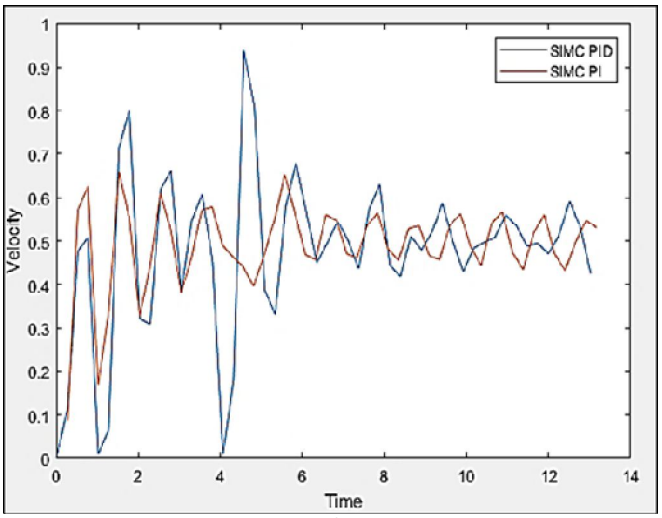


Figure 9: Step Response of SIMC PID and PI Tuning

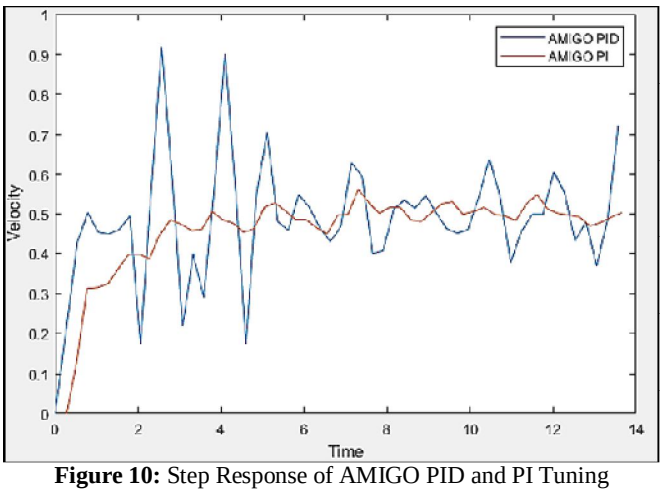


Figure 10: Step Response of AMIGO PID and PI Tuning

3.2 Analysis of Various PI Tuning Methods in Simulation and Real Application.

Table 4 shows the performance indicators results of all tuning methods in MATLAB simulation. In terms of rise time, ZN is the best with 0.213s followed by SIMC, CHR, GA and AMIGO. Contrary for the overshoot, AMIGO is the best with 10.91%, 4.97% better than GA. Based on the results, the methods that produce shorter rise time tend to produce big overshoot. As stated in the introduction, the effect of overshoot is undesirable in the cart follower application as it makes abrupt spike and drop in velocity when there is disturbance acted onto the system. This could lead to shorten lifespan of actuators and hardwares, and migh topple the payload on the cart. Another important parameters is settling time, where GA is the best with 3.964 seconds, 26.24% faster than second best tuning methods AMIGO.

In terms of steady state error, GA is the best with 0.000078, 178.9% better than CHR and 180.22% better than AMIGO. Based on simulation, it could be concluded that GA performed the best with balance results in key indicators. Although GA and AMIGO is among the worst in terms of rise time, but both of the tuning method are the best in terms of overshoot.

Figure 11 shows step response of all tuning methods. It could be clearly seen that ZN oscillate the most compare with other tuning methods. Therefore ZN could be excluded to be used in the application. In order to see the response of tuning methods clearly, figure 12 is plotted without SIMC and ZN response as both of the methods yields too much oscillation due to both tuning methods produce big numbers in integral gain, KD that lead to oscillation.

It could be seen in figure 12 the best response is from GA and AMIGO. GA have 44.9% better rise time, but AMIGO produce less 4.06% overshoot. It could be concluded that AMIGO and GA have small difference in overshoot percentage performance and quite significant difference in rise time performance.

Table 4: Performance for the PI Tuning Methods in Simulation

Tuning Method	Rise Time(s)	Overshoot (%)	Settling Time(s)	Steady-state error
ZN	0.2130	55.30	12.13	0.004100
SIMC	0.2898	41.17	9.728	0.002300
CHR	0.2895	30.06	5.610	0.001400
AMIGO	0.4967	10.91	5.161	0.001500
GA-ITSE	0.3145	14.97	3.964	0.000078

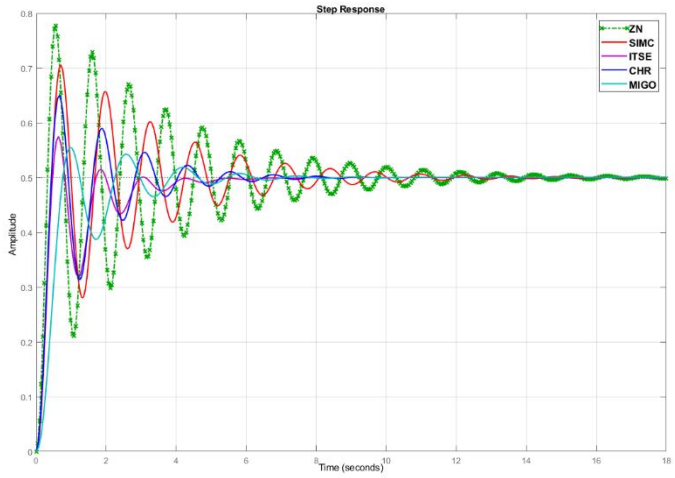


Figure 11: Step Response of Various Tuning Methods – Simulation

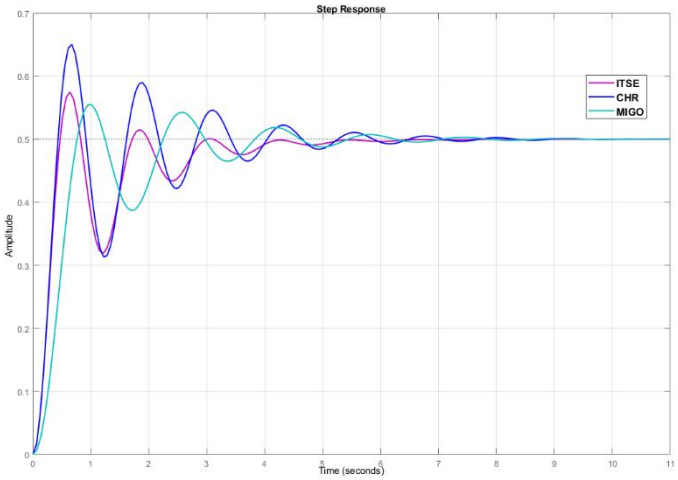


Figure 12: Step Response Excluding ZN and SIMC – Simulation

Gains obtained from simulation is inserted to the PID coding in Arduino. The velocity readings over time is recorded and plotted as in figure 13 and figure 14. The performance indicators results obtained from the step responses is tabulated in table 5.

Figure 13 shows the performance of all tuning methods when implemented while figure 14 excluded ZN as it produced oscillation output. It could be seen clearly from the response in figure 14 that AMIGO have the slowest rise time despite of the smooth response. SIMC produce some oscillation compare with another tuning methods.

As stated above, from simulation, ZN yields too much oscillation. As in figure 13, ZN produce much more oscillation and could not returned to steady state condition. Therefore ZN is excluded to be used. Based on the table 5, ZN does not have settling time as it is unable to return to steady state condition.

Based on table 5, GA is the best in terms of settling time and overshoot, 31.96% and 13.63% better than second best tuning method in those performance indicators, AMIGO. In terms of rise time GA is the third best compared with AMIGO, that have 140.135% worse rise time than GA. It could be concluded that GA outperform all of tuning methods in terms of overshoot percentage and settling time when implemented. In terms of rise time, although ZN and SIMC were better than GA, ZN produce huge number of overshoot percentage and could not return back to steady state condition while SIMC have second worst performance in terms of overshoot and settling time.

Based on the implementation on real application of cart follower, GA is decided to be used in cart follower application as it is the best in terms settling time and overshoot percentage with good performance in rise time compare with AMIGO and CHR. ZN and SIMC is excluded to be used as both of tuning methods produced oscillatory response and huge overshoot percentage.

Table 5: Performance for the PI Tuning Methods Real Application

Tuning Methods	Rise Time(s)	Overshoot (%)	Settling Time(s)
ZN	0.2528	180.75	-
SIMC	0.3023	31.73	14.076
CHR	0.4181	8.662	12.452
AMIGO	2.1812	7.8952	7.429
GA-ITSE	0.3839	6.8880	5.382

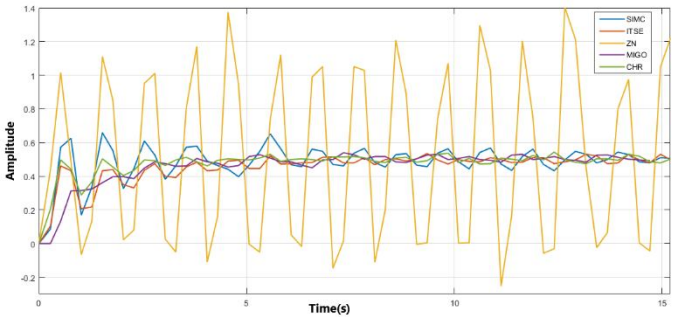


Figure 13: Step Response of Various Tuning Methods – Implementation

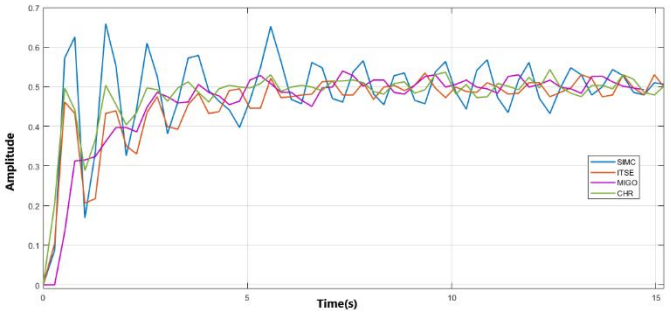


Figure 14: Step Response Excluding ZN - Implementation

4. CONCLUSION

The performance of GA tuning methods is evaluated and compared with ZN, CHR, SIMC, and AMIGO by using MATLAB simulation and real application. In simulation, GA is the best in terms of steady state error and settling time. AMIGO is the best in terms of overshoot while ZN is the best in terms of rise time. In real application, GA is the best among

all of tuning methods in terms of settling time and overshoot percentage. Although ZN and CHR is better than GA in terms of rise time, oscillating response and huge overshoot percentage is the main reason why both of the tuning methods is not suitable to be used in the application. Therefore GA is the fittest among all of the tuning methods as it yields the best performance in terms of settling time and overshoot percentage, with good performance in terms of rise time compared with AMIGO and CHR.

ACKNOWLEDGEMENT

This work was supported by Universiti Sains Malaysia, under the Bridging grant: 304/PELECT/6316200.

REFERENCES

1. A. Abdulameer, M. Sulaiman, M. Aras and D. Saleem. **GUI Based Control System Analysis using PID Controller for Education**, *Indonesian Journal of Electrical Engineering and Computer Science*, 3(1), pp. 91-101, 2016. https://doi.org/10.11591/ijeecs.v3.i1.pp91-101
2. A. Abdulameer, M. Sulaiman, M. Aras and D. Saleem. **Tuning Methods of PID Controller for DC Motor Speed Control**, *Indonesian Journal of Electrical Engineering and Computer Science* 3(2), pp. 343-349, 2016. https://doi.org/10.11591/ijeecs.v3.i2.pp343-349
3. Y. Nishikawa, N. Sannomiya, T. Ohta and H. Tanaka. **A method for auto-tuning of PID control parameters**, *Automatica*, 20(3), pp. 321-332, 1984. https://doi.org/10.1016/0005-1098(84)90047-5
4. A. Kiam Heong, G. Chong and L. Yun. **PID control system analysis, design, and technology**, *IEEE Transactions on Control Systems Technology*, 13(4), pp. 559-576. doi:10.1109/TCST.2005.847331, 2005.
5. S. Khandelwal, S. Aldhandi, and K.P. Detroja. **Would SISO IMC and SIMC tuning work for MIMO processes?**, *Indian Control Conference (ICC)*, IEEE, pp. 194-199, 2017. https://doi.org/10.1109/INDIANCC.2017.7846474
6. D.D. Kumar, C. Dinesh and S. Gautham. **Design and implementation of Skogestad PID controller for interacting spherical tank system**, *International Journal of Advanced Electrical and Electronics Engineering*, 2(4), pp.117-120, 2013.
7. K.H. Raunt and S.R. Vaishnay. **A Study on Performance of Different PID Tuning Techniques**, *International Conference on Electrical Engineering and Computer Science*, pp. 250-254, 2012.
8. H.E.A Ibrahim, F. N. Hassan and O.S. Anas. **Optimal PID control of a brushless DC motor using PSO and BF techniques**, *Ain Shams Engineering Journal 5(2)*, pp. 391-398, 2014. https://doi.org/10.1016/j.asej.2013.09.013
9. A. A. Aly. **PID parameters optimization using genetic algorithm technique for electrohydraulic servo control system**, *Intelligent Control and Automation 2.02*, p. 69, 2011. https://doi.org/10.4236/ica.2011.22008
10. A. Mirzal, S. Yoshii and M. Furukawa. **PID parameters optimization by using genetic algorithm**, *arXiv preprint arXiv*,1204.0885, 2012.
11. T. Samakwong, and W. Assawinchaichote. **PID controller design for electro-hydraulic servo valve system with genetic algorithm**, *Procedia Computer Science 86*, pp. 91-94, 2016 https://doi.org/10.1016/j.procs.2016.05.023
12. A. Sungthong and W. Assawinchaichote**. Particle swam optimization based optimal PID parameters for air heater temperature control system**, *Procedia Computer Science 86*, pp. 108-111, 2016. https://doi.org/10.1016/j.procs.2016.05.027
13. G.S. John and A.T. Vijayan**. Anti-windup PI controller for speed control of brushless DC motor**, *IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, 2017, pp. 1068-1073. https://doi.org/10.1109/ICPCSI.2017.8391874
14. A.A.M. Zahir, S.S.N. Alhady, W.A.F.W. Othman, A.A.A. Wahab, and M.F. Ahmad. **Design and Modeling of Autonomous Cart Follower for Wheelchair User**, *Platform : A Journal of Engineering*, In-Review.
15. A. A. M. Zahir, S.S.N. Alhady, W.A.F.W. Othman and M.F. Ahmad. **Genetic Algorithm Optimization of PID Controller for Brushed DC Motor**, *Intelligent Manufacturing & Mechatronics: Proceedings of Symposium*, Springer Singapore, 2018, pp. 427-437. https://doi.org/10.1007/978-981-10-8788-2_38