Control System Design: Tuning a P.I and P.I.D Controller

Control systems Laboratory

Module: Engineering Dynamics and Control Engineering E2

Course: Mechanical Engineering

Author: Joshua Jones

Tutor: Dr. A Jones

Last updated/submitted: March 28, 2014

Table Of Contents

Summary

The aim of the lab experiment was to gain an appreciation of P.I and P.I.D controller behaviour when applied to a real system, in this case a fan blowing air over a heating element and a temperature sensor at the end of the airflow, the PID controller is employed to control the heating element to produce a desired temperature at the end of the airflow, this is done by measuring the error of a measured value compared to it's set point, in context the current temperature, the desired temperature, and the error between them.

The broad aims were to investigate feedback control (A closed loop system), explore the behaviour of the P.I.D controller, estimate the tranfer function values through iterative experiment, and to compare with simulink simulations.

1 Introduction

The broad aims were to investigate feedback control (A closed loop system), explore the behaviour of the P.I.D controller, estimate the tranfer function values through iterative experiment, and to compare with simulink simulations.

1.1 Open Loop System vs Closed Loop System

And open loop system has an input, a transfer function, and an output, this system is controlled by varying the input, and is not autonomous, the output has no effect on the input.

However in a closed loop system, there is an input, a transfer function, and an output, however a feedback element is included that adjusts the input, reactively based on the output, in this way a desired output can be determined and the input can be varied in a much more controlled, precise, and autonomous way.

The experimental control system in this lab was a closed system feedback control system that made use of the error between the current value and the set point value for the temperature reading of the test cell, this value of error was incorporated proportionally, integrally and differrentially making use of a P.I.D controller.

2 Theoretical Background

2.1 Control Systems

The Process variable (Also known as the control variable) is the variable being controlled by the control system, in this case the process variable is the temperature at the end of the airflow, which is controlled by the heating element, which in turn is controlled by the supply voltage to that element, this is controlled by the P.I or P.I.D Controller. The current temperature is whatever is measured at that point in time, and the desired temperature specified is known as the set point.

Setpoint is the desired process output that an automatic control system will aim to reach. For example, a boiler might have a temperature setpoint,

that is the temperature the control system aims to attain.[1]

For the scope of this system the aim of the control system was to reach the setpoint as quickly as possible with the least amount of deviation (overshoot/undershoot)

The most simple type of control is a binary (On/Off) system to turn a component on or off when the control variable deviates too far from the setpoint, however the system for this application is controlling the voltage to the heating element, and can provide control in a more analogue (as opposed to descrete/binary control) thus allowing for quicker response and more incremental adjustment than the two extremes of (On) or (Off).

In feedback control, it is standard to define the error as the difference between the desired value (setpoint) y_S and the current value (measured) y. If the error is large, then the control action is large.

2.2 What is a P.I Controller

P.I stands for Proportional, Integral controller, this controllers output is determined by 2 values, one value proportional to the gain, and one value which is the integral of the gain.

2.3 What is a P.I.D Controller

P.I stands for Proportional, Integral controller, Derivative this controllers output is determined by 3 values, one value proportional to the gain, one value which is the integral of the gain, and one value which is the derivative of the gain.

2.3.1 How are P.I.D Control Loops used

Many real world processes build up over time. When you step on the accelerator, your car moves slowly, then faster, and faster still, until you let off the gas pedal. As you speed up, you press the gas pedal less, then a little less, then even less, until you reach the speed limit. Unlike the digital world where things are either “on” (1) or “off” (0), real processes have varying degrees of “on”. In the driving example, how much the accelerator is turned

"on" depends on the car's current inertia and how different the car's speed is from the speed limit. Controlling such a process with inertia can be done with only five lines of code. But, just like learning to drive, it takes practice to know if you are starting too quickly, or if you'll overshoot the speed limit.

Cruise control is one example of a PID control loop. To calculate the output, it needs three factors. The first, (P), is the difference between the current speed and the desired speed. The second, (I), is the sum of the differences over time. And, the third, (D), is the rate of change between sampled differences. Each factor is scaled by a gain constant; they are refered to as Kp, Ki, and Kd. The value of these gain constants determines how responsive the output will be. If the Kp, Ki, and Kd values are too high, the output (car's speed) will far exceed the set point (speed limit). Set too low, the output may never reach the set point (like driving 40mph on the highway).[2]

2.4 Typical P.I.D outputs

In the images below, the green line is the set point, the blue line is the input (or process value), and the red line is the output (or manipulated value).

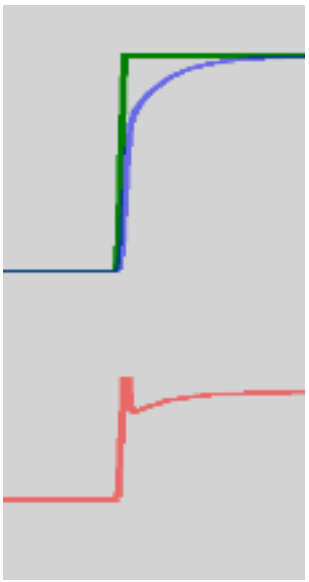

Figure 1: Correct control signal. All three gain constants are set correctly.

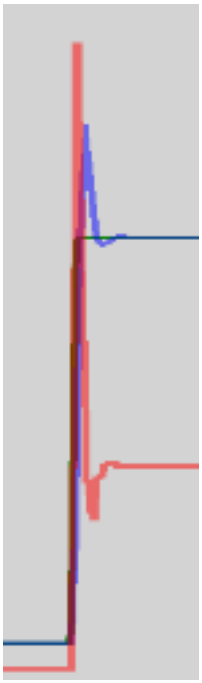

Figure 2: Overshoot. Integral constant too high.

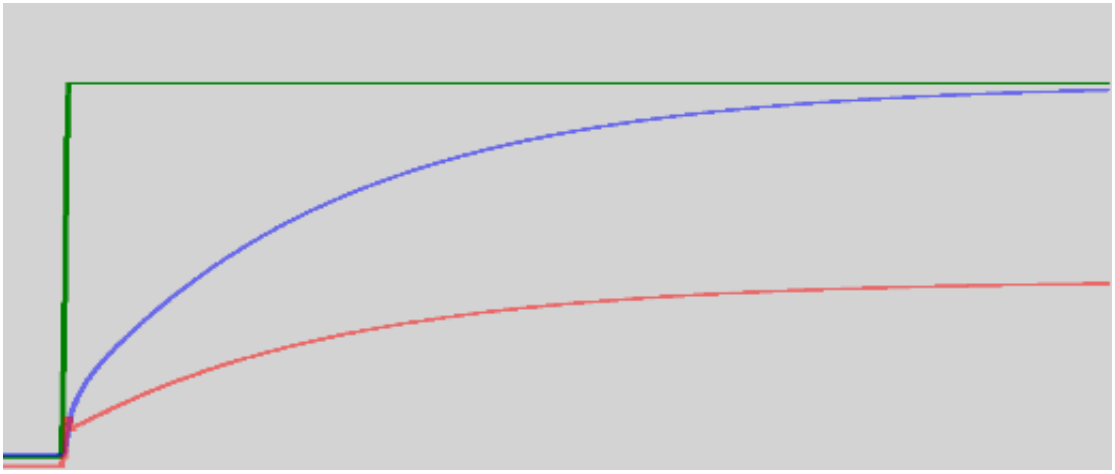

Figure 3: Undershoot. Integral constant too low.

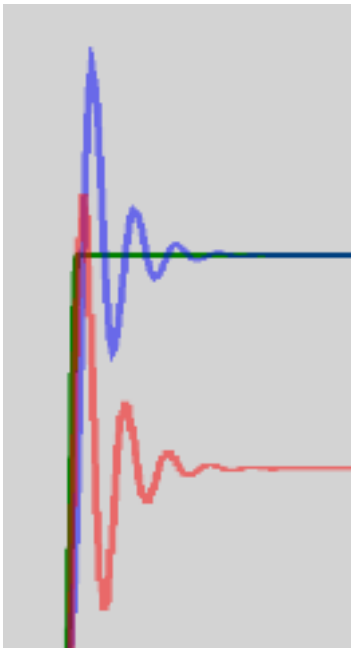

Figure 4: Ringing. Integral and Proportional gain constants too low.

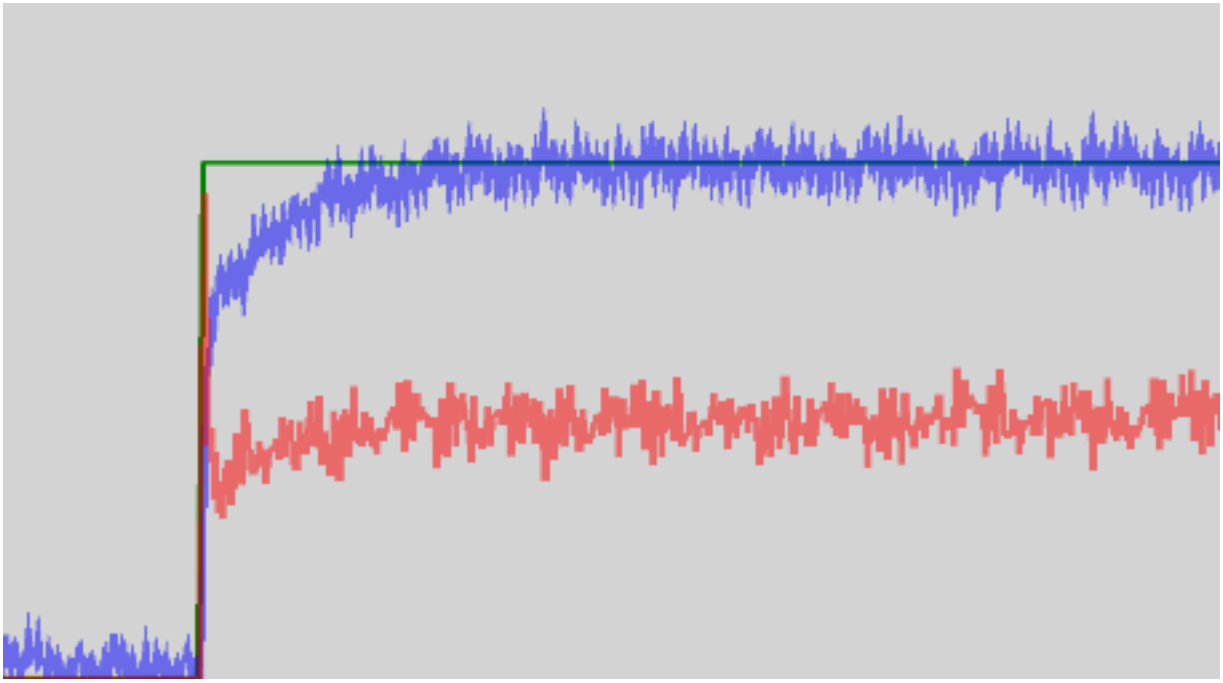

Figure 5: Noise (under control). All three constants are set correctly.

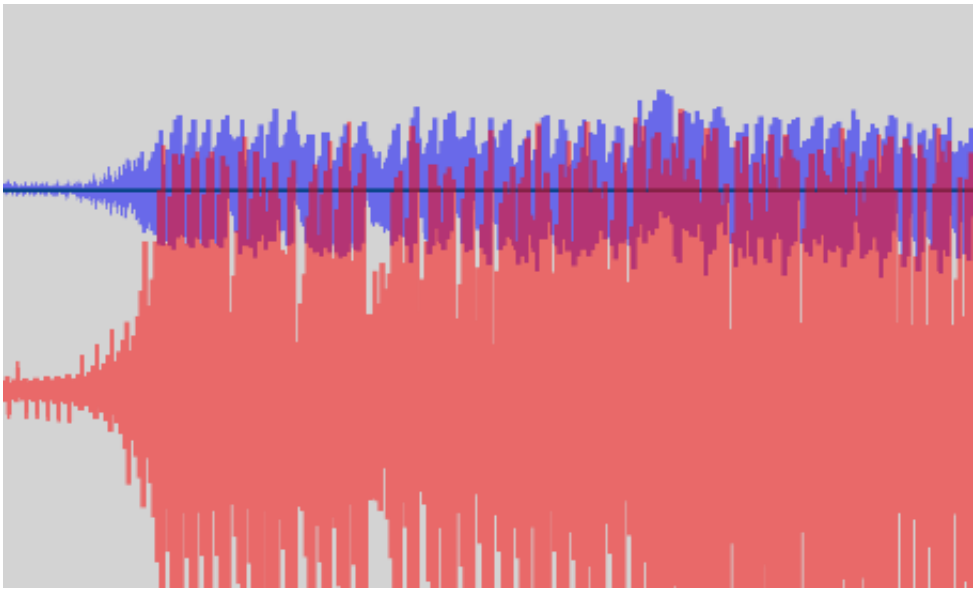

Figure 6: Noise (not controlled). Derivative gain constant too high.

2.5 Tuning a controller

Better PID/P.I loop tuning results in efficient energy use.[2]

3 Experimental Investigation

The purpose of the laboratory session was to explore the methods of tuning a P.I and P.I.D controller, both iterartively, judging the displayed output and making small adjustments, and also predicting the required ratio of P:I:D using the Simulink software.
Employing a closed loop feedback system after analysing the open loop step response to tweak the behaviour of the system using three parameters, the accuracy and precision of the error value difference compared to the set point, and the time the system takes to reach this steady state value, while also minimizing both under and overshoot.

The initial exploration was to plot a graph of a P.I.D output varying just the Propotional value, and setting both Differential and intergral aspects to 0.

P value	outcome
0.2	Undershoot, never reaches setpoint
0.5	Undershoot, never reaches setpoint
1	Only slight Undershoot, never reaches setpoint
1.5	reaches setpoint however lots of variability and does not reach steady state
2	Overshoot

Clearly proportional control alone is not allowing the system to reach it's desired performance level of a fast rising and accurate (minimised error between value and setpoint) step response.

Next a gradual introduction of Integral control was applied with Proportional and derivative values set to 0.

I value	outcome
0.2	Long rise time, never reaches setpoint
0.5	Long rise time, reaches setpoint after too much elapsed time
1	Only slight Undershoot, never reaches setpoint
1.5	reaches setpoint however lots of variability and does not reach steady state
2	Overshoot wildly

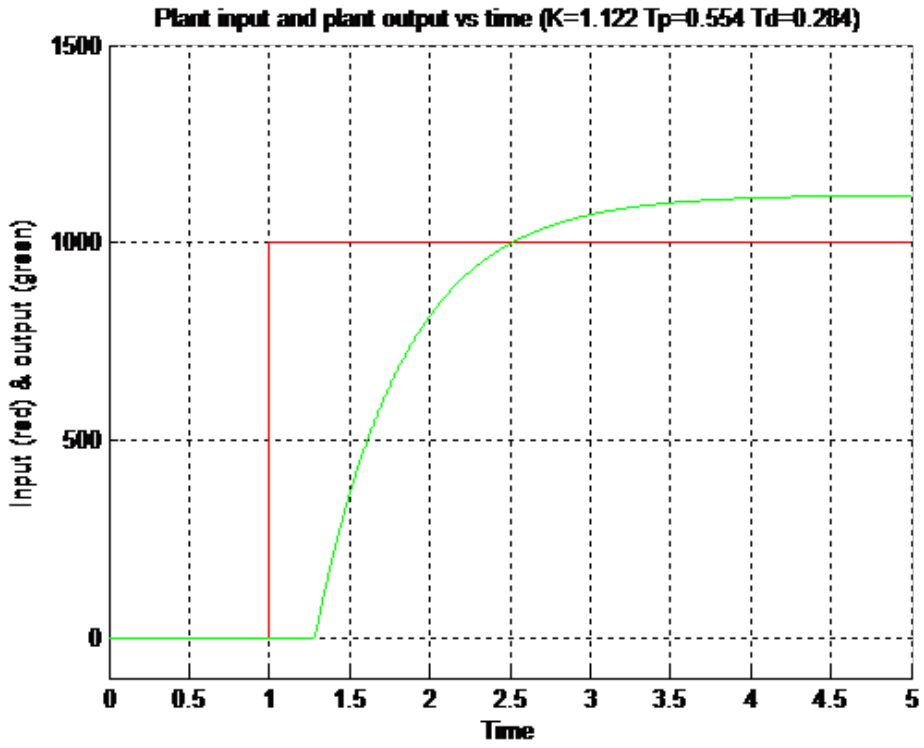

Figure 7: Open Loop Step Response

The equivalent P.I.D values for this step response were determined to be $K = 1.122$, $Tp = 0.554$, $Td = 0.284$ this was determined by a graphical method using a print out of the graph, similar to interpreting an oscilloscope print out, the gradient was calculated as a time constant Tp, time delay Td, and the gain K. The value of 0.63 was used to multiply the gain value and determine a gradient, 0.63 is produced from the fact one equation is

$$K(1 - e^{\frac{-t}{Tp}})$$

K being gain. when t, time is equal to the time constant the equation becomes

$$K(1 - e^{-1})$$

which is $0.63K$

3.1 Method

The initial experiment was to explore the characteristics of the individual parts of the P.I.D controller, graduating through values 0.2,0.5,1,1.5,2 to determine their effects on the test system.
Then the aspects were combined to find a tuned system first using only P.I tuning and then P.I.D tuning to refine the step response.

3.2 Results

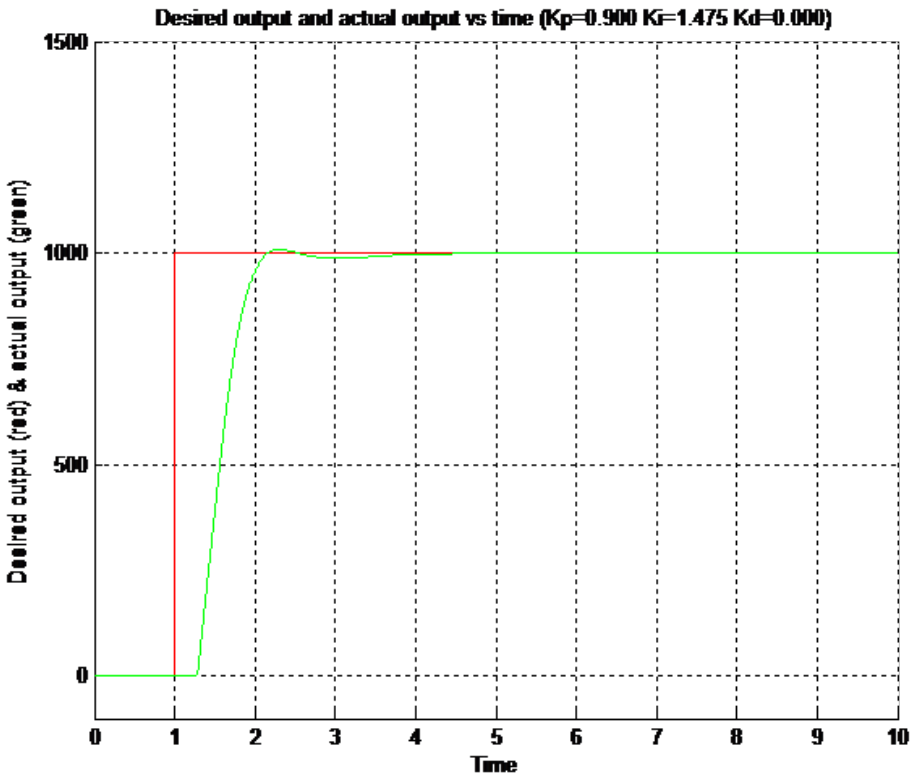

Figure 8: closed Loop Step Response for P.I

Having values $Kp = 0.9$, $Ki = 1.475$, $Kd = 0$

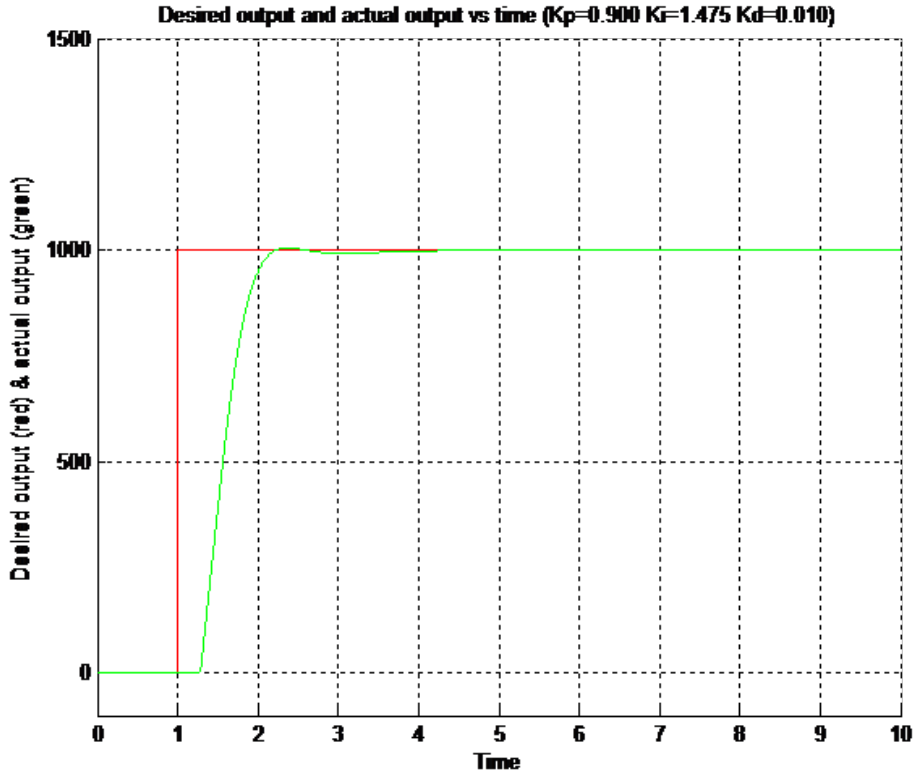

Figure 9: closed Loop Step Response for P.I.D

Having values $Kp = 0.9$, $Ki = 1.475$, $Kd = 0.01$ Here, only a small amount of derivative control was added to slightly tweak the already, quite good step response.

4 Theoretical Investigation

The software package MATLAB/Simulink was used to model and simulate the P.I.D responses for the system.

4.1 Method

The experimental values were input and then tweaked using Simulink

4.2 Results

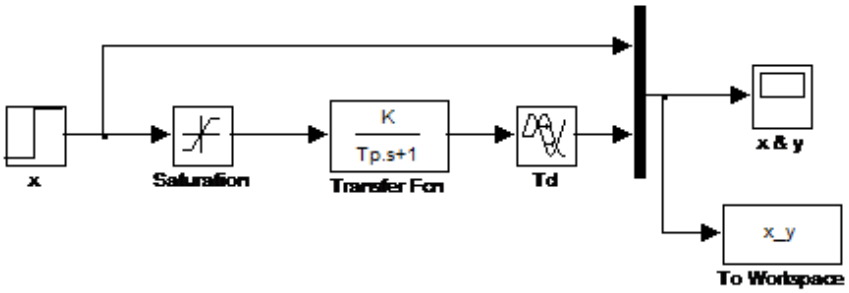

Figure 10: Open Loop system scehmatic

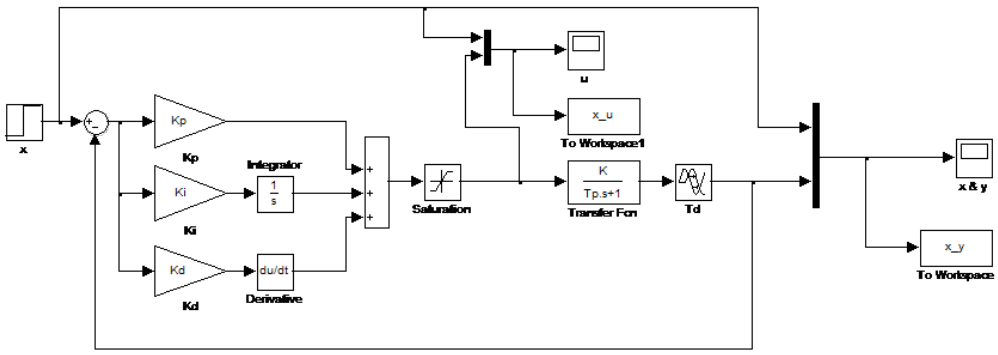

Figure 11: closed Loop schematic for P.I.D

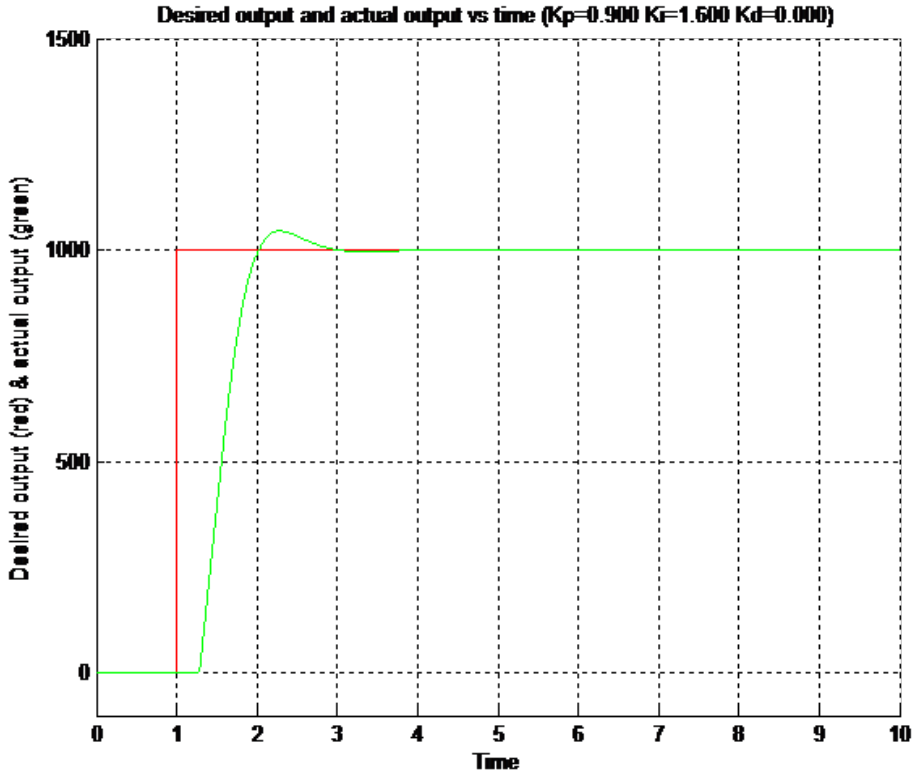

Figure 12: closed Loop Step Response for P.I

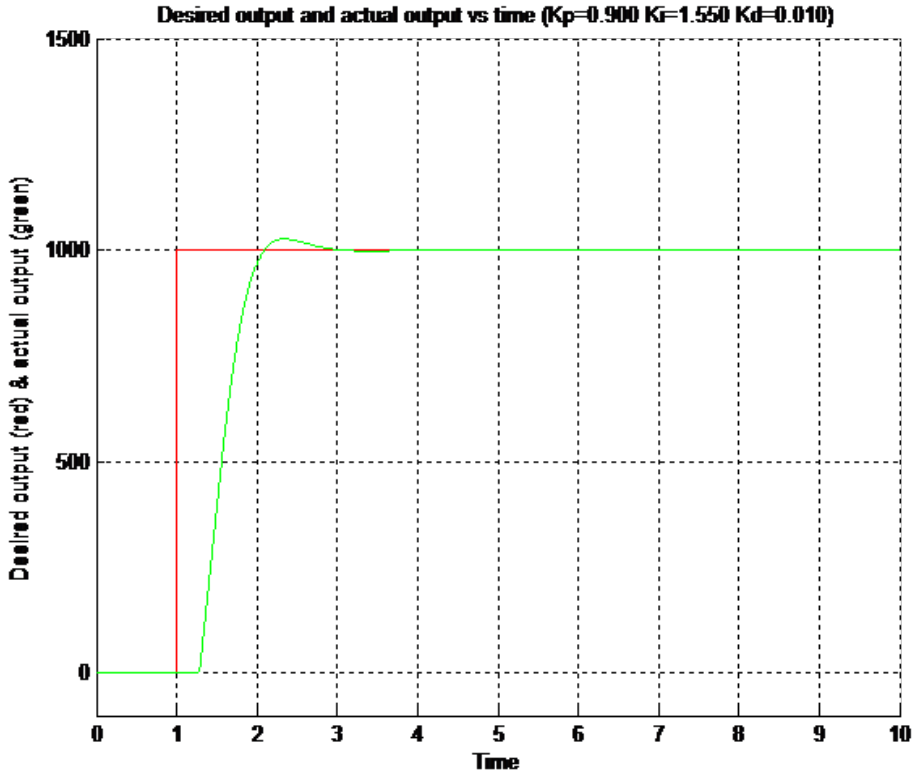

Figure 13: closed Loop Step Response for P.I.D

5 Discussion

The results show that the tuning of the PID is indeed quite similar to the simulation, and also that the results are quite good for this system, if implemented in a real system, these P.I.D tuning values should perform well with this system, and provide the correct amount of feeback to ensure the system never deviates too far from the steady state.

6 Conclusion

In conclusion the lab session has been a introduction to P.I.D control, and the concepts of modeling in matlab/simulink and tuning a P.I.D controller have been conveyed, to this end the results were very good. The characteristics of the variables of a P.I.D controller have been determined.

7 References

[1] Åström, Karl J.; Murray, Richard M. (2008). Feedback Systems: An Introduction for Scientists and Engineers. Princeton: Princeton University Press. p. 293. ISBN 0-691-13576-2.

[2] Icady, (14 May 2009), PID process control a "Cruise Control" Example, Available: http://www.codeproject.com/Articles/36459/PID-process-control-a-Cruise-Control-example, Last accessed 25/03/2014.