Closed Loop Pressure Control in Fixed Volume Container

In this project, the parameters of First Order Plus Dead Time (FOPDT) model of Pressure Control Setup in Fixed Volume Container were identified using bump test procedure. Then various controllers with the help of Integral of Time Weighted Absolute Error (ITAE) and Internal Measurement Control (IMC) techniques were implemented.

The pressure inside of a fixed container is controlled via different type of controllers in this project. A mini air compressor is used as an actuator to the plant, which is a fixed container. Actuator can pump pressure up to 250 PSI, so it requires high power to be driven. Thus, 100 Watt power supply is utilized for this purpose. Actuator and power supply can be seen in Figure 1. Overall block diagram of the closed loop system can be seen in Figure 2.

Figure 1: Actuator (on the left) and power supply (on the right)

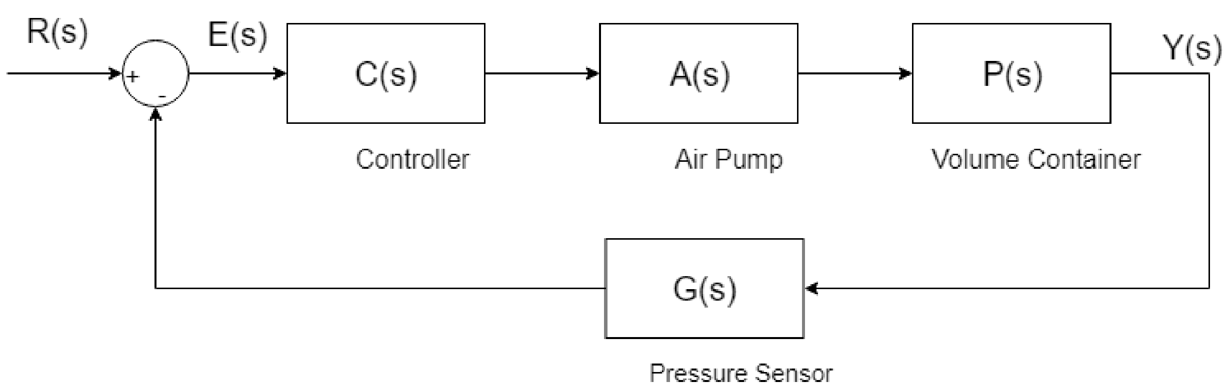

Figure 2: Block diagram of the system

To implement control algorithms, Arduino (nano) is used as a microcontroller in this project. In addition, digital pressure sensor BMP180 is used to measure the pressure changes in the fixed container. Arduino nano microcontroller and BMP180 digital sensor can be seen in Figure 3.

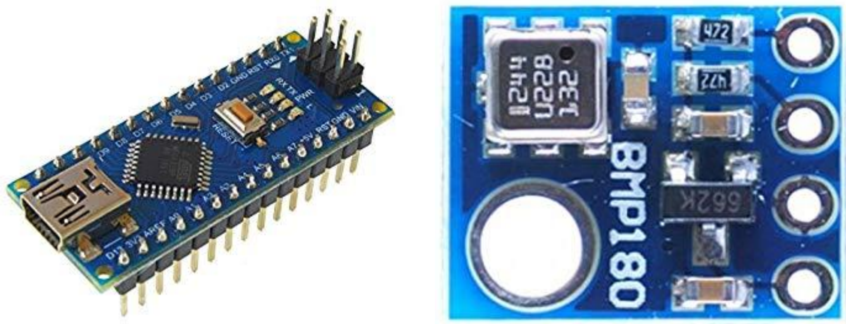

Figure 3: Arduino Nano (on the left) and BMP180 pressure sensor (on the right)

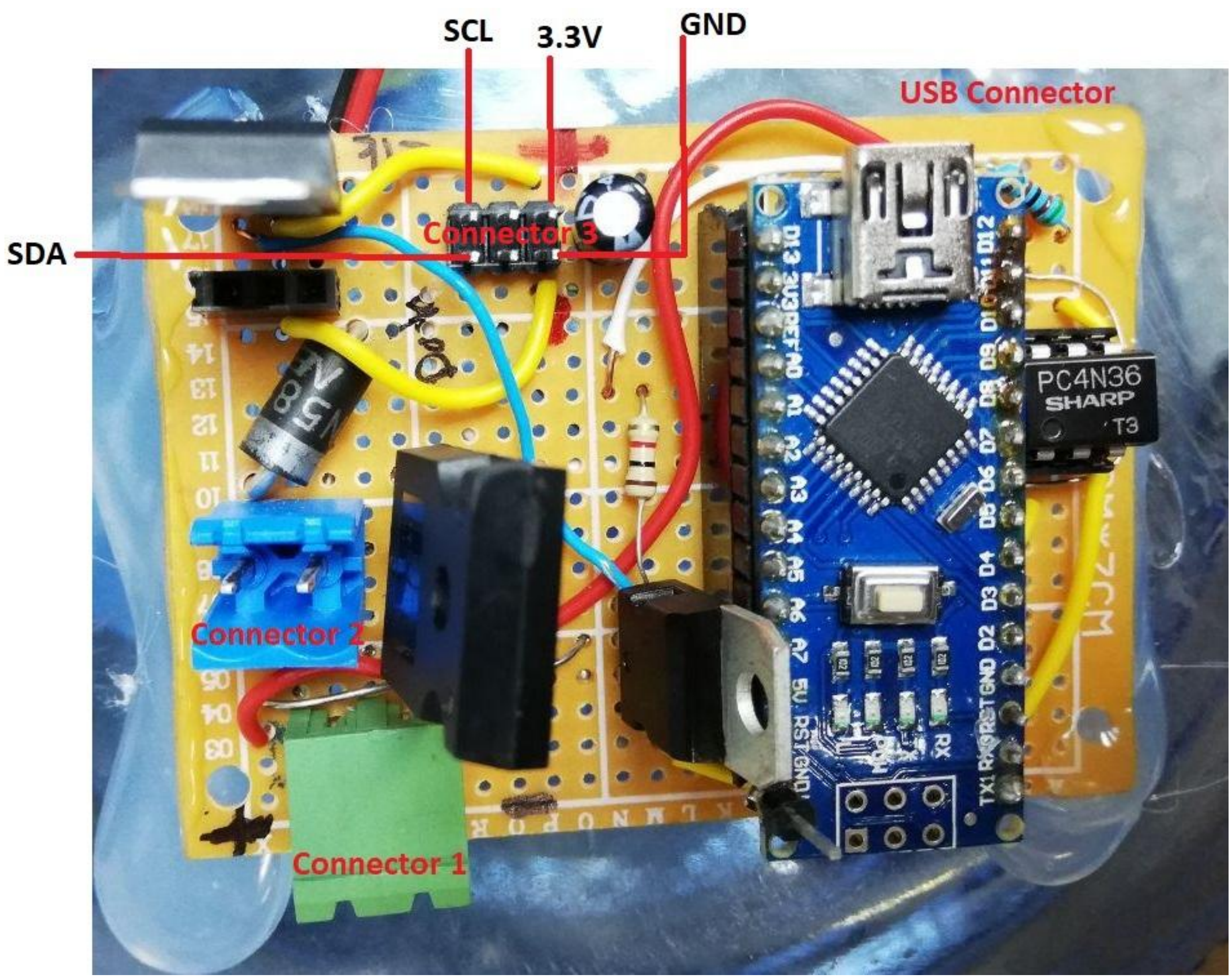

Figure 4: Control board of the setup

To be able to run the setup, one shall make the following connections to the control board seen in the Figure 4.

- Connect output of the power supply to Connector 1
- Connect motor to Connector 2
- Connect sensor to the Connector 3 from its specified pins
- Connect your PC to arduino using mini USB connector

Bump Test Based System Identification

Assuming that the system is self-stabilizing, bump test can be applied properly and response data obtained from wire out to wire in is used for calculating necessary parameters to identify the system completely as can be seen in figure 4.

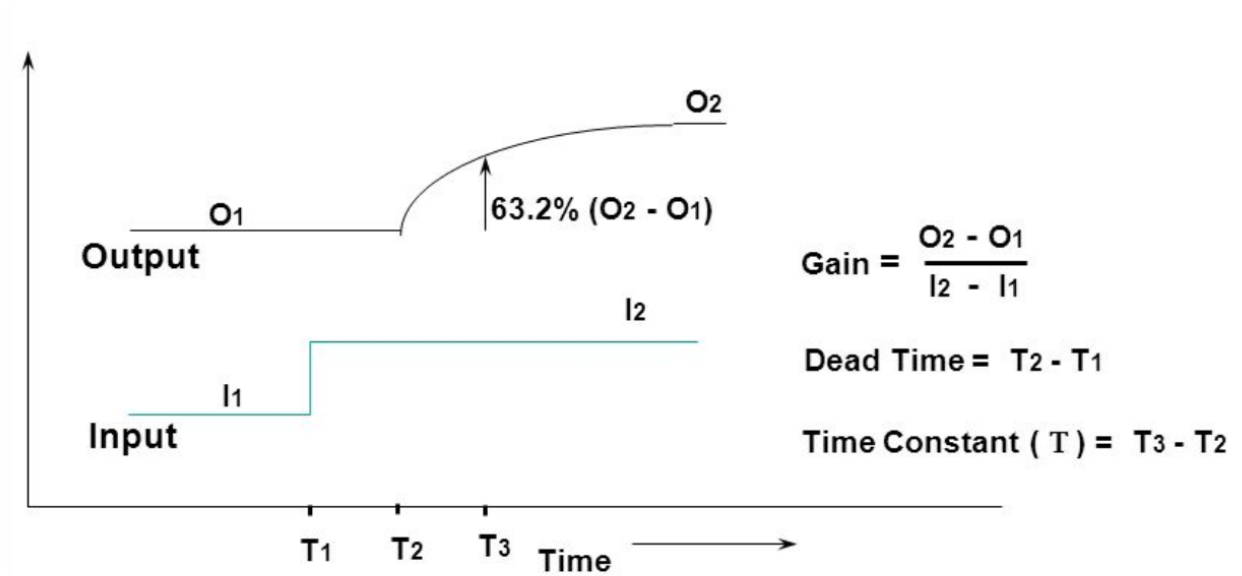

Figure 5: Response of an hypothetical bumped process

T1:Bumped time
T2:Time where the response starts to change
T3:Time it takes to cover %63 of the total change of the response
O1:Steady state operating point
O2: Steady state point after bumped test is applied

Pressure Control

Pressure control is achieved with different types of controllers(on-off, P, PI, PID) in this project.

1 On-Off Control

In this mode of control, the actuator of the system is driven with a binary control signal. For this reason, the output signal generated by the on-off controller gets either HIGH or LOW values. When error signal passes over a predetermined positive value, the controller goes into off mode whereas the signal goes below a pre-determined negative value, operation mode is switched on. Moreover, range between maximum and minimum values that output gets is called as deadband.

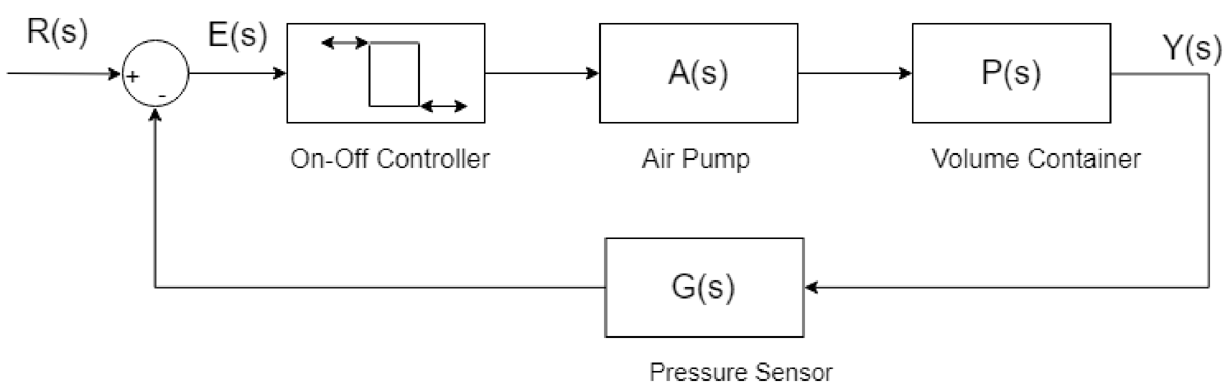

Figure 6: System’s Block Diagram with On-Off controller

2 Proportional-Integral-Derivative Control Family

There are various methods to design and tune PID controller family. In this project, obtained FOPDT parameters were used to design PID controllers. For this purpose, pre-computed values in ITAE and IMC techniques were employed to find PID controllers’ parameters.

Procedure & Results

Bump Test Identification:

1. Open the arduino script code
2. Check the variable MATLAB_MODE. It should be set to 1 to use Arduino with MATLAB.
3. Equal “state” to zero. This will activate bump-test with MATLAB interface.
4. Click tools at the top of the window.
5. Select board as: Arduino Nano.
6. Check cables that connect arduino to computer. They should be plugged in.
7. Again click tools and choose port for Arduino.
8. Now you are ready to upload the code to Arduino.

9. Now open MATLAB script

10. Go to line 8 and change the serial port name as in step 7 (like COM3).

11. Run “OBTAIN DATA” and “Calculation of the Plant Transfer Function” Sections.

12. MATLAB will give an output like in the Figure 6. Save the plot.

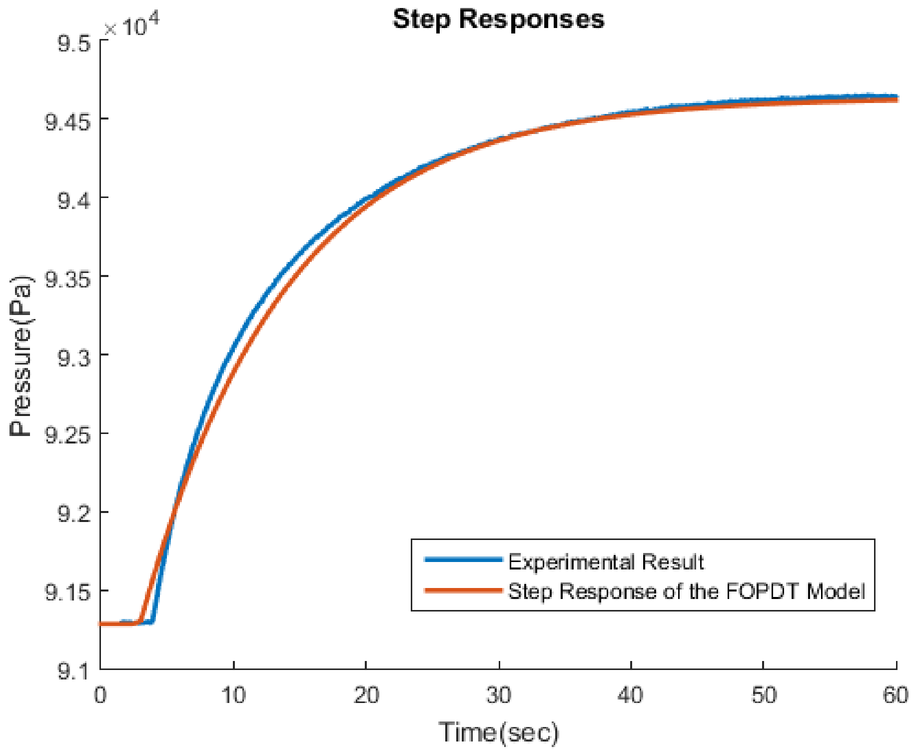

Figure 7: Example Bump Test Response

13. You should be able to see FOPDT parameters Kp, theta_p and tao_p for the system in the MATLAB workspace. Save the parameters.

<u>On-Off Controller:</u>

14. Open the arduino script code

15. Check the variable MATLAB_MODE. It should be set to 1 to use Arduino with MATLAB.

16. Equal “state” to 1. This will implement an on-off controller with a certain dead-band.

17. Go case 1 of switch-case structure. Set tolerance equal to 10. This will specify deadband length of the on -off controller.

18. Repeat steps 4-8.

19. Now open MATLAB script

20. Go to line 8 and change the serial port name as in step 7 (like COM3).

21. Run “OBTAIN DATA” Section.

22. You should be able to see a plot like in Figure 7. Save the plot.

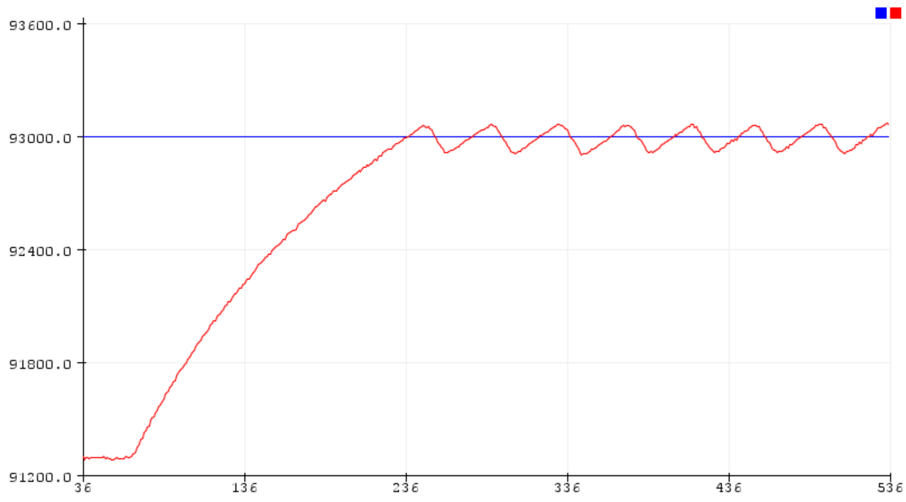

Figure 8: On-off controller response of the system

23. Repeat Steps 14-22 with only one difference. In step 17 change tolerance as 2.

<u>P Controller:</u>

24. Repeat steps 1-8, but set state as 2 not zero.
25. Repeat steps 9 and 10.
26. Run ”OBTAIN DATA” and “P-Controller with ITAE calculation” section. It creates an aggressive P controller.
27. You should be able to see a MATLAB plot like in the Figure 8. Save the plot.

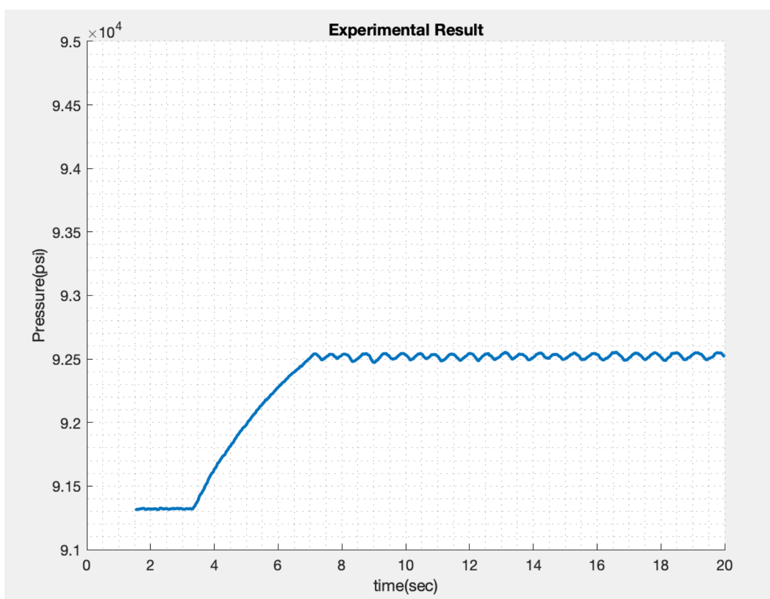

Figure 9: P controller response

<u>PI Controller:</u>

28. Repeat steps 1-8, but set state as 3 not zero.
29. Repeat steps 9 and 10.
30. Run ”OBTAIN DATA” and “PI-Controller with IMC Rules” section. It creates an aggressive PI controller.

31. You should be able to see a MATLAB plot like in the Figure 9. Save the plot.

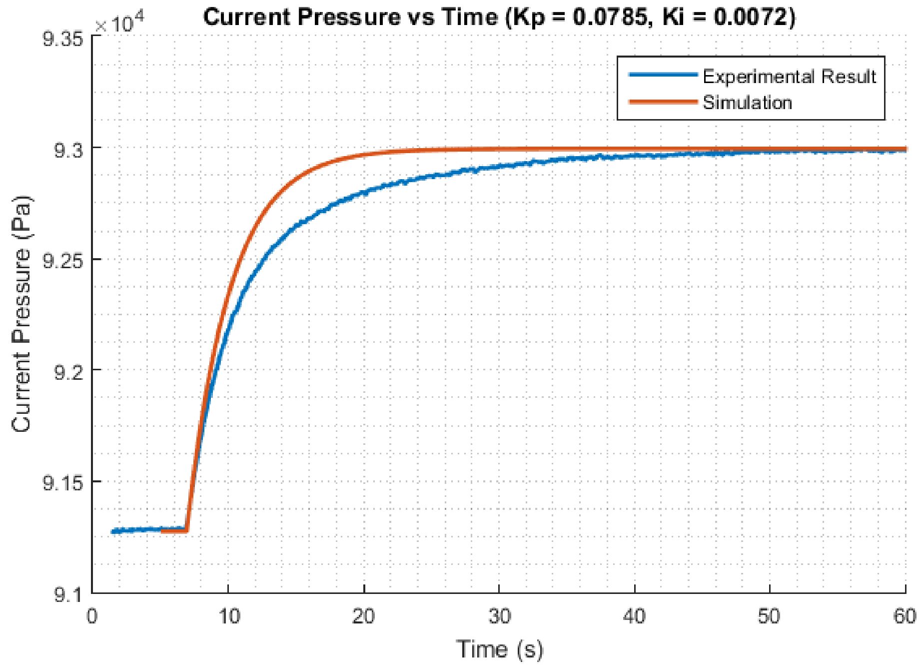

Figure 10: PI Controller Response

<u>PID Controller:</u>

32. Repeat steps 1-8, but set state as 4 not zero.
33. Repeat steps 9 and 10.
34. Run ”OBTAIN DATA” and “PID Controller” section. It creates an aggressive PI controller.
35. You should be able to see a MATLAB plot like in the Figure 10. Save the plot.

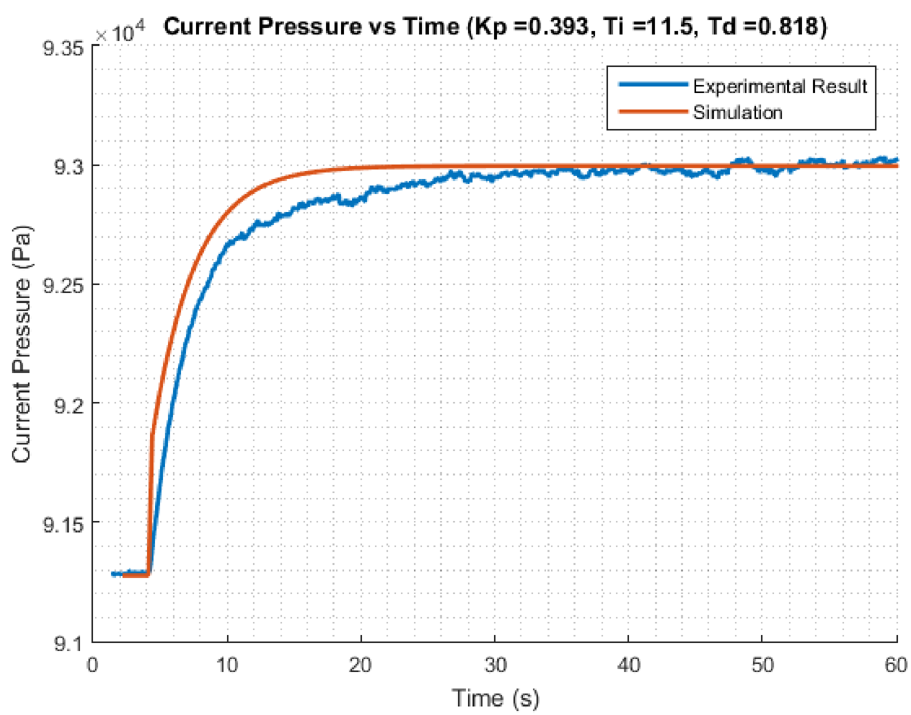

Figure 11 : PID Controller Response