ZULFAQAR Journal of Defence Science, Engineering & Technology
e-ISSN: 2773-5281
Vol. 8, Issue 1 (2025)
DOI: https://doi.org/10.58247/jdset-2025-0801-03
Journal homepage: https://zulfaqarjdset.upnm.edu.my

MODELLING AND IMPLEMENTATION OF CONTROL LAW DESIGN FOR UAV CONTROL SURFACE USING PIDS CONTROLLER AND FEEDBACK CONTROLLER WITH INTEGRATOR

Zuhairi Abdul Rashid[a*], Syed Mohd Fairuz Syed Mohd Dardin[a], Khairol Amali Ahmad[a], Akram Abdul Azid[a]

[a] Department of Electrical & Electronic Engineering, Faculty of Engineering, National Defence University of Malaysia, Sg. Besi Camp, 57000 Kuala Lumpur, Malaysia

ARTICLE INFO

ARTICLE HISTORY
Received : 07-10-2024
Revised : 10-12-2024
Accepted : 12-04-2025
Published : 31-05-2025

KEYWORDS
Fixed-wing UAV
PID Controller
Pole Placement
Control Surfaces
Robust System

ABSTRACT

This paper discusses the design of a control law for UAV control surface deflection. The approach in obtaining the control law start by designing the test bench follows by identifying the characteristic of test bench, follow by designing the control law itself. Two control design methods, the PID controller and the pole placement method, have been used, resulting in four derived control laws. Only one control law was selected—the PI controller with a 0.3s settling time. The control law selected shown its capability to reject perturbation while maintaining the desired response characteristic. Thus, the control law for UAV control surfaces has been successfully design and implemented.

1.0 INTRODUCTION

Control surfaces were used in fixed wing type aircraft to control the attitude of the aircraft around its centre of gravity. This is also true for fixed wing type UAVs except for the method of deflecting the control surfaces are different. For conventional type aircraft, the control surface was deflected using hydraulic actuator. But for more modern aircraft, the electro-mechanical actuator system is in high demand as to reduce weight and increase safety [1]. Regardless of whether it is in an aircraft or in a UAV, the control surface deflection gov-ern the aircraft attitude and need to be reliable when deflected. The reliability of a control surface is its ability to stay on a given angle of deflection and reject any perturbations acting on it. For conventional aircraft, the control surface is equipped with positioning sensors for measuring the control surface deflection typically using the linear variable differential transformer (LVDT), or the rotary variant (RVDT) [2]. But often, for smaller UAVs, such sensors are not equipped with assumption that the control surfaces will remain on its de-flection regardless of any perturbations acting on it. This assumption is good enough in modelling the UAVs characteristic as we can see in the modelling of the X-bound vehicle and the P15035 UAV [3].

For this project, a study will be done for a UAV with wing loading criteria of below 50 kg/m2 (low) and weight below 5 kg (very light) [4, 5]. This study is conducted in motivation on designing robust Unmanned Aerial Vehicles (UAV) with low wing loading criteria for operating in a heavy crosswind condition. For that, the control surface deflection needs to hold its position correctly at all costs. Thus, a robust controller needs to be designed for the deflection of the control surfaces with addition of position sensor for feedback. This will create the ability for the control surface to reject any perturbation acting on it. The design for the deflection of the control surfaces with addition of position sensor for feed-back. This will create the ability for the control surface to reject any perturbation acting on it. Preliminary results for this study have been done previously and presented in 2019 4th International Conference on Control, Robotics and Cybernetics (CRC) [6]. From the original work done, the perturbation rejection is successfully implemented for a UAV control surface using poles placement method with integrator. However, the actual load test on the control

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

surfaces has not been done thoroughly to evaluate the perturbation rejection capability of the designed controller. On top of that, there are jitter or noise at the sensors reading and servo motor inputs for the designed controller as shown in Figure 1. This will affect the movement of the control surface whereby observation, the control surface is vibrating according to the noise or jitter.

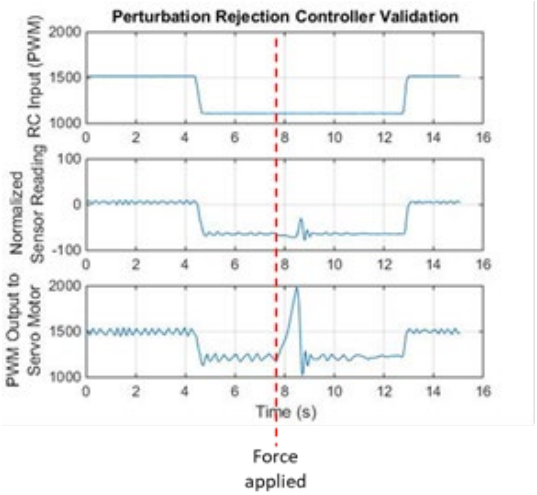


Figure 1. Results from previous study [6]

In this paper, a filtering method will be presented and implemented to reduce the effect of jittering. This will include the selection criteria for the filtering method to be used in the UAV control surface controller and the best way to implement it. This paper will also discuss on designing the perturbation rejection controller using PID controller as well as poles placement method. Controller design will be based on the output signal which has been filter using the proposed designed filter. The need to compare both methods of control laws is because there are no indefinite studies that show one method is superior to the others. For examples, Talat et al. has concluded that poles placement method with integrator is superior compared to PID controller in model-ling air heating system, while Kasturi et al. concluded that PID controller is better for their flow control [7-8]. The comparative summary between PID controller and poles placement method can are shown in Table 1.

Table 1. Comparison between PID controller and poles placement method

Parameter	Parameter Description	PID	Poles Placement
MIMO/SISO	What type of system is suitable using this method?	SISO	MIMO
Method	What is the method/calculation used to obtain the gain/desired results	Tuning	Calculation
Integrator	Integrator in a system will ensure no steady state error	Readily available	Need to be added to the calculation
MATLAB's Toolbox	Is there any toolbox available in MATLAB to ease the calculation?	PID Tuner	No available toolbox
Advantages	What benefits are there using this method?	Ease of tuning	Specific characteristics can be imposed

In Table 1, the PID controller design method is readily available in MATLAB while poles placement method allow user to dictate specified characteristic onto the control laws. Preliminary, there are not enough arguments to select one single method to be used onto the system. Therefore, a thorough investigation for both controllers need to be done specifically in controlling the UAV's control surfaces. The overall proposed design for filter and controller will be implemented using a test bench created for testing and validation purposes. This is to ensure the design is correctly done before implementing it onto the real control surfaces of an UAV. The objectives for this paper are as follows:

a. To identify the natural characteristics for the control surface test bench.
b. To design and implement a filter for the control surface deflection position.
c. To design and implement a perturbation rejection controller in controlling the control surface using PID tuning method.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

d. To design and implement a perturbation rejection controller in controlling the control surface using poles placement method.
e. To evaluate the best control laws scheme that can be implemented for controlling the UAV's control surfaces.

The outcome of this project is to have a workable perturbation rejection controller for the UAVs control surfaces that are implemented into a microcontroller which can be further explored under dynamic conditions.

2.0 METHODS AND MATERIAL

2.1 Test Bench Setup

The testing will be done using the test bench representing the actual control surface deflection as shown in Figure 2.

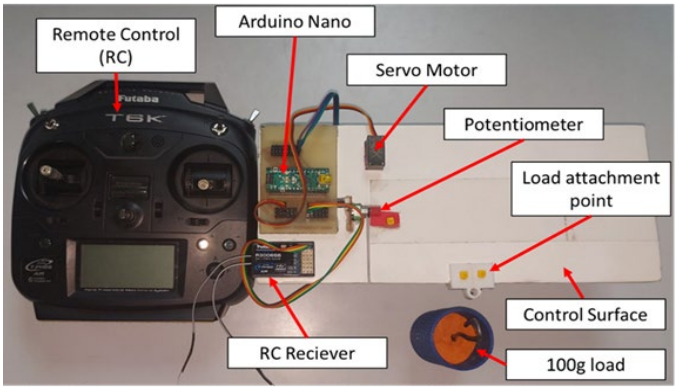


Figure 2. Control surface test bench

In Figure 2, the input for the test bench is the PWM input from the remote control (RC) receiver while the output is the potentiometer reading representing the control surface de-flection. These parameters as well as another related parameter will be shown in Arduino IDE terminal on a PC. The 100g load will be attached to the load attachment point for load testing when required. With the control surface area for the test bench of 0.01m2, the 100g load will simulate a 10kg/m2 wing loading on the control surface. However, this calculation is an estimation and simplification of the actual wing loading acting on the control surface. To have an actual wing loading acting on the control surface will be a complex process, and extra calculation needs to be conducted as shown by Manuel et al. [9]. As the focus of this project is to design a control law for the control surfaces, the estimation stated is used. The control surface will be deflected by the servo motor trough a push rod as shown in Figure 3.

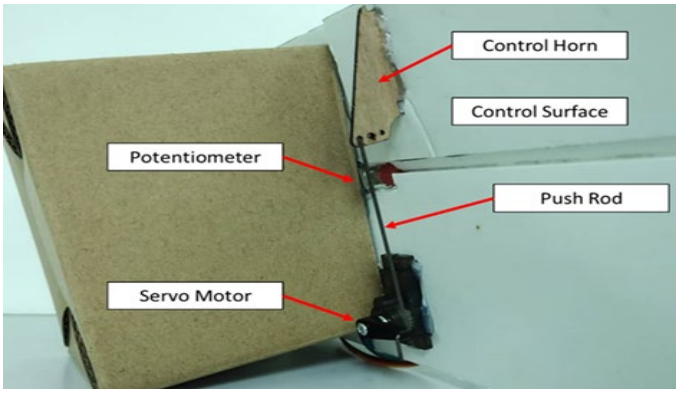


Figure 3. Bottom view of the test bench showing connection from servo motor to control surface

In Figure 3, the connection between servo motor and control surface will create at least a first order response. This has been discussed previously in [10]. The control surface position can be altered by

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

introducing some force on it which can be equivalent to the high wind speed perturbation effect. For this project, the force for alteration is done using the 100 g load.

2.2 Methodology

The methodology for this project is constructed based on the defined objectives. The process flow for implementation of this project is as shown in Figure 4.

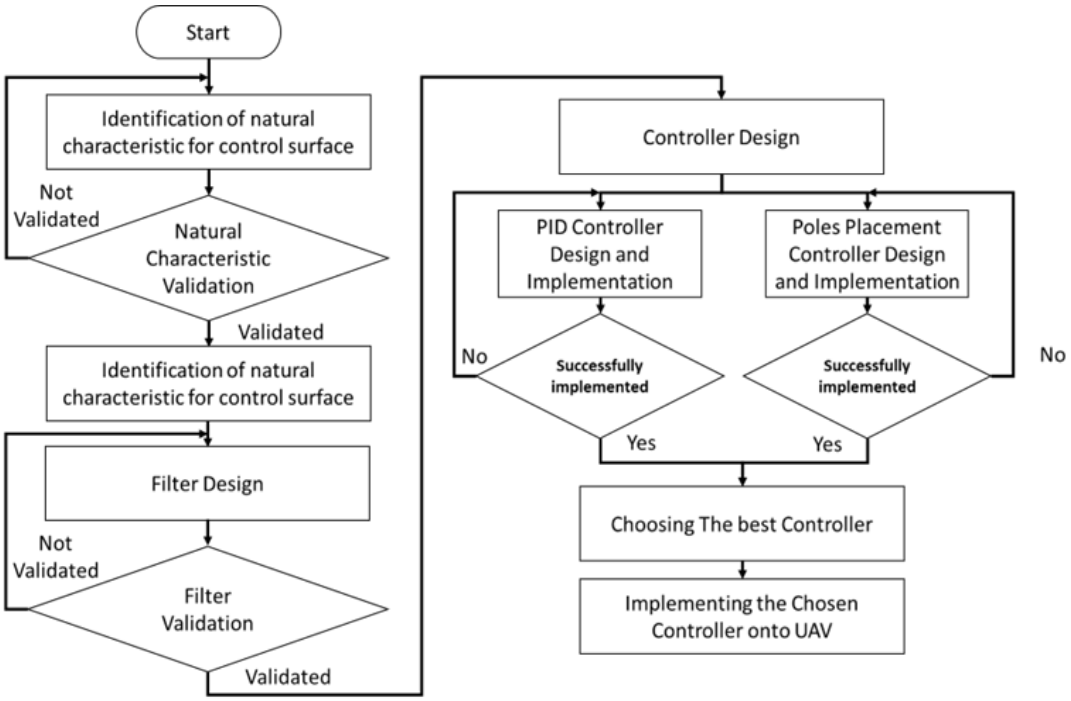


Figure 4. Project flow chart

In Figure 4, the process flow will begin with identifying the natural characteristic of the test bench control surface. Follows by designing and implementing filter to the test bench. After that, two methods of control laws design will be studied for the control surface to have a perturbation rejection capability.

2.2.1 Identification of Natural Characteristic for the Test Bench Control Surface

The identification process will be done based on the data collected from the test bench. Before the system identification process is executed, a test will be done to examine the movement of the control surface on the test bench and capability of the control surface to reject perturbation. The movement test will be done by moving the control surface and observe the potentiometer reading while the perturbation rejection test will be done by imposing the 100g load onto the control surface and observe the potentiometer readout while moving the control surface. The system identification process will only be done if the open loop control surface cannot reject perturbation. The result from the identification process will be in the form of a transfer function representing the natural characteristic of the control surface deflection for the test bench. This is also known as the open loop system for the test bench. The input for the test bench will be the PWM signals from the receiver and the output will be the potentiometer reading representing the control surface deflection.

The system identification will be done using MATLAB system identification toolbox [11]. Two sets of data will be used for identification process, which one is used for identification, and one is used for validation. The whole process of system identification will also give the operating range to the system which is needed in designing the controller. The operating range will be the PWM input range and the potentiometer output range. There will be a few results gained in the System identification toolbox. However, the results with the highest fitting percentages will be selected as the open loop system representing the test bench.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

2.2.2 Filter Design

The filter will be used to filter the potentiometer data and implemented into Arduino Nano. There is multiple type of filter design that can be done for this project including MDFT filtering and B-spline filtering [12-13]. However, this filtering method is complicated and difficult to be implemented onto a low-cost microcontroller module such as Arduino Nano. Therefore, a simpler method of filtering will be used for this project. The filter for this project will be design in continues or s-domain and converted into discrete or z-domain for implementation using zero-order-hold method [14]. This method is chosen due to its simplicity with assumption that the transfer function in s-domain will have the same response in z-domain. The process for filter design are as follows:

a. Analysing the frequency spectrum from the collected data using Fourier transform.
b. Create a transfer function for the desired filter in s-domain.
c. Convert the filter transfer function in s-domain into z-domain.
d. Test the converted filter in MATLAB.
e. Implement the filter in Arduino.

The filter chosen will be a low pass filter whether it is first order or second order filter. The best filter will be chosen in terms of the filter capability to reduce the noise without compromising the actual value in terms of delay and phase shift.

2.2.3 Control Law Design

There are two types of controllers that will be tested and compared in this project. The first controller is the PID controller, and the second controller is the feedback controller using poles placement method with integrator (later will be known as feedback controller with integrator). Both of this controller is chosen for their characteristic that able to reject perturbation and have no steady state error. Both controllers will be designed, simulated, implemented into test bench, and validated accordingly. The controller design will be based on the plant that consists of the open loop system representing the test bench connected in serial with the chosen filter. The plant is illustrated as in Figure 5.

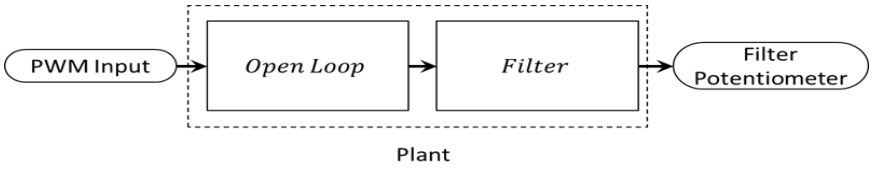
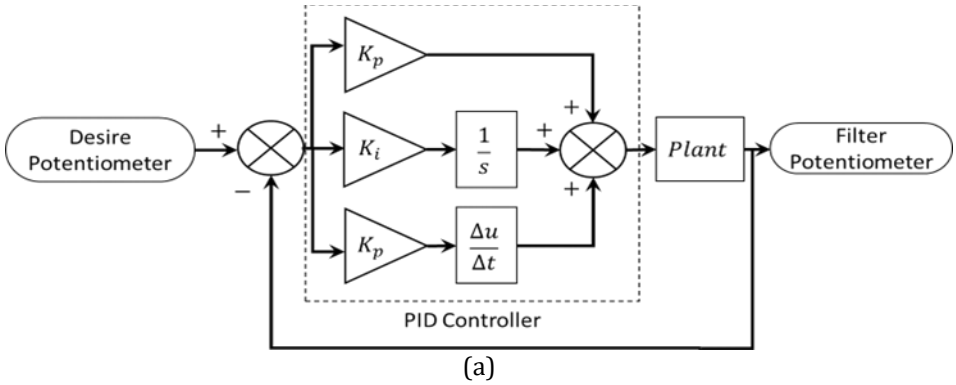


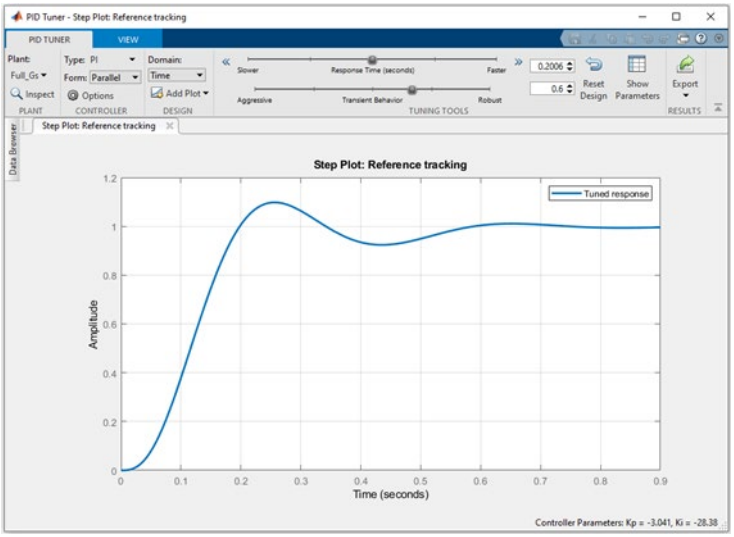
Figure 5. Plant for controller design

The PID controller will be designed based on diagram in Figure 6 with value of Kp, Ki, and Kd obtained using PID tuner apps in MATLAB.



(a)

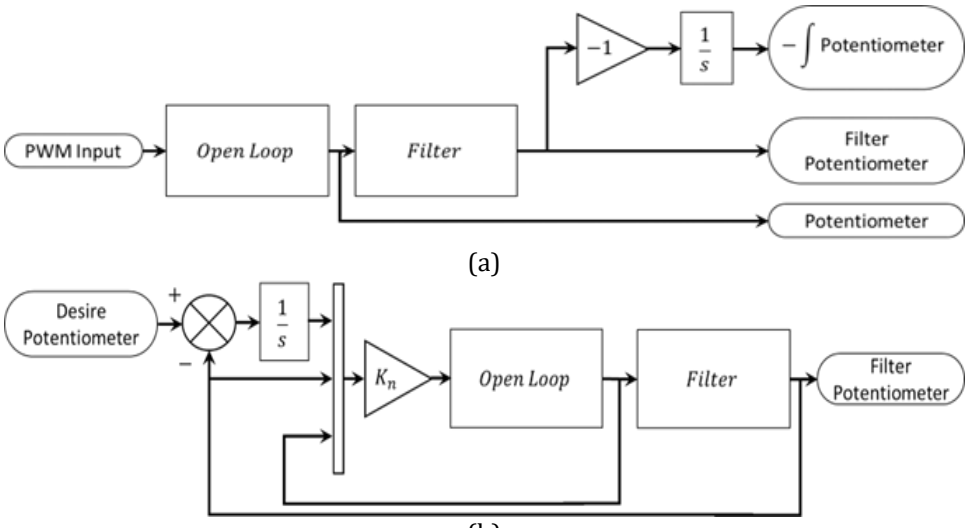
*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com



(b)

Figure 6. PID controller design: (a) block diagram for PID controller; (b) MATLABs PID Tuner environment

The feedback controller with integrator will be designed and implemented based on diagram in Figure 7. The implementation of this controller will have two feedback in the system.



(a)

(b)

Figure 7. Feedback controller with integrator design: (a) open-loop diagram; (b) closed-loop diagram

The feedback gain, K_n in Figure 7(b) will be in the form of 1 x 3 matrix to cater for 3 inputs with one output. The integrator for both PID controller and feedback controller with integrator will be done using trapezoidal rules. Both PID controller and feedback controller with integrator will be designed with limitation set from the system identification and validated through simulations and on test bench testing.

2.2.4 Controller Validation

Both designed PID controller and feedback controller with integrator to be validated in terms of their performance especially for their responses, capability to reject perturbation, and steady state error. The performance of the design controller is done by comparison with the open loop system. The validation will be conducted through simulation and on test bench testing. For simulation validation, the designed model will be put into Simulink and tested. The testing will be done to evaluate the response time and ability for the designed controller to reject perturbation. The Simulink diagram for simulation validation is as shown in Figure 8.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

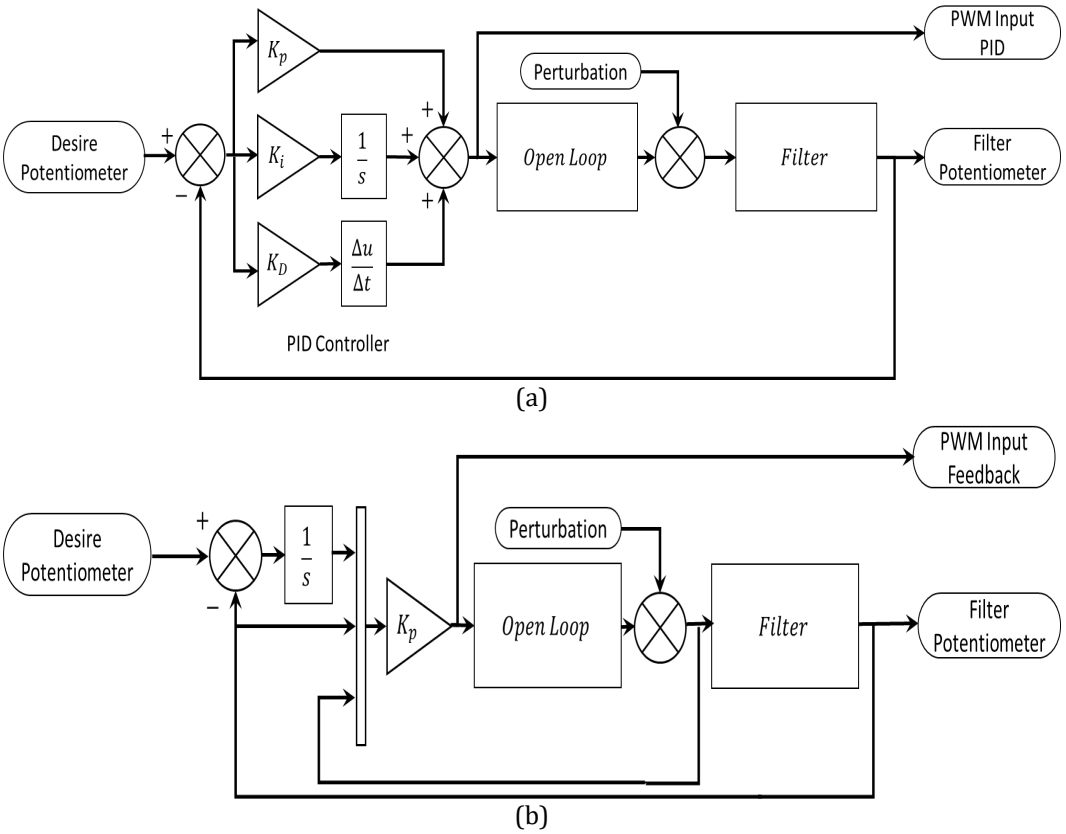


Figure 8. Simulink diagram for validation: (a) PID controller design; (b) feedback controller with integrator design

In Figure 8, the perturbation is given to the potentiometer output before filtering be-cause the potentiometer output is connected directly to the control surface and will be affected directly by the external perturbation. For validation on the test bench, both controllers will be programmed into Arduino on the test bench. Two tests will be done for this part of validation which are the movement test and the load test. The movement test is done by observing the potentiometer reading as well as the physical movement or vibration of the control surface during its deflection. This will validate the effect of controller and filter design in previous steps. The load test is done to validate the capability of the design controller to reject perturbation. The load chosen for this test is in weight of 100g. This will simulate a perturbation estimated around 0.1N acting on the control surface.

3.0 RESULTS AND DISCUSSION

The results of this project will be presented based on the methodology discussed previously as well as any issues that have been encountered during the whole process. The results will include the transfer function obtain for the open loop modelling, transfer function and equation for filter, equation for PID controller and feedback controller with integrator, and validation result using simulation and on test bench.

3.1 Open Loop Testing

There are two open loops testing that had been done. The first test is done to perceive any irregularities occur during movement of the control surface. The test is done by observing the PWM input and potentiometer reading while the control surface is moving. The results from this test are as shown in Figure 9. In a first glance, Figure 9 shows that the relation between PWM signals with is inversely proportional the potentiometer reading. This relation could impose some problems in designing the control laws later. It is suggested to invert the reading before continuing the next step. This can be done by a simple mirroring technique at the centre value of the potentiometer reading. The mathematical equation for this method is as shown in (1) or (2).

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

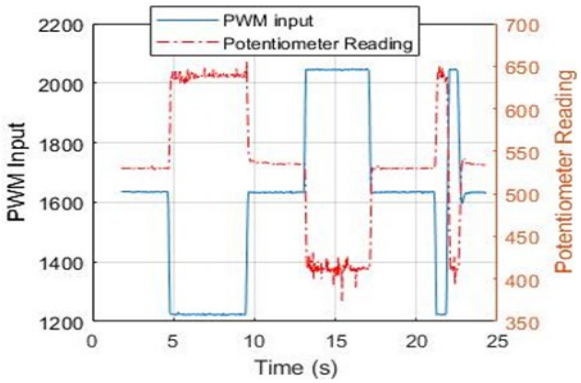


Figure 9. Result for open loop movement test

$$P_{new} = P_{centre} - (P_{old} - P_{centre}) \quad (1)$$

$$P_{new} = P_{centre} - P_{old} \quad (2)$$

The value for centre position for the potentiometer, P_{centre} is equal to 533. The results from implementation (1) are as shown in Figure 10.

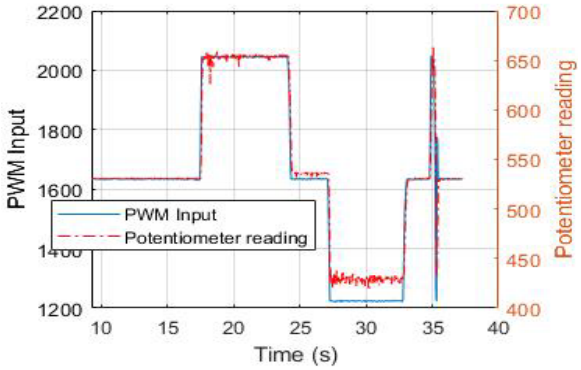


Figure 10. Mirrored collected data from test bench

In Figure 10, the PWM input and the potentiometer output are now positively proportional to each other. The next step is to identify the operating range for the test bench. The range for potentiometer as well as the PWM input is obtained based on the maximum stretch of the servo motor arm can travel. This is done by manually imposes a PWM length in Arduino and observe the servo arm stretch and control surface deflection. The operating range obtained for the test bench is as shown in Table 2.

Table 2. Operating range for test bench

	Minimum	Maximum
PWM Input	900µs	2600µs
Potentiometer reading	390	690

The operating range values obtained in Table 2 will be the limitation for designing the control laws later. For the design and testing purposes, the potentiometer reading is set to range from 400 to 600 only. This will give some room for the output to overshoot if required.

The next test is the load test for open loop control surface movement. The 100g load is loaded onto the control surface and PWM is given to move the control surface. The result of this test is as shown in Figure 11.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

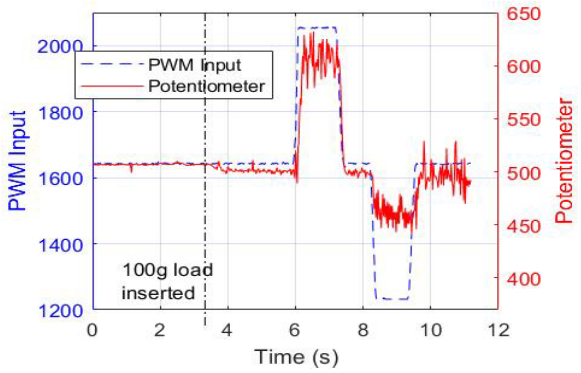
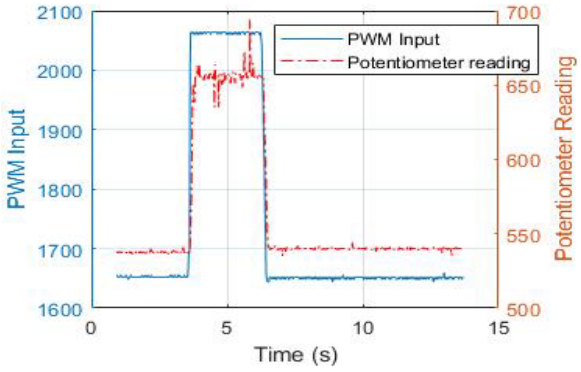


Figure 11. Result for open loop movement test with 100g load

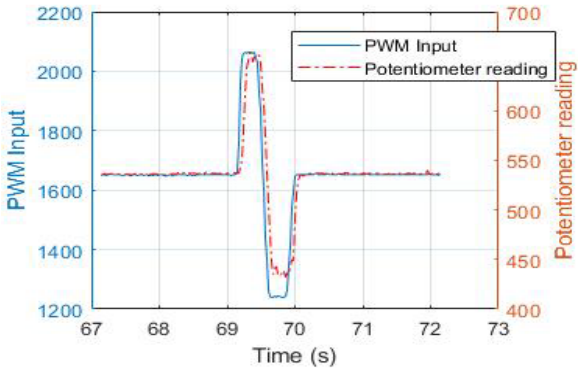
In Figure 11, the 100g load is added onto the control surface at a time of 3.4 s. The 100g load immediately affected the potentiometer reading by pulling down the control surface. Compared to no load movement in Figure 10, movement with load in Figure 11 shows that there is significant vibration induced by the load on the control surface. The system identification process will be done using data in Figure 10 where the potentiometer reading has been altered and there is no load present on the control surface.

3.2 Results for Identification of Natural Characteristic for the Test Bench Control Surface

The identification process is done by first observing the PWM input and potentiometer output. Throughout this project, the sampling time for Arduino is set to be at 0.035 s. The system identification process requires two sets of data for identification and validation. These data for identification are as shown in Figure 12.



(a)



(b)

Figure 12. Data of PWM input and potentiometer reading versus time: (a) Data used for system identifications; (b) Data used for validations

The data in Figure 12 are pre-processes by removing their mean value before been put into the system identification process. The results from system identification are as shown in Figure 13 with a fitting percentage of 89.47%. The transfer function from the results is as in (3).

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

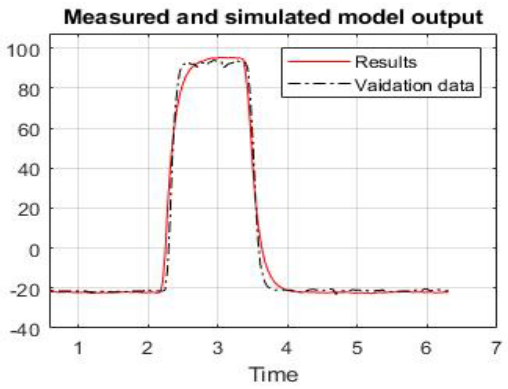


Figure 13. System identification results

$$G(s) = \frac{0.2842}{0.06006s + 1} \tag{3}$$

The results from the system identification show a first order transfer function as the response time is fast and there are no overshoot present in the plotted data. The block diagram for the control surface deflection is as shown in Figure 14 and this model will be used for controller design later.

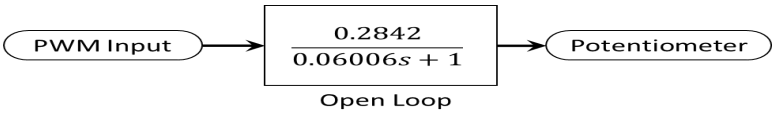


Figure 14. Test bench block diagram

3.3 Results for Filter Design

The filter design process is done starting by evaluating the frequency spectrum of movement for the control surface on the test bench. The frequency spectrum is taken from the movement in Figure 10 with the results in Figure 15.

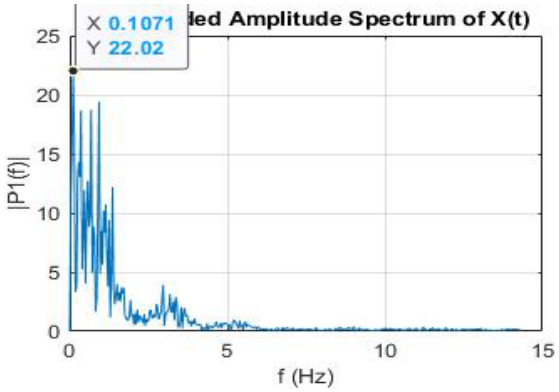


Figure 15. Frequency spectrum for control surface movement

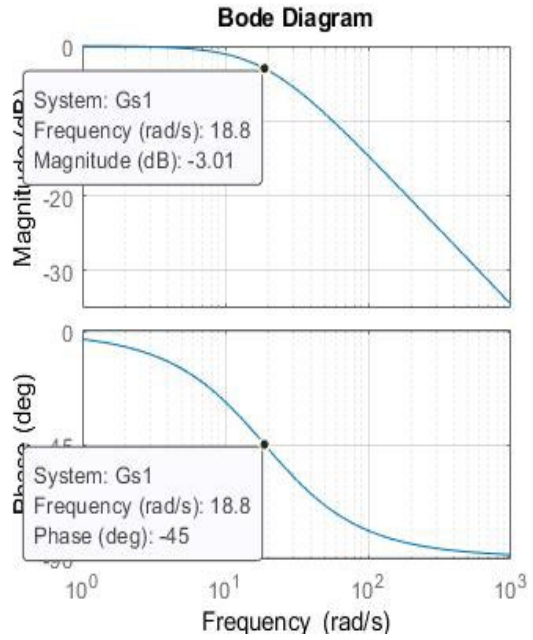
In Figure 15, the range of frequency with high amplitude is between 0 Hz to 3 Hz with the dominant frequency at 0.1 Hz and amplitude of 22.02. This is due to the fastest possible movement of the control surface deflection commanded by the remote controller. Study shows that human movement is around 1.27 Hz but for this project the cut-off frequency is selected to be at 3 Hz [19]. Two types of filters will be studied in this paper which are the first order low-pass filter and the second order low-pass filter. For the second order filter, the damping ratio is set at 0.8 to reduce the amount of overshoot. The transfer function for the first order and second order filter with cut-off frequency of 3 Hz is as shown in (4) and (5) respectively.

$$G(s) = \frac{18.85}{s + 18.85} \tag{4}$$

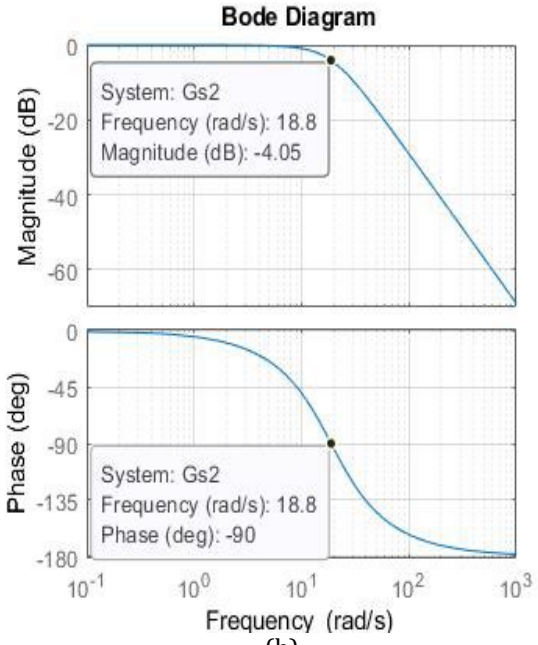
*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

$$G(s) = \frac{355.3}{s^2 + 30.16s + 355.3} \quad (5)$$

The transfer function in (4) and (5) have a Bode diagram as in Figure 16.



(a)



(b)

Figure 16. Bode diagram for the selected continuous filter design: (a) Continues First order low-pass filter; (b) Continues Second order low-pass filter

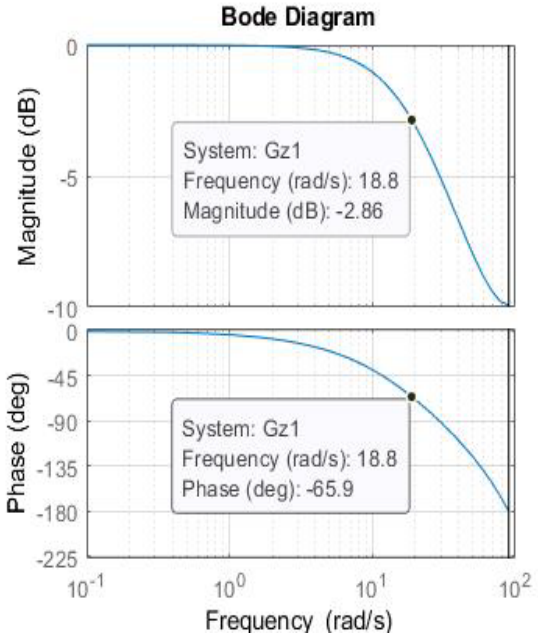
Figure 16 shows that the cut-off frequency for both filters is set to be at 18.8 rad/s or at 3 Hz. The transfer function in (4) and (5) are then converted into z-domain transfer functions as in (6) and (7) using the zero-order hold method with sampling time of 0.035 s. This is because only the filter in z-domain can be implemented into the microcontroller.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

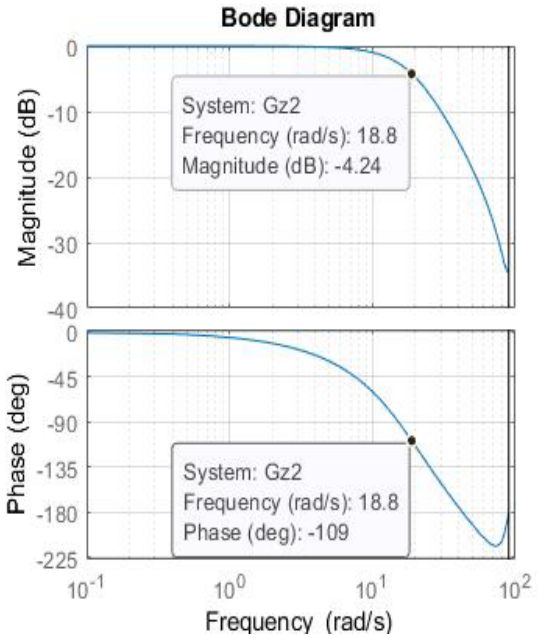
$$G_1(z^{-1}) = \frac{0.4813z^{-1}}{1 - 0.5169z^{-1}} \quad (6)$$

$$G_2(z^{-1}) = \frac{0.1525z^{-1} + 0.107z^{-2}}{1 - 1.088z^{-1}0.348z^{-2}} \quad (7)$$

The z-domain transfer function for both filters will have a Bode diagram as in Figure 17.



(a)



(b)

Figure 17. Bode diagram for the selected discrete filter design: (a) Discrete First order low-pass filter; (b) Discrete Second order low-pass filter

In Figure 17, the Bode diagram for both filters in z-domain transfer function shows that the discrete filter will be able to filter the output at 3 Hz (18.8 rad/s). However, the phase shift for both filters will be greater than the filter in s-domain (-65.9^0 compared to -45^0 for first order, and -109^0 compared to -90^0 for

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

second order at cut-off frequency). These shifts will create a significant delay in the output and need to be taken into consideration. Both filters in z-domain are simulated in MATLAB using data from Figure 10 with results as in Figure 18.

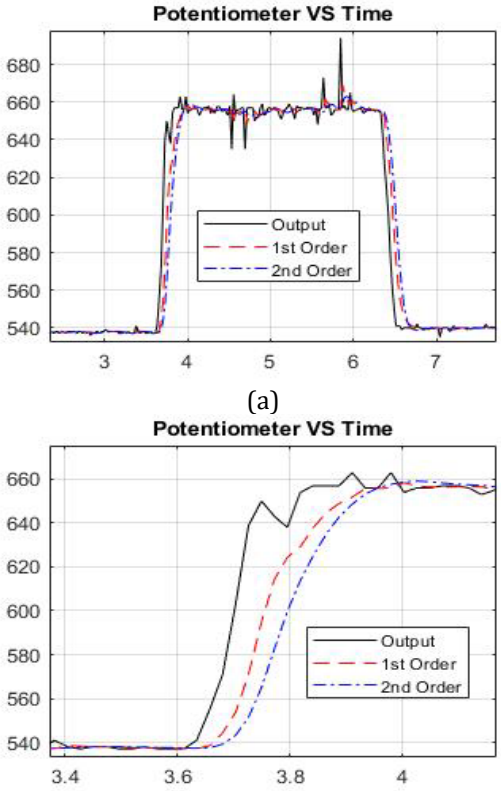


Figure 18. Simulated results for filter implementation: (a) First and second order low-pass filter; (b) Zoomed in view First and second order low-pass filter

The simulated results for filtering in Figure 18 show that the discrete second order filter will have a significant delay compared to discrete first order filter. This may be from the effect of converting the filter from continues to discrete domain as mentioned before. Thus, the first order filter is chosen to be implemented into the Arduino. The result of the implementation is as shown in Figure 19.

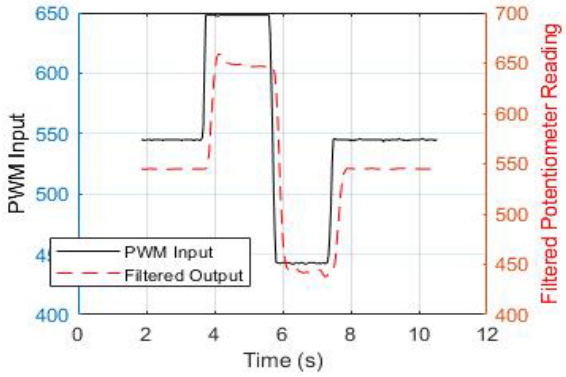


Figure 19. Results for filter implementation onto the test bench

In Figure 19, the potentiometer reading has been successfully filtered using the discrete first order filter with the delay from the phase shift is clearly seen in the graph. The filter that will be used in designing the control laws will be the continuous first order filter with the assumption that it is equivalent to the discrete filter. Thus, the block diagram for the filter is as shown in Figure 20 and this will be used in the designing stage later.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

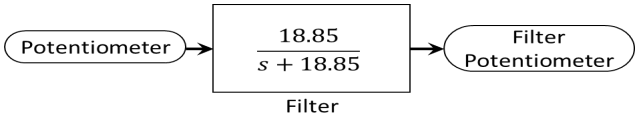


Figure 20. Filter block diagram

3.4 Control Law Design

Before the controller design process took place, the natural characteristic of the open loop system for the test bench needs to be acquired. This is done by examining the time do-main as well as the frequency domain parameter for the plant model in Figure 21.

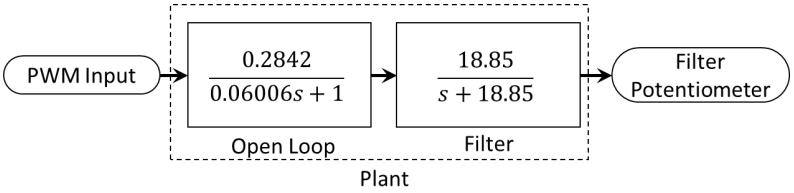


Figure 21. Plant for the controller design

The block diagram in Figure 21 shows that the plant for this project consists of the open loop representation of the system and the filter. The time domain (step response) plot for the plant is shown in Figure 22 with its parameter in Table 3.

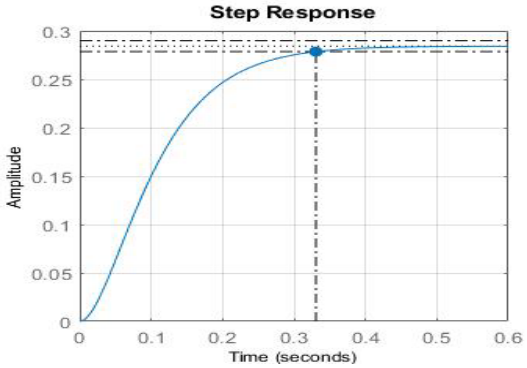


Figure 22. Step response for the plant

Table 3. Parameter for plant step response

Parameter	Value
Settling time	0.331 s
Rise time	0.19 s
Steady state	0.284
Percent overshoot	0
Poles	-18.85, 16.65
Damping pole 1	0
Damping pole 2	0

The controller will be designed to have a settling time equal or better than the plant response. This means that the settling time for the closed loop system should be less than or equal to 0.331s. The damping ratio for the controller is set to be higher than 0.7 for the system to have a percentage overshoot less than 4.6 %. The frequency domain (Bode diagram) plot is shown in Figure 23 with its parameter in Table 4.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

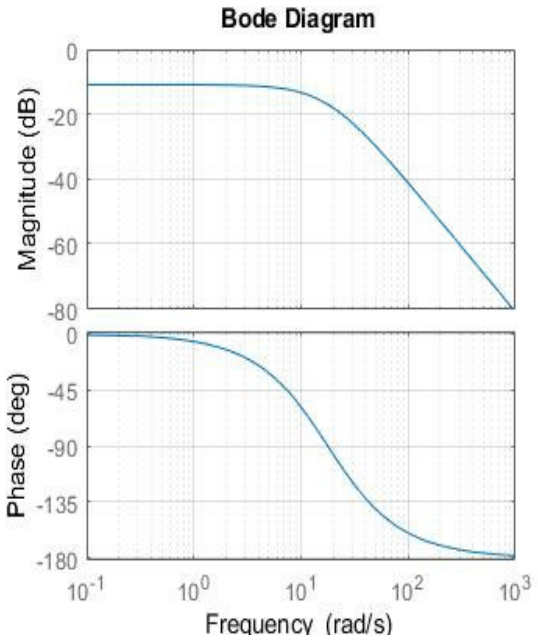


Figure 23. Frequency response for the plant

Table 4. Bode diagram parameter for plant

Parameter	Value
Frequency	16.7, 18.9 (rad/s)
Phase margin	Inf
Gain margin	Inf

From the plant frequency domain analysis, the plant will have phase margin and gain margin of infinity. This indicates that the plant is robust and will be able to handle any gain and phase uncertainty or time delay. With all parameter controller designs set, the controller is ready to be designed.

3.4.1 PID Controller Design

The PID controller is designed using the PID tuner applications in MATLAB. The end state for the design is to obtain the three gains for the PID controller which are the K_p, K_i, and K_d gain. The plant for the PID tuner will be the open loop model in Figure 21 with the expected implementation of PID controller as shown in Figure 24.

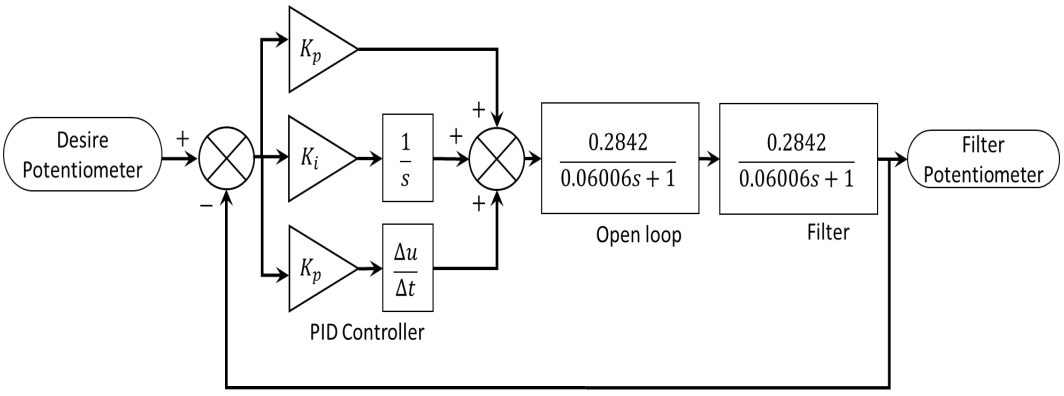


Figure 24. PID controller implementation

In Figure 24, the PID gains (K_p, K_i, and K_d) will be obtained using the PID tuner application in MATLAB. However, the validity of these gains depends on the input limitation to the plant where the PWM input must be in range between 900μs to 2600μs. If this limit is exceeded, the controller will not work on the test bench. The PID tuning process is done by trial and error. The PID gain is obtain first using the PID tuner

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

apps then tested in Simulink to validate whether the input limitation is exceeded or not. The first trial is done by setting the settling time of 0.3s and damping of 0.8. The results in PID tuner apps are as shown in Figure 25 with the PID transfer function in (8).

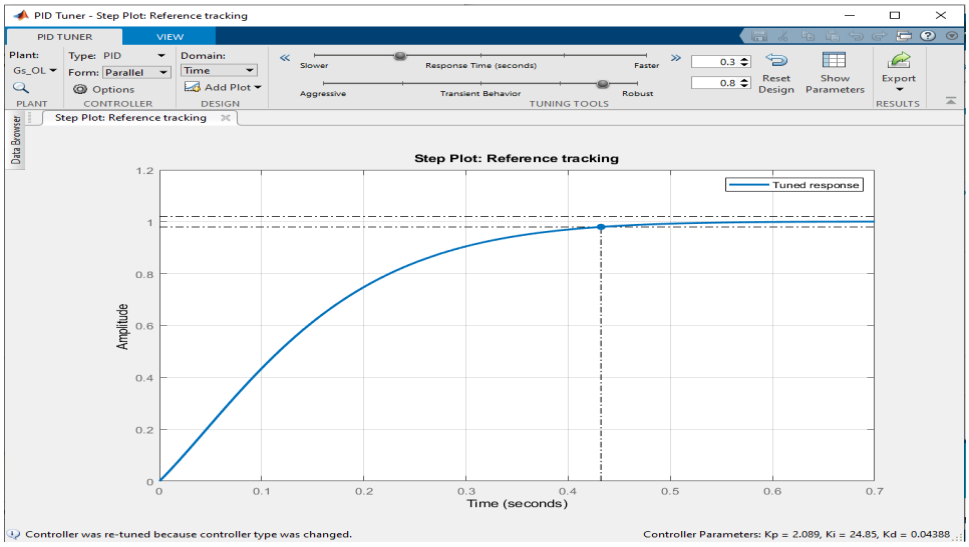
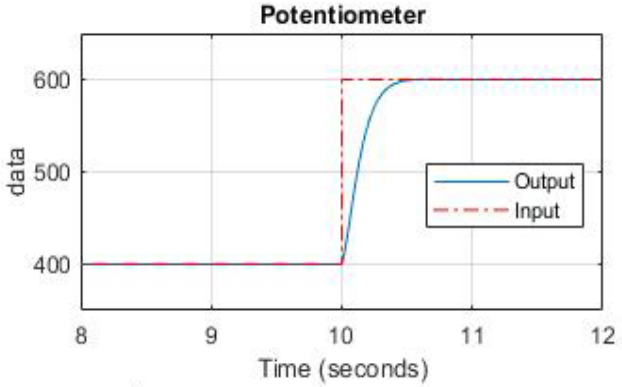


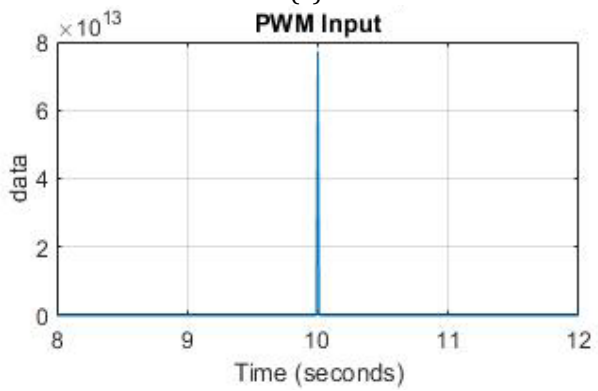
Figure 25. First trial using PID Tuner

$$\mathrm{PID}(s) = 2.089 + 24.85\frac{1}{s} + 0.04388s \tag{8}$$

The resulted PID controller in (8) is then tested in Simulink while the potentiometer output and PWM input are observed. The simulation is done for potentiometer input from 400µs to 600µs which is within the stated range. The simulation results are shown in Figure 26.



(a)



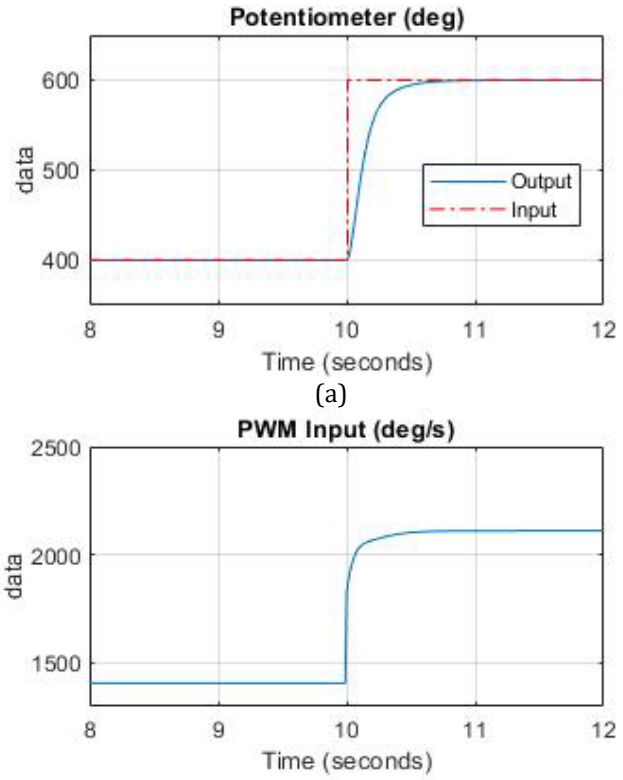
(b)

Figure 26. Simulated results for the first trial PID controller: (a) Potentiometer output response; (b) PWM input to the plant

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

In Figure 26, the potentiometer output response is in accordance with the desired parameter of settling time of 0.3s and damping of 0.8. However, the PWM input for the controller has exceeded the limitation. The PWM input should be in the range between 900µs to 2600µs but now it is range between 0 s to 7.8 x 1013s. Thus, the controller cannot be used for the test bench and a new PID gain value needs to be obtained. After a few times of tuning, a workable solution obtained is a PI controller (K_d = 0) where the settling time is set to be at 0.3s and damping set at 0.8. This will give a PI controller as in (9) with a simulation result shown in Figure 27.

$$\text{PID}(s) = 2.089 + 22.9\frac{1}{s} \tag{9}$$



(a)

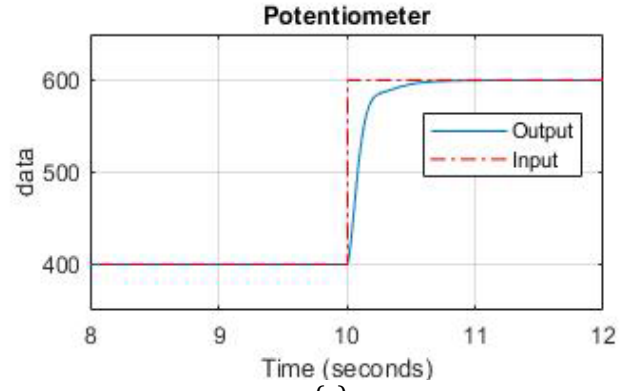
(b)

Figure 27. Simulated results PI controller with settling time at 0.3s and damping at 0.8.: (a) Potentiometer output response; (b) PWM input to the plant

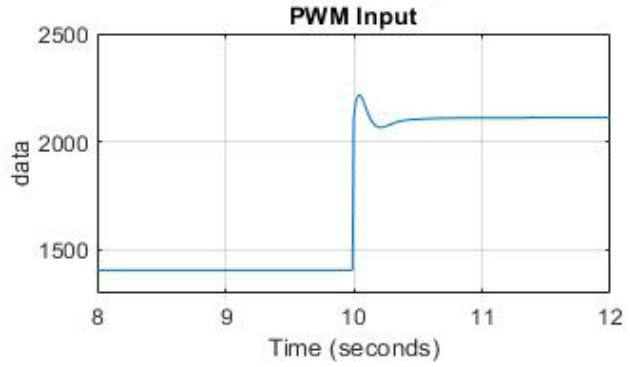
The simulation results in Figure 27 show that the output response with the PI controller implemented is better than open loop simulation. The PWM input is also within the stated range. Thus, the PI controller in (9) is suitable to be implemented into the test bench. The PI controller design is pushed further with settling time set to 0.2s while damping ratio maintains at 0.8. This will give a PI controller as in (10) with a simulation result shown in Figure 28.

$$\text{PID}(s) = 3.503 + 30.52\frac{1}{s} \tag{10}$$

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com



(a)



(b)

Figure 28. Simulated results PI controller with settling time at 0.2s and damping at 0.8.: (a) Potentiometer output response; (b) PWM input to the plant

The simulation results in Figure 28 show that the PI controller can be implemented as the PWM input for the plant is within the located range.

3.4.2 Poles Placement Method with Integrator

The poles placement method with integrator is done by first modifying the plant model with integrator. This is illustrated as in Figure 29.

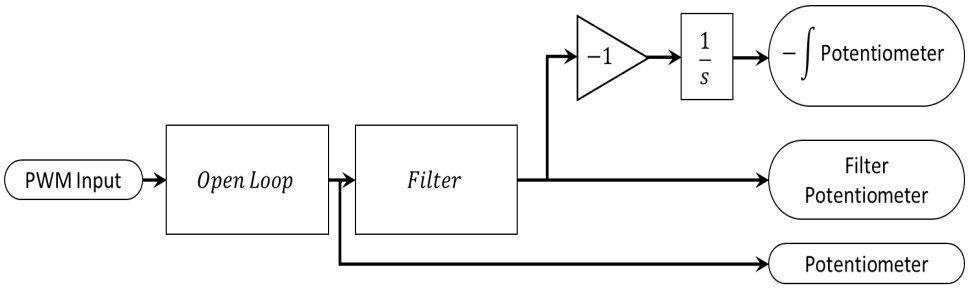


Figure 29. Modified plant with integrator

The model in Figure 29 will have a state-space representation as in (11) and poles location as in (12).

$$\begin{aligned} \dot{X} &= \begin{pmatrix} 0 & -18.85 & 0 \\ 0 & -18.85 & 4.732 \\ 0 & 0 & -16.65 \end{pmatrix} X + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} U \\ Y &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 18.85 & 0 \\ 0 & 0 & 4.732 \end{pmatrix} X \end{aligned} \tag{11}$$

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

$$p_1 = \quad 0$$
$$p_2 = -18.85 \tag{12}$$
$$p_3 = -16.65$$

The desire poles for the closed loop system are calculated to have a damping ratio of 0.8 and settling time at 0.3s. This means that the closed loop system should have a percentage overshoot of 1.52% and settle at faster speed than open loop system (at 0.3 s rather than 0.331 s). The desire poles are set as in (13).

$$p_1 = -13.33 + j10$$
$$p_2 = -13.33 + j10 \tag{13}$$
$$p_3 = \quad -20$$

The p_1 and p_2 are calculated based on the desired response, while p_3 can be changed accordingly. The value of -20 for p_3 is optimum with respect to the PWM input limitations. The model for feedback controller using poles placement method with integrator is as shown in Figure 30 with the feedback gain matrix calculated in (14). The simulation results for the feedback model are as shown in Figure 31.

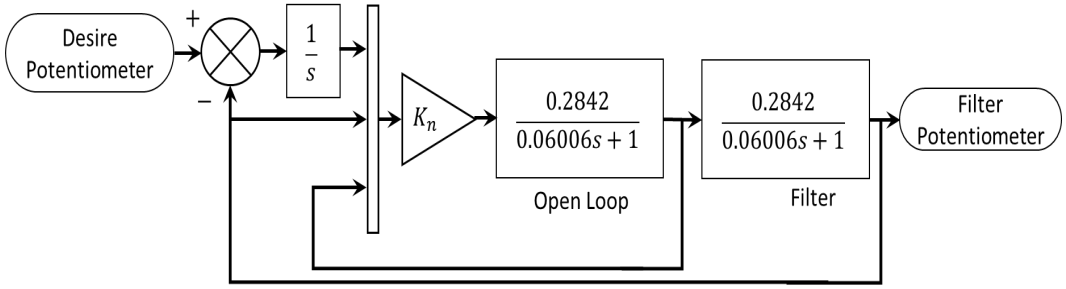
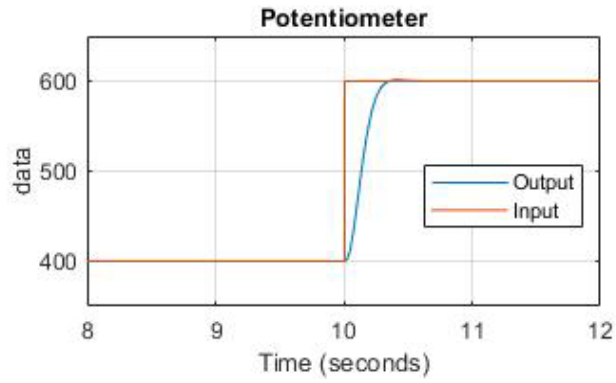
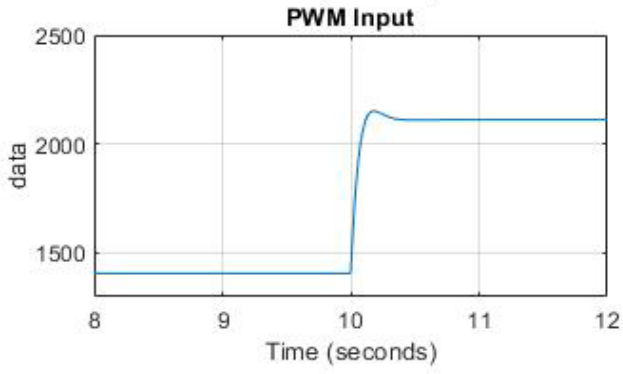


Figure 30. Feedback controller using poles placement method with integrator model

$$K_n = (62.2841 \quad -3.2150 \quad -2.3598 \tag{14}$$



(a)



(b)

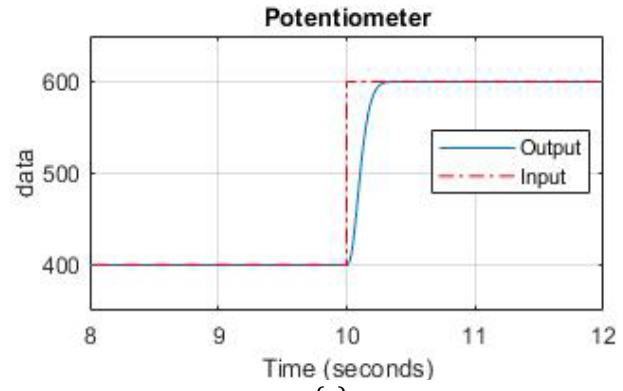
*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

Figure 31. Simulation results for feedback control with integrator with damping ratio of 0.8 and settling time at 0.3s. (**a**) Potentiometer output response; (**b**) PWM input to the plant

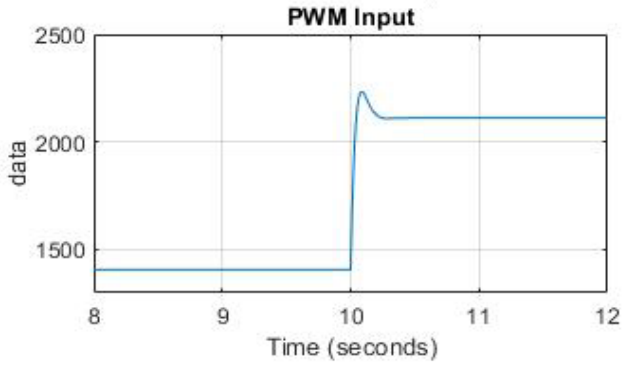
The result in Figure 31 shows that the response for the potentiometer output is accordance with the parameter sets for poles placement method. However, the output response seems to have a small overshoot which will be discussed later. The PWM input is between 1407μs to 2191μs. Which is within the range stated. Therefore, this feedback controller with the integrator is suitable to be implemented onto the test bench.

The feedback controller with integrator design is pushed further with settling time set to 0.2s while damping ratio maintains at 0.8. This will give a gain calculated as in (15) with a simulation result shown in Figure 32.

$$K_n = (140.1393 \quad -7.2796 \quad -5.1776 \tag{15}$$



(a)



(b)

Figure 32. Simulation results for feedback control with integrator with damping ratio of 0.8 and settling time at 0.2s. (**a**) Potentiometer output response; (**b**) PWM input to the plant

The result in Figure 32 shows that the response for the potentiometer output is accordance with the parameter sets for poles placement method. The output response is better compared to previous results where there are no overshoot present in the response. The PWM input is range between 1407μs to 2233μs. Which is within the range stated. Therefore, this feedback controller with the integrator is suitable to be implemented onto the test bench.

3.5 Simulation Comparison

The designed controller (PI controller and feedback controller with integrator) is compared in terms of their step responses and capability to reject perturbations. The com-pared step response for all designed controllers is as shown in Figure 33 with the step response parameter in Table 5.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

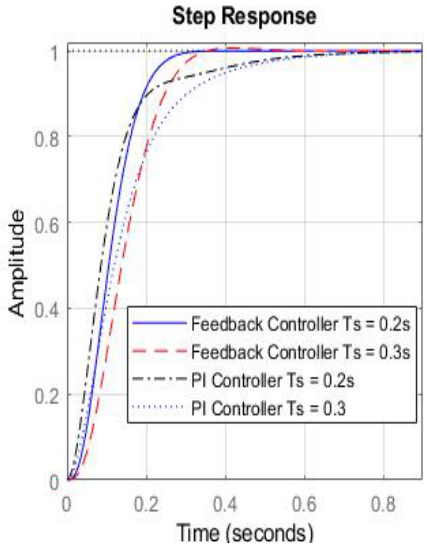
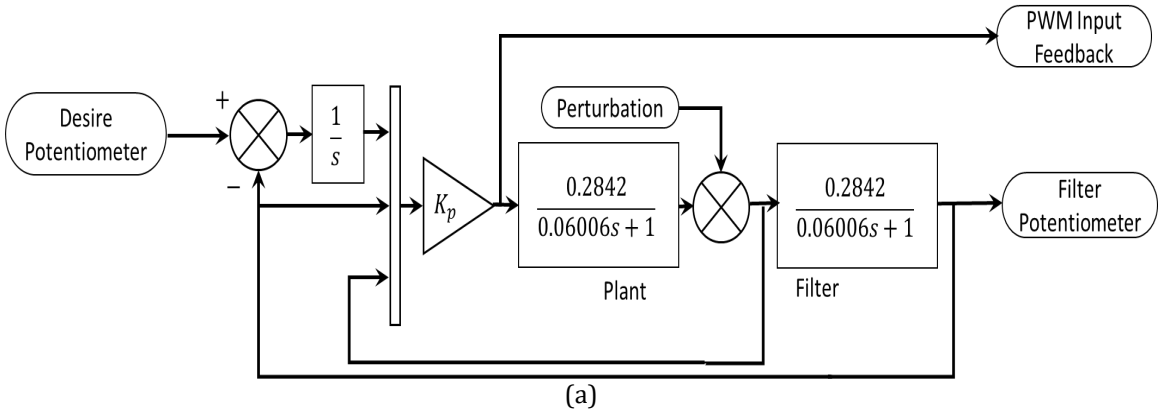


Figure 33. Step response simulation comparison for all designed controllers

Table 5. Step response parameter for designed controller

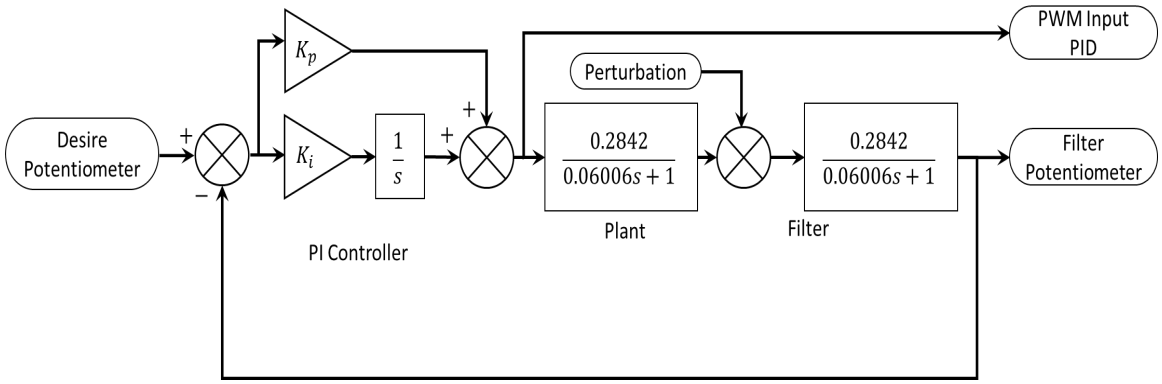
	PID Controller		Feedback Controller	
	$T_s = 0.2s$	$T_s = 0.3s$	$T_s = 0.2s$	$T_s = 0.3s$
Settling Time	0.512 s	0.55 s	0.255 s	0.311 s
Rise Time	0.177 s	0.266 s	0.148 s	0.187 s
% Overshoot	1	1	1	1.01
Steady State Error	0%	0%	0%	0.633%

From the comparison in Table 5, the feedback controller with integrator with settling time of 0.2s has a better output response compared with other controller designed. The comparison also shows that feedback controller is superior in terms of the settling time and rise time compared to PI controller. However, there is a slight overshoot for a feedback controller with integrator with settling time of 0.3s. This may cause problems for the UAV during flight and further investigation needs to be done physically. The next simulation test is to determine the perturbation rejection capabilities of all designed controllers. A perturbation is added to the simulation after the open loop block to simulate a wind force affected on the control surface. The block diagram with perturbation is as shown in Figure 34.



(a)

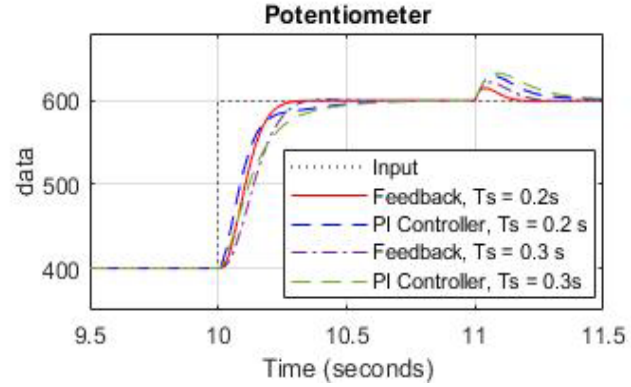
*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com



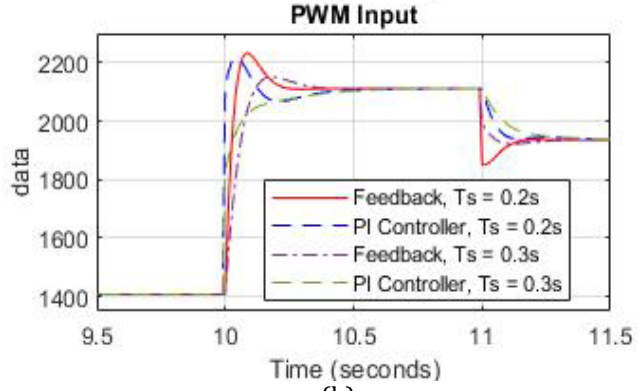
(b)

Figure 34. The block diagram with perturbation: (a) Feedback controller with integrator; (b) PID controller

The simulation in Figure 34 is set to have a step input of desire potentiometer value from 400 to 600 at time of 10s and perturbation with value of 50 (potentiometer reading) at time of 11s. The results for the simulation are as shown in Figure 35. The comparison results of all four controllers designed are as shown in Table 6.



(a)



(b)

Figure 35. Simulation results for perturbation rejection capabilities: (a) Potentiometer output response; (b) PWM input to the plant

Table 6. Comparison results for perturbation rejection capabilities

	PI Controller		Feedback Controller	
	$T_s = 0.2s$	$T_s = 0.3s$	$T_s = 0.2s$	$T_s = 0.3s$
Highest perturbation amplitude	629.1	632.6	614.8	621.7
Perturbation rejection time	>0.5s	>0.5s	0.18s	0.29s
% Overshoot	0	0	0	46.6

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

From the results in Table 6, the feedback controller with integrator with settling time of 0.2s has the best perturbation rejection capabilities followed by the feedback controller with integrator with settling time of 0.3 s. However, these simulation results need to be validated with physical implementation on the test bench before a real conclusion can be made.

3.6 Controller Validation Based on Test Bench

For this part, all the designed filter and controller will be implemented into the test bench for testing and validation. There will be two types of tests to be done, which are the movement test and the external perturbation test. The movement test is done by executing a step input and observing the output. This test is done to see the real output response com-pare to the simulation done previously. The external perturbation test is done to examine the perturbation rejection capability for both controllers in real applications. All four controllers are implemented into the Arduino along with the filtering method. The input for the system is initially a PWM length between 1000μs to 2000μs. This value needs to be converted into the desired potentiometer value from 400 to 600. This can be done by implementation (16). In (16), the value of 0.2 is the ratio between PWM ranges with the potentiometer range.

$$P_D = (PWM_{in} - 1000)(0.2) + 400 \quad (16)$$

The movement testing is done by varying the desired potentiometer value and observing the potentiometer read out. This is done for all controllers, and the results are as shown in Figure 36.

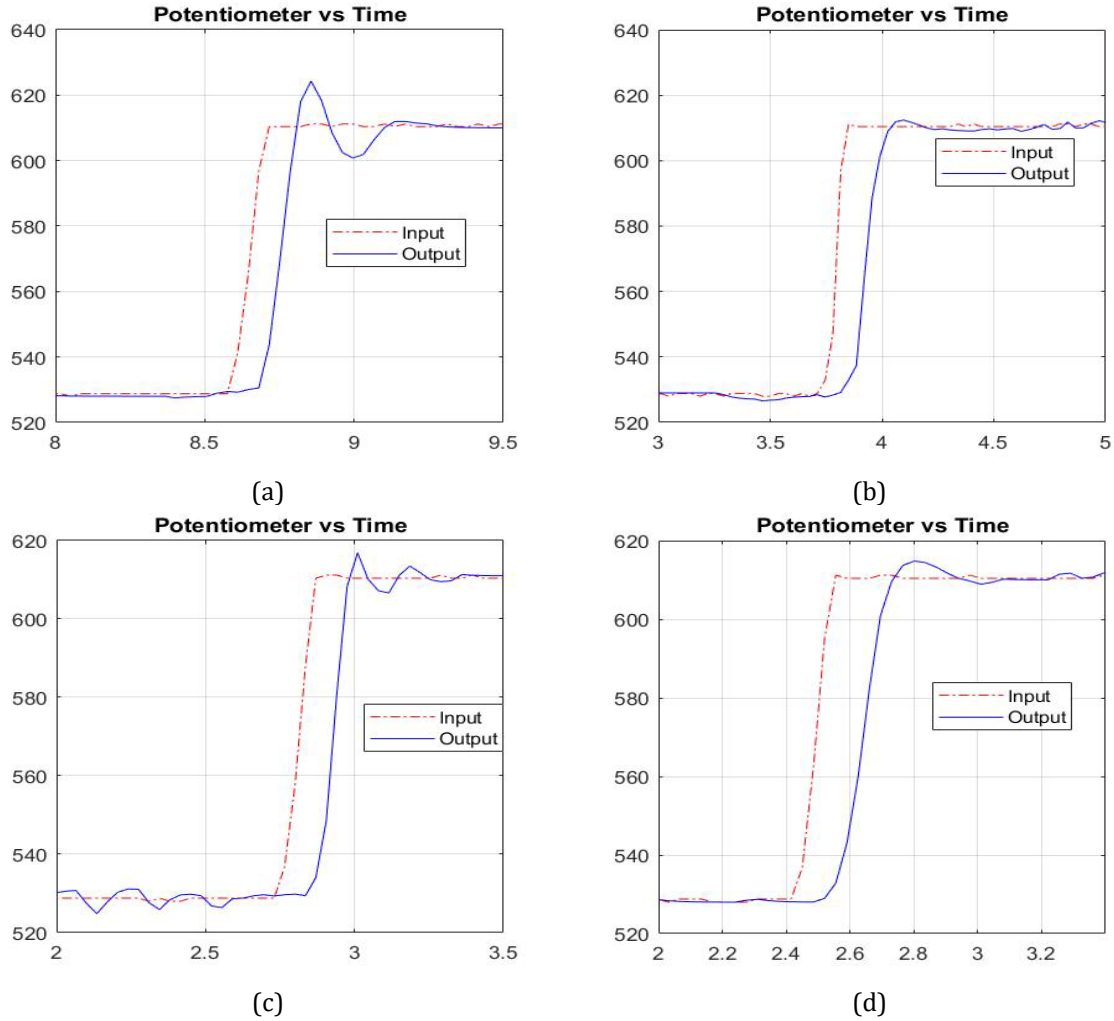


Figure 36. Movement test on test bench: (a) PI controller with settling time of 0.2s; (b) PI controller with settling time of 0.3s; (c) feedback controller with integrator with settling time of 0.2s; (d) feedback controller with integrator with settling time of 0.3s

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

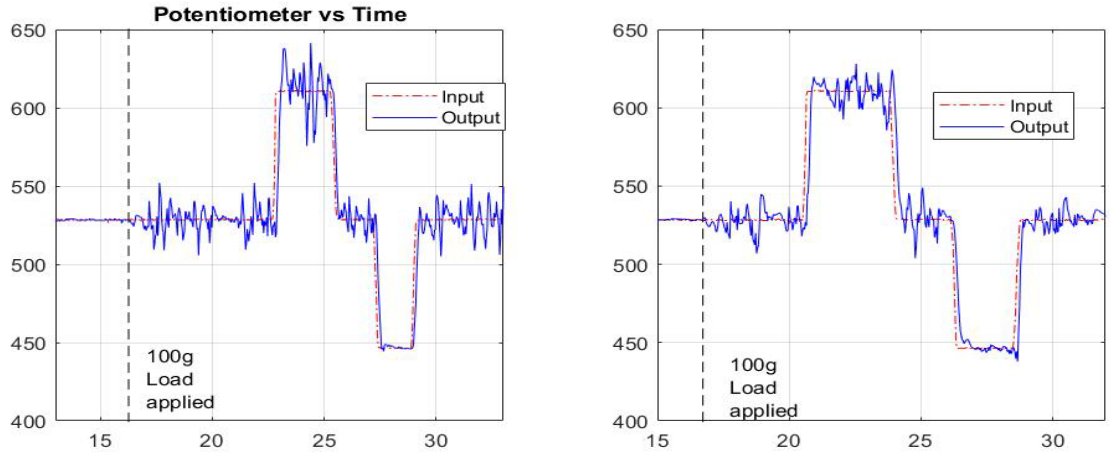
The results for PI controller movement test in Figure 36(a) and Figure 36(b) show that there is significant overshoot in the output response, and it is worse for settling time set at 0.2s where the overshoot is significantly high with long oscillation at almost 1s long. This is against the simulation results compared in Figure 35 where there should not be any overshoot for PI controller output response. This phenomenon may occur due to the approximation of the open loop modelling where the fitting percentage is not at 100%. Another reason for this is because of the approximation of filter design where in designing the controller, the filter uses is in continuous domain, where for implementation, the discrete domain filter were used. For the results for feedback controller with integrator for settling time of 0.2s movement test in Figure 36(c) shows that there are some oscillations throughout the output response while for settling of 0.3s in Figure 36(d), the output response shows some over-shoot.

Overall, both responses are in accordance with the simulation compared in Figure 35 despite the oscillation in Figure 36. The controller with settling time of 0.2s does not have any overshoot while the controller with settling time of 0.3s has slight overshoot. It seemed that approximation of open loop modelling and filter design has little or no effect on feedback controllers with integrators compared to PI controllers. One reason for this is because of the multiple feedback of this controller (in this case 3 feedback) which make it much more superior compared to PI controller. However, the multiple feedback also amplified the oscillation from the unfiltered output which resulted in oscillation in Figure 36. To further investigate the differences between the implementation of the designed controller with the simulation, the step response parameter is calculated manually in assumption that the input is stiff enough to be a step input. This will result in a response parameter in Table 7.

Table 7. Calculated parameter for step response of the implemented controller

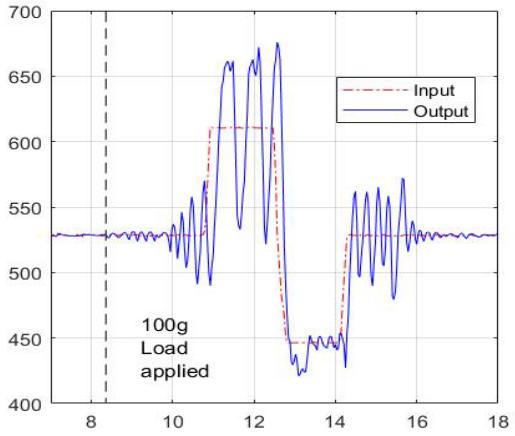
	PI Controller		Feedback Controller	
	$T_s = 0.2s$	$T_s = 0.3s$	$T_s = 0.2s$	$T_s = 0.3s$
Settling Time	0.35s	0.26s	0.15s	0.215s
% Overshoot	17.61%	2.46%	7.96%	5.39%
Steady State Error	0	0	0	0

Compare the calculated parameter of step response in Table 7 with simulated step response parameter in Table 5, the implementation has better settling time than the simulation. However, there is a significant increase in overshoot for every response where there should not be any overshoot except for feedback controller with integrator with settling time set at 0.3s. The highest overshoot is at 17.61% which occurs for the PI controller with settling time of 0.2s and this overshoot display itself significantly on the control surface physical movement. Overshoots for other designed controllers display little effect on the control surface physical movement and are not visible to naked eye. From the test result in Table 7, it can be said that the best controller for the test bench is the feedback controller with integrator with settling time of 0.2 s. This is due to its low settling time and percentage overshoot of less than 10%. However, these results need to be validated with the controller ability to reject perturbation. The perturbation rejection test is done by applying the 100g load onto the control surface and a PWM input is given to observe the movement of the control surface with the attached load. The result of the perturbation test on the test bench is as shown in Figure 37.

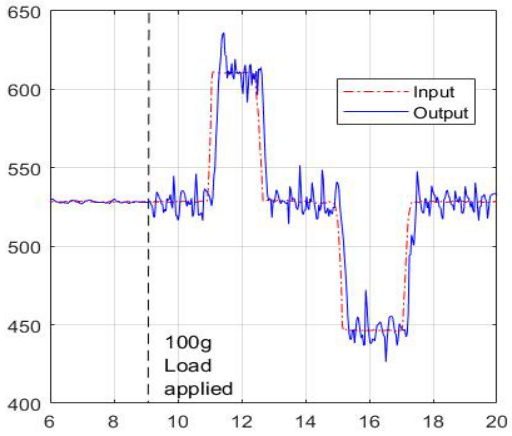


(a) (b)

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com



(c)



(d)

Figure 37. Movement test on test bench with 100g load attached as perturbation: (a) PI controller with settling time of 0.2s; (b) PI controller with settling time of 0.3s; (c) feedback controller with integrator with settling time of 0.2s; (b) feedback controller with integrator with settling time of 0.3s

The results from the load test in Figure 37 show that the 100g load will induce oscillation to the output response. The summary of the highest amplitude after overshoot is as in Table 8.

Table 8. Summary of the oscillation highest amplitude after overshoot

Controller	Oscillation highest amplitude
PI Controller, Ts = 0.2s	31.1
PI Controller, Ts = 0.3s	17.7
Feedback Controller, Ts = 0.2s	61.6
Feedback Controller, Ts = 0.3s	18.8

Summary in Table 8 shows that PI controller with settling time of 0.3s induce the smallest oscillation follows by feedback controller with integrator with settling time of 0.3s. The feedback controller with integrator with settling time of 0.2s shows that the multiple feedback from the controller design has induced the oscillation tremendously which is worst from the open loop system even though the system has better settling time compared to others.

3.7 Selection of the Suitable Controller Design

The selection for the most suitable controller for the test bench is done based on the movement test and load test as summarized in Table 6 and Table 7. This led to the best solution of PI controller with settling time of 0.3s. This is due to its parameter of overshoot of 2.46% and load induced oscillation amplitude of 17.7 which is the lowest among the design controllers. The settling for this controller is also among the fastest at 0.26s, which is still under 0.331s compared to the open loop system. The comparison between the selected controller with the natural characteristics is as shown in Table 9.

Table 9. Comparison between selected PI controller with the natural characteristics of control surfaces

	PI Controller	Natural Characteristics
Settling Time	0.26s	0.331s
% Overshoot	2.46%	0%
Steady State Error	0	0.716
Perturbation Rejection	Yes	No

Based on the comparison in Table 9, the chosen PI controller is better than the natural characteristics of the control surface test bench. However, the implemented PI controller introduces a small overshoot onto the system. This is a trade-off for better settling time and to have the perturbation rejection capability. Nonetheless, the effect of the overshoot toward the actual flight needs to be further investigated during actual flight.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

3.8 Consideration and Implications for Practical Implementation

Based on the simulations and test bench results, the most suitable controller to be used will be the PI controller with 0.3s settling time. However, there are several considerations that need to be considered before it can be implemented onto a real UAV. The considerations include:

a. Modelling for every control surfaces. The PI controller discussed in this paper only valid for the control surfaces test bench. For implementation onto a UAV, natural characteristics for all control surfaces need to be defined first.
b. Different values for PI gain. As the natural characteristics for every control surface might be differ, the values for PI gain will also be differ This will result longer code for implementation as different code are required for PI controller implementation for each control surfaces.
c. Delay Consideration. As been discussed earlier, the filtering will result in delay to the system. This delay will result in delay on the control surfaces deflection after the control inputs is requested. From the phase margin of the Bode Diagram, the system will still be stable event with existing delay, however this may affect the overall flight experience of the UAV and further testing are required.
d. Natural perturbation. From the load testing, the PI controller chosen will be able to withstand high load perturbation acting on the control surfaces. However, the vibration effect has not yet been considered in the design of the controller. This vibration will likely occur during flight from wind flow over the control surfaces which may have effect on the controller.
e. Hardware Constraint. The biggest constraint might be to mitigate the delay effect not only from the filtering process, but also from the coding implementation. Thus, the hardware constraint will come from the micro controller coding and implementation where delay might instigate.

4.0 CONCLUSIONS

Inconclusion, the control laws for UAV control surfaces have been successfully designed and tested on the designated test bench. The most suitable controller for the UAV control surfaces is the PI controller with a settling time of 0.3s. This controller has shown its ability to reject perturbation as well as maintaining a quick response time. However, there exist an overshoot which makes the control surface bounce before settling. This may result in disturbance of the airflow on the control surfaces and the wing of the UAV and need to be further tested during actual flight testing.

5.0 CONFLICT OF INTEREST

The authors declare no conflicts of interest.

6.0 AUTHORS CONTRIBUTION

Abdul Rashid, Z. (Main author, conducting MATLAB simulation and calculations).
Dardin, S. M. F. S. M. (Project supervisor, responsible for data collection and validations).
Ahmad, K. A. (Project supervisor, responsible for data collection and validations).
Abdul Azid, A. (Responsible for test bench setup and programming).

7.0 ACKNOWLEDGEMENTS

This study was not supported by any grants from funding bodies in the public, private, or non-profit sectors. The authors fully acknowledged Ministry of Higher Education (MOHE) and National Defence University of Malaysia (NDUM) which makes this important research viable and effective.

List of Reference

[1] Mazzoleni, M., Di Rito, G., & Previdi, F. (2021). Electro-mechanical actuators for the more electric aircraft (Vol. 2021). Cham, Switzerland: Springer.
[2] Smetana, F. O., & Covert, E. E. (2002). Flight vehicle performance and aerodynamic control. *Appl. Mech. Rev., 55*(1), B13-B14.
[3] Liu, M., Egan, G. K., & Santoso, F. (2015). Modeling, autopilot design, and field tuning of a UAV with minimum control surfaces. *IEEE Transactions on Control Systems Technology, 23*(6), 2353-2360.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com

[4] Valavanis, K. P., & Vachtsevanos, G. J. (2014). Handbook of unmanned aerial vehicles. Springer Publishing Company, Incorporated.

[5] Kumari, P., & Raghunath, I. (2016). Unmanned aerial vehicle (DRONE). *Int. J. Eng. Comput. Sci, 5*(6), 16761-16764.

[6] Rashid, Z. A., Dardin, S. M. F. S. M., Azid, A. A., & Ahmad, K. A. (2019, September). Perturbation Rejection Controller Design for UAV's Control Surfaces. *In 2019 4th International Conference on Control, Robotics and Cybernetics (CRC)* (pp. 7-11). IEEE.

[7] Özden, T., Tuğal, H., & Okumuş, H. İ. (2012, March). Control of an optimum modelled air heating system and real time simulation. *In 2012 16th IEEE Mediterranean Electrotechnical Conference* (pp. 940-945). IEEE.

[8] Pawar, K. S., Palwe, M. V., Ellath, S. B., & Sondkar, S. Y. (2018, August). Comparison of performance of PID controller and state feedback controller for flow control loop. *In 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)* (pp. 1-5). IEEE.

[9] Pusch, M., Knoblach, A., & Kier, T. (2019). Integrated optimization of control surface layout for gust load alleviation. *CEAS Aeronautical Journal, 10,* 1059-1069.

[10] Rashid, Z. A., Dardin, A. S., Azid, A. A., & Ahmad, K. A. (2018, February). Identification and modelling of flight characteristics for self-build shock flyer type UAV. I*n AIP Conference Proceedings* (Vol. 1930, No. 1, p. 020002). AIP Publishing LLC.

[11] Valoušek, L., & Jalovecky, R. (2021, June). Use of the MATLAB® System Identification Toolbox® for the creation of specialized software for parameters identification. I*n 2021 International Conference on Military Technologies (ICMT)* (pp. 1-5). IEEE.

[12] Karp, T., & Fliege, N. J. (1995, April). MDFT filter banks with perfect reconstruction. *In Proceedings of ISCAS'95-International Symposium on Circuits and Systems* (Vol. 1, pp. 744-747). IEEE.

[13] Hou, H., & Andrews, H. (1978). Cubic splines for image interpolation and digital filtering. *IEEE Transactions on acoustics, speech, and signal processing, 26*(6), 508-517.

[14] Zhang, Z., & Chong, K. T. (2007, October). Comparison between first-order hold with zero-order hold in discretization of input-delay nonlinear systems. *In 2007 International Conference on Control, Automation and Systems* (pp. 2892-2896). IEEE.

*Corresponding Author | Abdul Rashid, Z. | zuhairirashid@gmail.com