mekatronika – Journal of Intelligent Manufacturing & Mechatronics
VOL. 02, ISSUE 01, 8 – 15
DOI: https://doi.org/10.15282/mekatronika.v2i1.3016

ORIGINAL ARTICLE

Implementation of a Discrete Prescribed Performance Controller (DPPC) based on a Discrete PID Scheme on a DC Motor System

A. Y. Yaakob[1] and M. H. M Ramli[1,2]

[1]Faculty of Mechanical Engineering, Universiti Teknologi MARA, 40450 Selangor, Malaysia.
[2]Sports Engineering & Artificial Intelligence Centre, UiTM Shah Alam 40450, Malaysia.

ABSTRACT – The Proportional, Integral, Derivative controller, or PID is the most widely used controller in industry today; A relatively simple controller with little parameters to tune but capable achieving great performance, provided with proper tuning. A fairly recent development is the Prescribed Performance Controller (PPC) by Bechlioulis and Rovithakis; an innovative controller that limits error propagation to within a pre-defined boundary region which is described by a performance function. This paper proposes a new controller that integrates the PPC solution into the current PID scheme in an attempt to achieve improved control performance compared to the original PID scheme, aptly named the DPPCPID controller. Further improvements is also analysed through the implementation of an anti-windup function and a fuzzy logic gain scheduling system. A DC motor system is used to verify the capabilities of the proposed controller. Preliminary testing concluded that the proposed DPPCPID controller, as well as its subsequent iterations implementing the additional improvement functions in various combinations, does indeed have the capabilities to provide performance benefits over the standard PID scheme, in addition to other benefits.

ARTICLE HISTORY
Received: 30 September 2019
Accepted: 8 December 2019

KEYWORDS
Adaptive control
PID
Prescribed performance
Anti-windup
Gain scheduling

Introduction

It has come to no surprise that the Proportional, Integral, Derivative or PID controller as it is more commonly referred, is the most popular in the market today. This is rightfully so thanks to its simple design, ease of tuning and great performance potential and applications. This popularity has led to a significant number of contributions in new studies and advancements in control theory, as demonstrated by [1-4], among many, many others. Reiterating [1], PID control has contributed to solving over 95% on industrial control problems.

The term Proportional, Integral, Derivative is a direct reference to the functions that constitute to the formation of the PID controller. Respectively these functions is a multiplication of the error proportional to the reference signal and the actual output, its integral and its derivative, with arbitrarily tuned constants [3]. The sum of the products of these functions constitute to the output of the PID controller- the control signal.

This simple design is what makes the PID so versatile. For any application it only needs three tuned constants to produce a proper control output. Often these constants are selected through trial and error, but many methods has been established to formally select these constants, such as the popular Ziegler-Nichols method as applied by [3].

A recent advancement in control algorithms is the Prescribed Performance Controller or PPC by Bechlioulis and Rovithakis [5]. The novelty of this controller is that it has the capability to guarantee convergence of the tracking error by limiting the tracking error evolution within a pre-defined region. This region is established arbitrarily, depending on the desired performance of the system, using a positive decreasing function, also known as the *performance function* [6]. The decreasing nature of this function results in a continually shrinking range of possible values for the tracking error, thus guaranteeing convergence over time. The rate of convergence, while depending greatly on the selected *performance function*, can be further tuned by constants introduced throughout the controller.

This study proposes that a new hybrid controller could be formulated by integration of the PPC into a working PID scheme. This new PPCPID controller hybrid would essentially feature the best of both its component controllers - namely being simple in design, easy to tune and capable of guaranteeing error convergence.

To evaluate the performance of the new controller, a simple DC motor system is to be used. This system would represent a common application of a one DOF

*CORRESPONDING AUTHOR | M. H. M. Ramli | ✉ haniframli@uitm.edu.my

system. In this application the controller input is the encoder reading of the angular position of the motor shaft, and the control output the magnitude and direction of the shaft torque, controlled by means of a commercial DC motor driver. Using this setup, three test cases were established. Case 1: Shaft angular position control using a step signal reference. Case 2: Shaft angular position control using a sine wave signal reference. Case 3: Pendulum motion damping. Two metrics were selected to quantify the controller performance: 1) The duration of the transient period and 2) The root mean square error of the controlled system.

The controller formulation and subsequent application were done in discrete time in account of the low number of established data of PPC integrated controllers done in discrete time specifically, and at the same time benefiting from the nature of discrete time systems where the computational load is easily variable by means of controlling the sampling rate.

Reduction of the sampling rate dramatically reduces the computational load of the controller, given the much lower amount of data being processed, but comes at the cost of reduced controller effectiveness. This also works in the other direction, with a fairly linear relation between computational requirements and performance. If set correctly, this can result in a more consistent sampling rate between all tested controllers, eliminating any factors of inadequate hardware processing capabilities that could affect the produced data. Given that the focus of this study is the performance comparison between different controllers, the effectiveness reduction can be neglected, so long as the sampling rate is set constant between all of them.

Anti-Windup and Gain Scheduling modules are formulated as an added improvement to the original controller. Ultimately this would serve as an example application of the controller system where simple augmentations are added onto the controller system to further improve controller effectiveness on the specific application.

To evaluate the performance of the formulated controllers, a simple DC motor system is set up as a simple SISO example. A commercially available Arduino microcontroller is used as the processor for this system. It is responsible for both calculating the controller output each cycle, and producing the respective data through the serial port. For every controller test, the Arduino is reprogrammed with the new algorithm and the test repeated. The DC motor comes with a built-in encoder that reads shaft angular position, serving as the input device of the system, and the output the angular position of the shaft. The controllers are evaluated based on several selected performance matrices including transient duration and root mean square error.

Preliminary

This section summarizes the preliminary knowledge on the concepts of the PID controller, discrete prescribed performance controllers (DPPC), PID-windup and gain scheduling system.

a. PID Control

The control action of a PID controller is the sum of its three components: the proportional component acting on the current error, the integral component on past errors, and the derivative component on future errors. Each with a gain constant $\boldsymbol{k_p}$, $\boldsymbol{k_i}$, and $\boldsymbol{k_d}$, respectively. This forms the equation

$$u(t) = k_p e(t) + k_i \int_0^t e(t)dt + k_d \frac{de(t)}{dt} \quad (1)$$

where $\boldsymbol{u(t)}$ is the control output. In discrete time, the equation is defined as

$$u_k = k_p e_k + k_i v_k + k_d (e_k - e_{k-1}) \quad (2)$$

where $\boldsymbol{v_k}$ is the integral of past errors, defined as

$$v_k = v_{k-1} + k_i e_k \quad (3)$$

b. Prescribed Performance Control

Here the PPC is defined directly in discrete time, reiterating [6]. Consider a general tracking error $\boldsymbol{e_k} \in \mathfrak{R}$. Prescribed performance is achieved if the following inequality is satisfied

$$-\underline{h}\lambda_k < e_k < \overline{h}\lambda_k \quad (4)$$

where $\boldsymbol{\lambda_k}$ is a positive decreasing function known as the *performance function* and both $\underline{\boldsymbol{h}}$ and $\overline{\boldsymbol{h}}$ are magnification constants. Function $\boldsymbol{\lambda_k}$ and constants $\underline{\boldsymbol{h}}$ and $\overline{\boldsymbol{h}}$ are selected to define the desired performance metrics of $\boldsymbol{e_k}$ in the transient and steady-state regions. For this study, the performance function $\boldsymbol{\lambda_k}$ are selected following [6], defined as

$$\lambda_{k+1} = (1-\omega)\lambda_k + \omega\lambda_\infty \quad (5)$$

for which $\boldsymbol{\omega} \in (\mathbf{0}, \mathbf{1})$, a constant defining the convergence rate of the function $\lim_{k\to\infty} \boldsymbol{\lambda_k} = \boldsymbol{\lambda_\infty}$, and $\boldsymbol{\lambda_0} > \boldsymbol{\lambda_\infty} > \mathbf{0}$. $\boldsymbol{\lambda_\infty}$ is the ultimate allowable steady state error boundary, while $\boldsymbol{\lambda_0}$ is the maximum bound of the initial tracking error.

To achieve the prescribed performance metric specified in (4), the original output error is first transformed into a new coordinate ε_k. Specifically,

mekatronika – Journal of Intelligent Manufacturing & Mechatronics
VOL. 02, ISSUE 01, 8 – 15
DOI: https://doi.org/10.15282/mekatronika.v2i1.3016

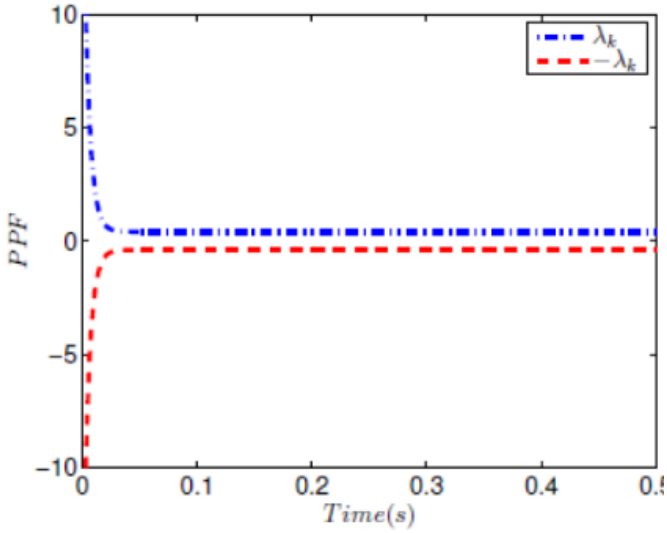


Figure 1. An illustration of the performance function specified in (5). Adapted from [6].

$$\varepsilon_k = \lambda_k \Gamma(e_k) \tag{6}$$

where $\boldsymbol{\Gamma(e_k)}$ is a strictly increasing function that satisfies

$$\Gamma(\varepsilon_k) \in (-1,1) \text{ for all real numbers of } \varepsilon_k \tag{7}$$

and

$$\lim_{\varepsilon_k \to +\infty} \Gamma(\varepsilon_k) = 1 \; and \; \lim_{\varepsilon_k \to -\infty} \Gamma(\varepsilon_k) = -1 \tag{8}$$

The inverse transformation of $\boldsymbol{\varepsilon_k}$ can easily be obtained due to the properties of $\boldsymbol{\Gamma(e_k)}$ and $\boldsymbol{\lambda_0 > \lambda_\infty > 0}$, forming the equation

$$\varepsilon_k = \Gamma^{-1}\left(\frac{e_k}{\lambda_k}\right) \tag{9}$$

Provided $\boldsymbol{\lambda_0}$ is properly selected, such that $\boldsymbol{\lambda_0 > e_0 > -\lambda_0}$, $\boldsymbol{k}$ is bounded and (2.2.1) is maintained, then $\boldsymbol{\Gamma(\varepsilon_k) \in (-1, 1)}$ holds. In this regard, the strictly increasing function $\boldsymbol{\Gamma(\varepsilon_k)}$ is given as

$$\Gamma(\varepsilon_k) = \left(\frac{\overline{h} e_k{}^{-\varepsilon_k} - \underline{h} e_k{}^{\varepsilon_k}}{e_k{}^{\varepsilon_k} + e_k{}^{-\varepsilon_k}}\right) \tag{10}$$

The transformed error $\boldsymbol{\varepsilon_k}$ can thus be derived as

$$\varepsilon_k = 0.5 \ln\left(\frac{\lambda_k \underline{h} + e_k}{\lambda_k \overline{h} - e_k}\right) \tag{11}$$

In view of equation (11), the problem of prescribed performance achievement is transformed into the less demanding boundedness preservation issue.

c. PID Windup

Whenever there is a large instantaneous change in the reference signal, the integral component of the PID controller would accumulate a large error value. This error subsequently results in a large overshoot when compensated by the PID controller - the initial 'wind up'. This overshoot is then itself another large error which causes another overshoot, repeating the cycle multiple times as the controller attempts to bring the system back to its steady state [7].

d. 2.4. Gain Scheduling System

Traditional applications of PID controllers often rely only on a single set of gain constants tuned for all cases of the control system. While simple, they are not always the most effective solution, given that for most systems, the best tuning gains would vary greatly depending on the current condition of the system. The fuzzy logic gain scheduling system introduces a method that allows dynamic tuning of the PID controller. Using the fuzzy logic module, the controller can automatically switch to specific sets of tuning gains depending on set conditions, thus maintaining the best tuning gains for every condition of the system. [8]

Controller Formulation

We begin by first designing the base hybrid controller that combines the PPC and PID controllers, then improving the newly formulated controller by the implementation of an anti-windup function and a gain

*CORRESPONDING AUTHOR | M. H. M. Ramli | ✉ haniframli@uitm.edu.my

scheduling system. In consideration of avoiding numerical approximation problems that could degrade system performance, the controllers in this study are designed directly in discrete-time, instead of the more common continuous-time domain.

a. DPPCPID Control

The Discrete Prescribed Performance Control Proportional Integral Derivative (DPPCPID) hybrid controller is simply the application of the DPPC equation (11) into the DPID equation (2), forming the control equation

$$u_k = k_p \varepsilon_k + k_i v_k + k_d(\varepsilon_k - \varepsilon_{k-1}) \tag{12}$$

ε_k is slightly modified to include a magnification constant β, forming the equation

$$\varepsilon_k = \beta\, 0.5 \ln\left(\frac{\lambda_k \underline{h} + e_k}{\lambda_k \bar{h} - e_k}\right) \tag{13}$$

b. DPPCPIDAW Control

The first improvement to the basic DPPCPID controller established in (12) is to address the windup problem discussed in the previous section, visualized in Figure 2. Implementing an anti-windup function into (12) forms the equation

$$u_k = Us + k_p \varepsilon_k + \left(k_i t - k_p\right)(\varepsilon_{k-1} + Es) \tag{14}$$

where $\boldsymbol{Us}$ is the previous control action bounded within the initially set limits

$$Us = \begin{cases} u_{max} & if\ u_{k-1} \geq u_{max} \\ u_{k-1} & if\ u_{min} < u_{k-1} < u_{max} \\ u_{min} & if\ u_{k-1} \leq u_{min} \end{cases} \tag{15}$$

and $\boldsymbol{Es}$ a magnification factor defined as

$$Es = \left(\frac{1}{k_p}\right)(Us - u_{k-1}) \tag{16}$$

Following (14), the controller is now abbreviated as DPPCPIDAW.

c. DPPCPIDAWGS Control

The second improvement is the implementation of the gain scheduling system, forming the DPPCPIDAWGS controller.

The gain scheduling system is set up by tuning several sets of the tuning gains for the different conditions prior system operation. For this study, these conditions are defined using the current absolute error of the system. For k_p,

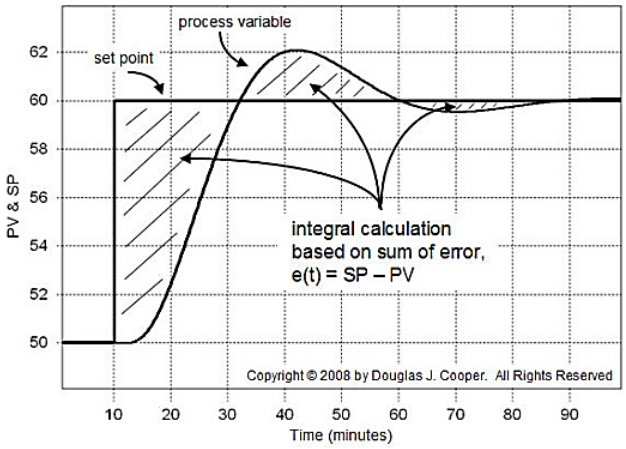


Figure 2. An illustration of the effects of the windup problem in PID controllers. Adapted from [7].

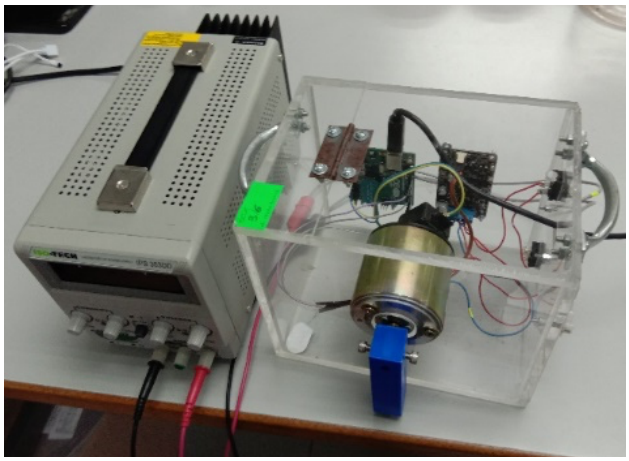


Figure 3. An overview of the fully assembled system built inside a clear acrylic case.

$$k_p = \begin{cases} \boldsymbol{k_{p1}} & \boldsymbol{if\ 0 \leq |e| \leq 25} \\ \boldsymbol{k_{p2}} & \boldsymbol{if\ 25 < |e| \leq 50} \\ \boldsymbol{k_{p3}} & \boldsymbol{if\ 50 < |e| \leq 100} \\ \boldsymbol{k_{p4}} & \boldsymbol{if\ |e| > 100} \end{cases} \tag{17}$$

and for k_i

$$k_i = \begin{cases} \boldsymbol{k_{i1}} & \boldsymbol{if\ 0 \leq |e| \leq 25} \\ \boldsymbol{k_{i2}} & \boldsymbol{if\ 25 < |e| \leq 500} \\ \boldsymbol{k_{i3}} & \boldsymbol{if\ 50 < |e| \leq 100} \\ \boldsymbol{k_{i4}} & \boldsymbol{if\ |e| > 100} \end{cases} \tag{18}$$

Experiment Design

This section discusses the experimental setup used to test the hybrid controllers.

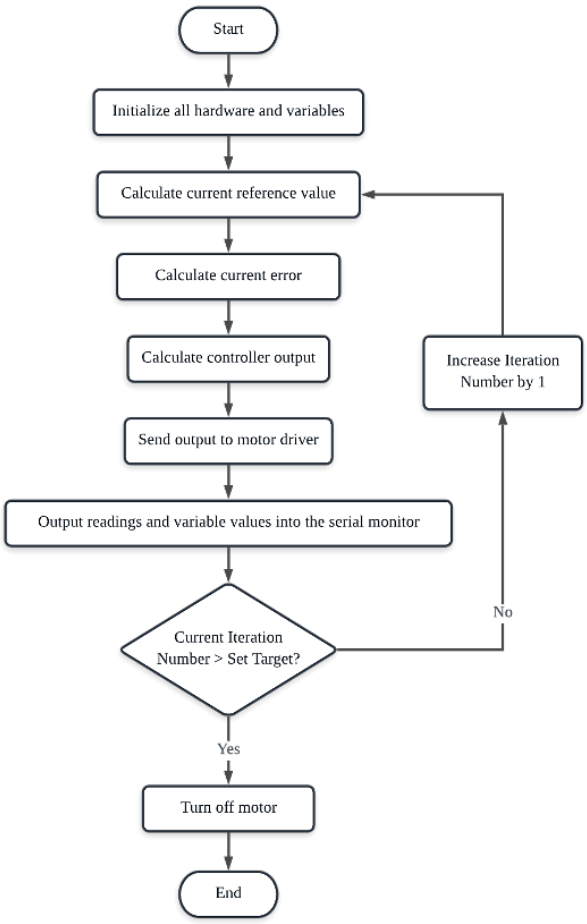


Figure 4. A flow chart illustrating the microcontroller program flow.

Table 1. List of components used in the DC motor system.

Component	Model
Microcontroller	Arduino Uno R3
Motor Driver	Cytron MD10C 10Amp 5-30V
Motor	RS Pro 263-6011 30W Servo
Encoder	Broadcom HEDS-5500#A11
Power Supply	ISO-Tech IPS 303DD

a. Hardware Setup

The DC motor system considered for the controller implementation comprises of several commercial components, listed in Table 1. Figure 3 illustrates the fully assembled system.

The Arduino microcontroller is connected directly to the computer (not illustrated in Figure 3) for programming and data collection. The programming structure is illustrated in Figure 4. The RS Pro 263-6011 servo motor is selected specifically for its built-in encoder, thus eliminating any possible compatibility issues. The servo designation is simply a manufacturer designation. For all intents and purposes the RS Pro is physically a DC motor. The built in Broadcom HEDS-5500#A11 encoder maps the shaft rotation to a 511-step position, each step representing 0.705° of shaft rotation. The Cytron MD10C driver is selected to supply enough power to the motor, ensuring optimal performance. It also features a simplified control interface, mapping the 255-bit PWM output of the Arduino microcontroller to 0% - 100% of the motor input voltage supplied by the MD10C driver, regulating applied torque on the motor shaft.

The control algorithm is implemented using the Arduino IDE on the connected computer. As illustrated in Figure 4, for each processing cycle of the microcontroller, a complete set of data including the controller output, shaft position, and reference, among other variables are sent to the built in Serial Monitor of the Arduino IDE. The time period of each cycle depends on the processing time, each cycle varying slightly with an average period of roughly 4.5ms per cycle. After each experiment run, the data set logged in the Serial Monitor is exported to Microsoft Office Excel 2016 for further analysis. The total run-time for each experiment run is controlled by a specified limit to the number of processing cycles, turning off the motor and thus ending the experiment run once the limit is reached.

b. Test Applications

Three test applications are proposed to evaluate the performance of the controllers. The first test using a step signal reference with a free shaft. Second, a constant sine wave signal reference and a free shaft. Third, with an attached pendulum mass and application of an initial displacement. Each test is thought to be analogous to a common control application of the DPID controller.

For each test application the DPPC and DPID controllers are first tuned manually by trial-and-error to achieve proper convergence and steady state stability. The tuning parameters are then used for the performance evaluation of the hybrid controllers. This is essential to ensure that the change in performance are a result of the architectural change of the controllers themselves, rather than the tuning quality. For the gain scheduling system, tuning is done by using the established tuning values in one of the condition rulesets. The rest of the conditions are then tuned by increasing or decreasing the value of k_p.

The metrics selected for evaluating performance are the transient period or settling time and root mean square error of the steady state region. In both metrics, lower values correspond to better performance. For the sine wave test, only the root mean square error is considered as the test is focused more on steady state stability of the system.

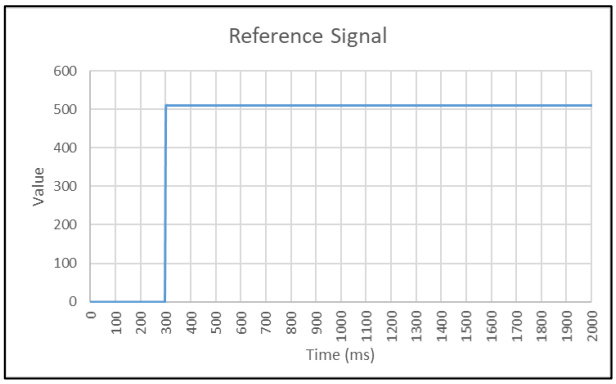


Figure 5. A graph plotting the step signal reference used in Application 1.

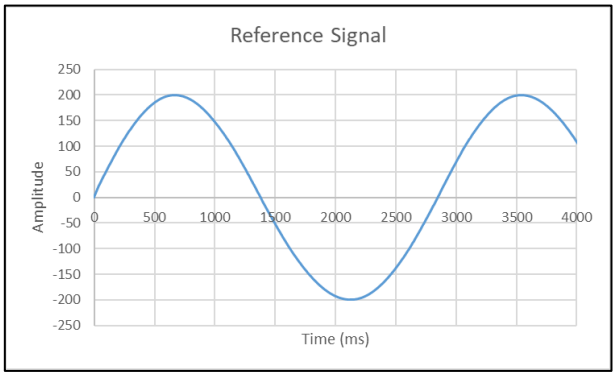


Figure 6. A graph plotting the sine wave signal reference used in Application 2.

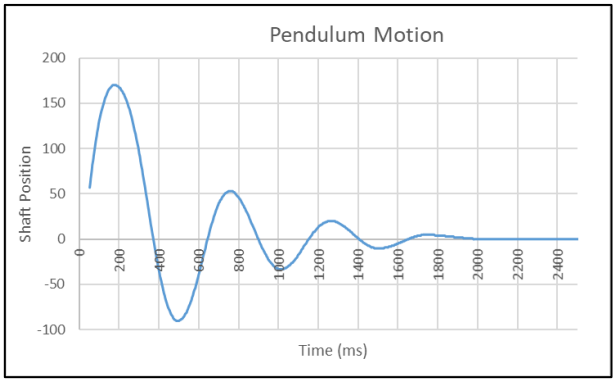


Figure 7. A graph plotting the initial displacement and subsequent pendulum motion without the any controller influence in Application 3. The quick settling a result of the magnetic resistance from the connected motor circuit.

Application 1: Step Signal Reference

Figure 5 shows a simple step signal starting with a continuous 0 value signal followed with a spike at 300ms with a value of 511. The signal value corresponds to the motor shaft position as read by the encoder. The controller is implemented to adjust the shaft position according to the reference.

Application 2: Sine Wave Signal Reference

Figure 6 shows a constant sine wave with a frequency of roughly 0.5 Hz and amplitude of 200, where the amplitude corresponds to the encoder reading of the motor shaft position. Like the previous setup, the controllers are required to match the shaft position to the reference signal. The test is focused on the steady state stability of the controllers following a dynamic reference, thus for this application the settling time is ignored.

Application 3: Pendulum Motion

A solid pendulum of mass 125g with length of 55mm from the center of rotation is attached to the motor shaft. At the start of each test an initial displacement is applied by sending maximum voltage to the motor for 55ms, causing a consistent small jolt to start the pendulum motion. The reference signal is set to a constant 0, referring to the resting position of the motor shaft prior to the initial displacement. The control challenge is to reduce the swinging motion of the pendulum and subsequently reach steady state as fast as possible. Figure 7 shows the initial displacement and subsequent pendulum motion without the any controller influence.

Results and Discussion

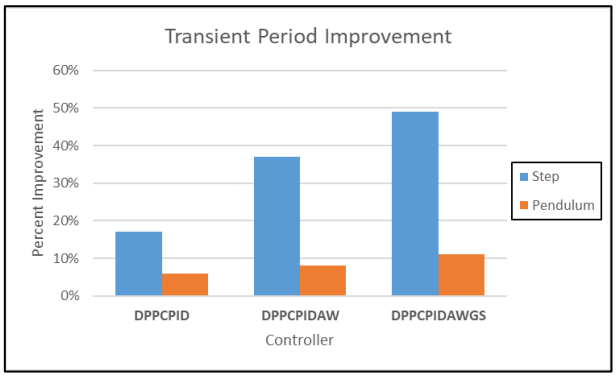
For every application, the controllers are evaluated in comparison to the standard DPID controller. The same tuning parameters are used for all controllers except for the DPPCPIDAWGS where a special tuning method is applied, as discussed in the previous section of this paper. The tuning parameters is listed first, followed by the performance comparison.

a. Application 1: Step Signal Reference

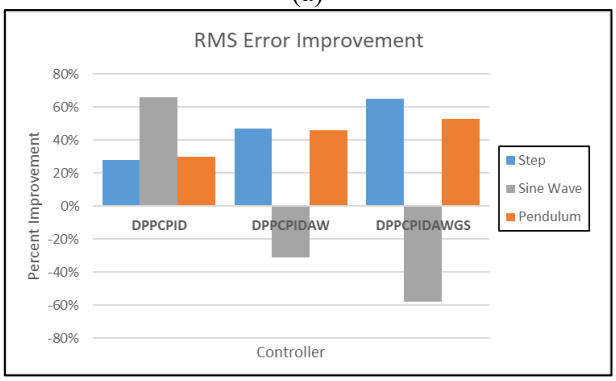
Performance improvement were achieved. The most notable being the DPPCPIDAWGS controller with 49% and 65% improvement compared to the DPID baseline for the transient duration and root mean square error respectively.

b. Application 2: Sine Wave Signal Reference

Performance improvement were only achieved for the base DPPCPID controller at a significant 66% compared to baseline. The DPPCPIDAW and DPPCPIDAWGS degraded system performance by 31% and 58% respectively. This could be attributed to the anti-windup function reducing the capability of the controller to adapt to constantly changing reference signals.



(a)



(b)

Figure 8. (a) The performance change of the hybrid controllers compared to the DPID controller baseline of transient duration and (b) root mean square error.

Table 2. The tuning parameters used for test Application 1.

Parameter	Description	Value
K_p	Proportional gain	0.7
K_i	Integral gain	0.02
K_d	Derivative gain	0
ω	Performance function gain	0.044
λ_∞	Allowable steady state error	1
$\bar{h}$	Performance bounds	1
$\underline{h}$		1
β	DPPC magnification constant	250

Table 3. The tuning parameters for the DPPCPIDAWGS controller in test Application 1.

Absolute Error	0 - 24	25 - 49	50 - 99	>100
K_p	1.5	1.3	0.9	0.7
K_i	0.02	0.02	0.02	0.02
K_d	0	0	0	0

Table 4. The performance comparison of the controllers in test Application 1.

Controller	Transient Duration (ms)	RMS Error
DPID	714.7	0.1977
DPPCPID	596.7(-17%)	0.1418 (-28%)
DPPCPIDAW	450.0 (-37%)	0.1046 (-47%)
DPPCPIDAWGS	364.0 (-49%)	0.0685 (-65%)

Table 5. The tuning parameters used for test Application 2.

Parameter	Description	Value
K_p	Proportional gain	0.9
K_i	Integral gain	0.05
K_d	Derivative gain	0
ω	Performance function gain	0.0001
λ_∞	Allowable steady state error	1
$\bar{h}$	Performance bounds	1
$\underline{h}$		1
β	DPPC magnification constant	650

Table 6. The tuning parameters for the DPPCPIDAWGS controller in test Application 2.

Absolute Error	0 - 24	25 - 49	50 - 99	>100
K_p	1.7	1.5	1.1	0.9
K_i	0.05	0.05	0.05	0.05
K_d	0	0	0	0

Table 7. The performance comparison of the controllers in test Application 2.

Controller	RMS Error
DPID	2.5441
DPPCPID	0.8604 (-66%)
DPPCPIDAW	3.3311 (+31%)
DPPCPIDAWGS	4.0319 (+58%)

Table 8. The tuning parameters used for test Application 3.

Parameter	Description	Value
K_p	Proportional gain	1
K_i	Integral gain	0.001
K_d	Derivative gain	0.45
ω	Performance function gain	0.001
λ_∞	Allowable steady state error	1
$\bar{h}$	Performance bounds	1
$\underline{h}$		1
β	DPPC magnification constant	250

Table 9. The tuning parameters for the DPPCPIDAWGS controller in test Application 3.

Absolute Error	0 - 24	25 - 49	50 - 99	>100
K_p	1.2	1.1	1.1	1
K_i	0.001	0.001	0.001	0.001
K_d	0	0	0	0

Table 10. The performance comparison of the controllers in test Application 3.

Controller	Transient Duration (ms)	RMS Error
DPID	1851.7	0.3288
DPPCPID	1748.3(-6%)	0.2298 (-30%)
DPPCPIDAW	1709.0 (-8%)	0.1780 (-46%)
DPPCPIDAWGS	1652.3 (-11%)	0.1554 (-53%)

c. Application 3: Pendulum Motion

Quite similarly to test Application 1, for test Application 3 performance achievement were achieved, albeit at a smaller scale. The most notable improvement is again the DPPCPIDAWGS controller with 11% and 53% improvement over baseline for transient duration and root mean square error respectively.

In static reference cases, as shown in test applications 1 and 3, the hybrid controllers were able to provide a considerable boost to the system performance over the standard PID controller, with the best being the DPPCPIDAWGS controller. In dynamic reference cases however, only the basic DPPCPID controller were able to provide a performance improvement.

Conclusion

This study proposes an innovative control strategy of combining the common PID controller with a recently developed Prescribed Performance Controller. This approach is chosen to improve the performance of said PID controller without the need for additional hardware or system modelling. Further improvements were also proposed by integration of an anti-windup function and implementation of a gain scheduling system. A simple DC motor system comprising of commercially available components were used as an evaluation platform, where three applications were tested. The results of this study have shown that the proposed hybrid controllers are indeed capable of providing an increase in performance over the baseline PID controller, depending on the application. In static reference applications, the DPPCPIDAWGS controller provides the highest increase in performance, while in dynamic reference applications, the basic DPPCPID is best.

References

[1] K. J. Astrom and R. M. Murray, Feedback Systems: An Introduction for Scientist and Engineers, Princeton, New Jersey 08540: Princeton University Press, 2012.

[2] A. Zulu and S. John, "A Review of control Algorithms for Autonomous Quadrotors," Open Journal of Applied Sciences, no. 4, pp. 547-556, 2014.

[3] J. C. Basilio and S. R. Matos, "Design of Pi and PID Controllers with Transient Performance Specification," IEEE Transactions on Education, vol. 45, no. 4, pp. 364-370, 2002.

[4] Z. Doulgeri and Y. Karayiannidis, "PID Type Robot Join Position Regulation with Prescribed Performance Guaranties," IEEE International Conference of Robotics and Automation, pp. 4137-4142, 2010.

[5] C. P. Bechlioulis and G. A. Rovithakis, "Robust Adaptive Control of Feedback Linearizable MIMO Nonlinear Systems with Prescribed Performance," IEEE Transactions on Automatic Control, vol. 53, no. 9, pp. 2090-2099, 2008.

[6] M. H. M. Ramli, N. A. C. Zakaria, Z. Mohamed and M. A. M. Dzahir, "Nonlinear Hysteresis Compensation via an Adaptive Extended Discrete-Prescribed Performance Control," International Journal of Engineering & Technology, vol. 5, 2017.

[7] "Integral (Reset) Windup, Jacketing Logic and the Velocity PI Form," Control Guru, 07 April 2015. [Online]. Available: https://controlguru.com/integral-reset-windup-jacketing-logic-and-the-velocity-pi-form/. [Accessed 22 May 2019].

[8] "Transient State," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Transient_state. [Accessed 12 June 2019].

[9] H. Ahmed and M. Ayyub, "A Scheme of Gain Scheduling for Tuning PID Controllers Using Fuzzy Logic," Department of Electrical Engineering, Aligarh Muslim University, Aligarh.