FUNDAMENTAL LEVEL MEASUREMENT AND CONTROL CONCEPTS DEMONSTRATED USING MICROPROCESSOR ACTIVITIES

M. A. K. Rasel, Yan Fang, Tracy J. Benson, and Peyton C. Richmond
Dan F. Smith Department of Chemical Engineering
Lamar University

Abstract

Basic concepts of process control are intuitive to understand but sometimes difficult to demonstrate. This paper describes a level control experiment used to provide hands-on control experience for both high school outreach students and undergraduate process control students taking Lamar University's Process Control Laboratory course CHEN4150. Students measure a tank's level, filter those measurements, calibrate the level, add control actions, and develop control algorithms using a microcontroller. In this paper, we provide the details of the experiment including Arduino microprocessor source code and the student tasks. Student performance results and a feedback survey show that this exercise improved their understanding of measurement and control concepts.

Introduction

To help students understand basic concepts of process control we designed a hands-on tabletop level control experiment using an Arduino [1] microcontroller. The role of process control in the process industry has expanded as technology has advanced. As Romagnoli and Palazoglu [2] noted, process control is replacing many manual manufacturing activities in industrial operations with full automation while continuing to maintain process variability near the desired values which contributes to safety, minimizes environmental impacts, and optimizes processes. Seborg et al. [3] mentioned that process control has become increasingly important in the process industries because of global competition, rapidly changing economic conditions, and more stringent environmental and safety regulations.

We wanted to provide an incentive for students to push for a deeper understanding of concepts taught in process control so they could master the material. Students' lack of both ability and interest has been cited as reasons for the poor academic performance of today's youth [4]. This lack of incentive may be a contributing factor to the poor retention of diverse undergraduate students in many engineering programs. We designed these modules with activities that are perceived as important in order to motivate students and develop their interest in hopes of achieving an intrinsic motivational orientation. The mastery goals set based on a well-developed interest are more desirable for engineering students because they lead to more persistence in the face of difficulty or failure [5].

Our pedagogical approach is similar to other microcontroller based educational activities [6, 7]. Students have full access to all of the hardware and software used for this experiment and are encouraged to experiment with the system. The teaching modules described here introduce process measurement, filtering, calibration, and control all implemented with an Arduino microprocessor using accessible and community supported hardware and software. The process measurement module has students connect a pressure sensor to the process and read voltages with the Arduino. Subsequent modules cover filtering the readings to reduce measurement noise, applying calibration to convert the voltages into a percentage level of water in the acrylic container, and varying the speed of two small submersible pumps to control the water level using a motor control shield. We assessed the module and student learning outcomes for the experiment with a student feedback survey after the experiment was performed.

This experiment is currently used as part of our CHEN4150 Process Control Laboratory course, a senior co-requisite course with our CHEN4332 Process Control II course. In the undergraduate process control laboratory, the lessons culminate with the traditional Proportional-Integral-Derivative (PID) control experience. To adapt this experiment for high school outreach activities we replace the PID control activity with a simple relay control algorithm. Since each experiment only cost slightly more than $200, we constructed eleven copies of the experiment, one for each desktop in our Process Control Laboratory. Thus, by forming 3-4 person groups, we were able to reach 35 high school students simultaneously.

Level Control Experiment

The Arduino-based level control experiment includes a small acrylic tank and two submersible pumps. One pump is placed into the reservoir and fills the tank while the other pump is used to empty the tank (see Figure 1). The reservoir pump is programmed through the Arduino to deliver water at a constant rate into the tank while the tank pump rate is varied to control the tank water level. The pumps' rates are controlled by Pulse Width Modulation (PWM), meaning that the voltage spends a certain percentage of the time high and a certain percentage of the time low. Since the Arduino is powered by a low amperage 5 Volt source and the pumps operate at 12 Volts and higher amperages, external power is required. A motor shield mounted on top of the Arduino board converts the 0-5 Volt Arduino PWM signal to a 0-12 Volt PWM signal from an external power supply.

An aquarium air pump is used to blow air through a straw leading to the bottom of the tank balancing the tank's liquid head pressure. A pressure sensor connected to the air tube that blows air into the straw converts the air pressure to a voltage signal between 0-5 Volts. (Blowing air through the straw also prevents water from contacting and destroying the pressure sensor.)

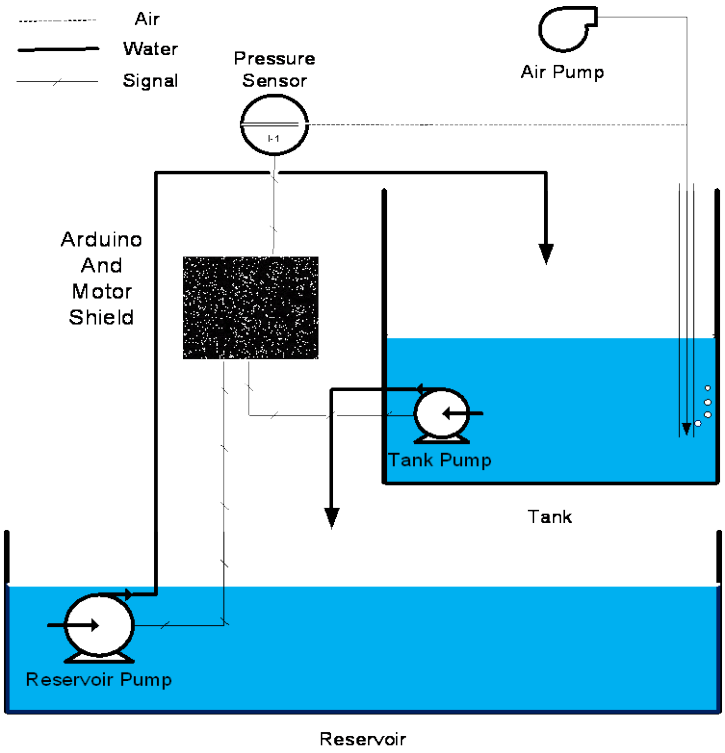

Figure 1: Level Measurement and Control experiment schematic diagram.

The air pump's rate can be adjusted manually to control the rate of air bubbles. Thus, the pressure sensor indirectly measures the liquid head in the tank and the bubbling rate determines the amount of measurement noise seen by the students.

The Arduino module is used to implement liquid level control by reading the pressure sensor voltage and converting it into a liquid level and then calculating an output based on a control algorithm to drive the pump. The Arduino is connected to a computer through a Universal Serial Bus (USB) interface to upload and run the code. The computer also works as a serial monitor to print the value of variables in the running code.

The entire tank and reservoir assembly is kept inside a bus box so that the electronics which are placed outside the bus box remain dry as shown in Figure 2. The complete list of items used in the Level Measurement and Control experiment are listed in Table 1.

Activity Modules

This section describes the Level Measurement and Control experiment equipment and theoretical concepts provided to the students. The students are required to draw a P&ID

Table 1: Cost of items for setting up Level Control experiment.

LEVEL CONTROL EXPERIMENT	
Undivided Bus Box × 1	$17.56
Storage box (6 quarts) × 1	$5.50
Storage containable(4"*4"*7") × 1	$5.99
Storage containable (3.5"*3.5" * 5") × 1	$6.35
Oggi acrylic straws× 1	$4.76
Arduino uno r3× 1	$22.01
Box Arduino× 1	$13.01
Fusion air pump× 1	$14.15
High temp mini DC Water Pump 12V × 2	$12.95× 2
Power supply regulated× 1	$19.40
Breadboard × 1	$5.50
Sensor pressure SMD 8-sop × 1	$13.94
4 way nickle plated valve× 1	$6.46
Silicone tubing × 2	$3.55× 2
Air pump check valve × 1	$5.13
Motor shield for Arduino× 1	$23.99
Tubing connector filters× 1	$4.35
Permeatex silicon adhesive× 1	$4.84
Krazy glue× 1	$3.99
TOTAL	$209.93

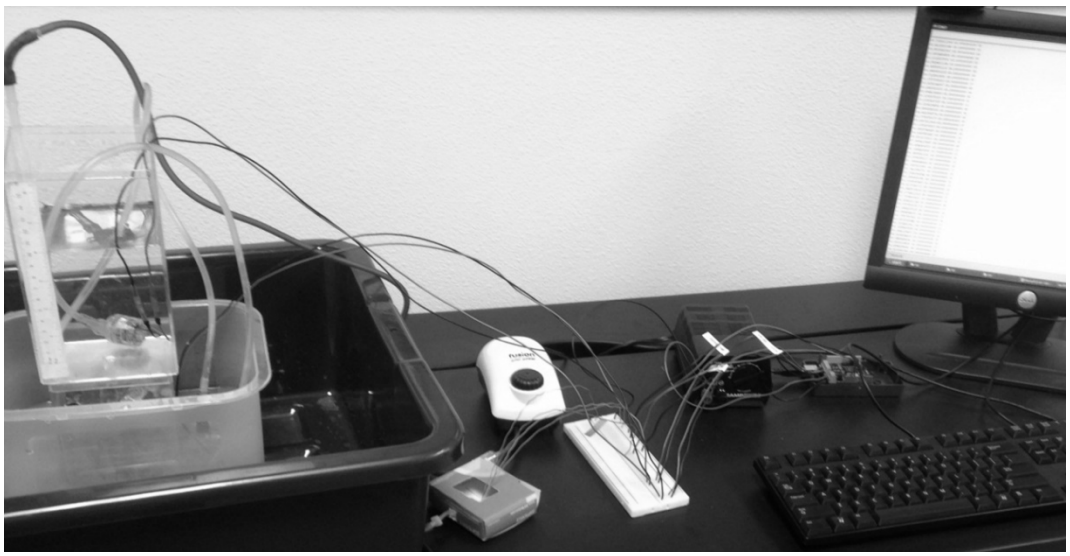

Figure 2: Level Measurement and Control experiment equipment.

Level Control Experiment: (1) pump control, (2) data acquisition and filtering, (3) calibration and measurements, and (4) PID level control. The Arduino program associated with each task is accessible via the corresponding author. After performing the lab, the students report their work procedure and results on each task.

Task 1

In this task students learn how to control the pumps using the Arduino and its motor shield. The two water pumps require DC current at twelve volts. However, we can vary the pump speed using a PWM signal. The PWM signal alternates voltage at a high frequency between 12 volts and 0 volts. The Arduino ***analogWrite()*** command is used to set the PWM duty cycle with a range from 0-255 that corresponds to 0% to 100%. For example, a PWM value of 255 provides a continuous 12V voltage to the pump while a PWM value of 127.5 provides a voltage of 12V 50% of the time and 0V otherwise.

Students first calculate the PWM signal to run the pump at a 75% duty cycle. To operate the pumps successfully students must locate the appropriate section of the Arduino code and enter the correct output value for the PWM signal required. The code that should be changed is set off by two comment lines full of asterisk marks as shown in Figure 3.

After students change the code that sends the output signal to the reservoir pump to 75% while setting the tank pump output signal to 0%, students upload the Arduino program and place the reservoir pump tube into the reservoir and place the tank pump tube into the tank. Then they turn on the 12V power supply and verify that the pumps operate as expected. As an additional step they measure and explain the voltage across each pump. After following this procedure, students are able to change the tank level by manipulating the pump signals from the Arduino and pumping water to and from the reservoir and tank.

Task 2

Students measure the pressure sensor reading using the Arduino board during this task. A pressure sensor is used to measure the tank level indirectly. The Arduino supplies voltage to the pressure sensor integrated circuit and reads the return voltage. The command to read the sensor's voltage is the ***analogRead()*** function, which returns a value between 0 – 1023 proportional to the voltage seen at the pin between 0V and 5V. The pressure sensor signal is noisy due to the bubbles resulting from blowing air through the straw. A low pass filter is used to reduce noise in the sensor readings. The filter is designed using the exponential moving average smoothing using the following equation.

$$y_t = \alpha * x_t + (1 - \alpha) * y_{t-1} \qquad (1)$$

In the above equation, x_t are the raw measurements and y_t are the filtered values at time t. The smoothing factor α has the value between 0 and 1.

Students find the Arduino pin connected to the pressure sensor and verify that this pin was referenced in the code. Then they find the variable that stores the sensor reading. Students can adjust the smoothing factor alpha in the

```
//*********************************************
pumpT = 0;  // Set Tank pump output; range 0-255
pumpR = 191.25;  // Set Reservoir pumpp output; range 0-255
//*********************************************
```

Figure 3: Partial Arduino Code of Pump.ino.

code. The initial smoothing factor (alpha) and the supplied filtering code are shown in Figure 4.

Students upload the program to the Arduino which begins collecting pressure sensor data. When the filtered output of the program stabilizes, students record thirty sets of the sensor data and the filtered output and calculate the average and the standard deviations of unfiltered and filtered data to compare their results. The advantage of noise reduction is observed by calculating the standard deviation of the sensor and filtered data. Table 2 below shows a sample result reported by students. Students are encouraged to experiment with changing the air bubbling rate during this experiment.

Task 3

In task 3, the students convert the pressure sensor signal from a voltage measured by the Arduino into a percent full reading for the tank level. The filtered pressure sensor signal is calibrated to level percent using a scale attached to the water tank. A linear calibration equation converts the sensor signal to percent tank level.

$$z_t = 100.0 * (y_t - y_1)/(y_6 - y_1) \qquad (2)$$

In the above equation, z_t is the percentage tank level; y_1 and y_6 are the filtered value at height 1 inch and 6 inch mark, respectively. The students calibrate the tank level to read 0% at 1 inch of water and 100% at 6 inches of water. First, they fill the tank to the 6 inch mark by operating the water pumps like in task 1 and record steady data of the filtered value from the serial monitor. Then they empty the tank to the 1 inch mark and record the filtered data. Students then set the variables of *head1* and *head6* to the filtered values obtained at the 1 inch and 6 inch marks, respectively. Students must write the code that converts the pressure sensor value to a tank percentage variable as shown in Figure 5.

```
float alpha = 0.6;                // alpha is the smoothing factor ranges 0 < alpha < 1
float filterVal = 1.15;              // the filtered value

//This is the setup function
void setup()
{
  ...
}

void loop() {
  // read the value from the sensor:
  analogVal = analogRead(sensorPin); // 10-bit digital number 0 = 0V to 1024 = 5 V

  // calculate the filtered value using simple low pass filter
  filterVal = alpha*analogVal + (1.0 - alpha)*filterVal;
  ...
}
```

Figure 4: Partial Arduino Code of Filter.ino.

Table 2: Analog and filtered variable data at one inch water height.

	Analog Reading	*Filtered Reading*
Average	290.60	290.67
Standard Deviation	10.15	6.43

```
// calculate the filtered value using simple low pass filter
filterVal = alpha*analogVal + (1.0 - alpha)*filterVal;

//*****************************************
head1 = 290;
head6 = 528;
headPCT = 100.0*(filterVal-head1)/(head6-head1);  // Students should write code to convert head to percentage
//*****************************************
```

Figure 5: Partial Arduino Code of Calibration.ino.

After performing this task, students change the tank level to the 2 inch mark and verify that the tank percent level calculated by the Arduino and displayed by the serial monitor is 20%.

Task 4

In task 4, students combine a prebuilt Arduino discrete PID control code with the pump control and pressure sensor codes they used in tasks 1-3. In the PID algorithm, the controller action MV is calculated from the error term e(t) according to equation 3.

$$MV = K_P\left(e(t) + \frac{1}{T_i}\int e(t)dt \; + T_d\frac{de(t)}{dt}\right) \quad (3)$$

Here,
e(t) = Set point (SP) – Process Variable (PV)
K_P = Proportional gain
T_i = Integral time constant
T_d = Derivative time constant

A discrete approximation of the above equation is implemented in the Arduino as follows:

$$MV = K_p\left(e(t) - e(t-1) + \frac{t_s e(t)}{T_i} - T_d\frac{[e(t)-2e(t-1)+e(t-2)]}{t_s}\right) \quad (4)$$

Here, t_s is the discrete step time.

In this task, students maintain the water level at the 4 inch mark. In the code, as shown in Figure 6, they set the set point (SP) value to 60 corresponding to the 4 inch mark.

Students then replace the value of the variables *head6* and *head1* with the calculated values obtained in task 3. After uploading the program, students record the variables set point (SP), process variable (PV) and control variable (CV) from the serial monitor to evaluate the performance of the PID controller. Students may tune the PID controller by adjusting the gain and re-run the level controller for comparison. A sample response of set point change from a student group is shown below in Figure 7.

For high school students, PID control is replaced with on/off control. Students use *if else* loop to maintain the tank percentage level between 20% and 80% by turning the pump on and off, respectively.

Student Feedback

A student feedback survey is conducted to assess the understanding of control loop using this experiment. The survey is prepared in the form of comments about the Level Measurement and Control experiment. The survey uses a Likert Scale with the labels: (1) Strongly Disagree; (2) Disagree; (3) Agree and (4) Strongly agree. The 5 survey questions are listed below. The result of the 28 student's responses to the Level Control survey questions is shown in Table 3.

```
//*****************************************
float SPP = 60; //Students should set the set point here in percentage of height
//*****************************************
```

Figure 6: Partial Arduino Code of PID.ino Showing Setpint change.

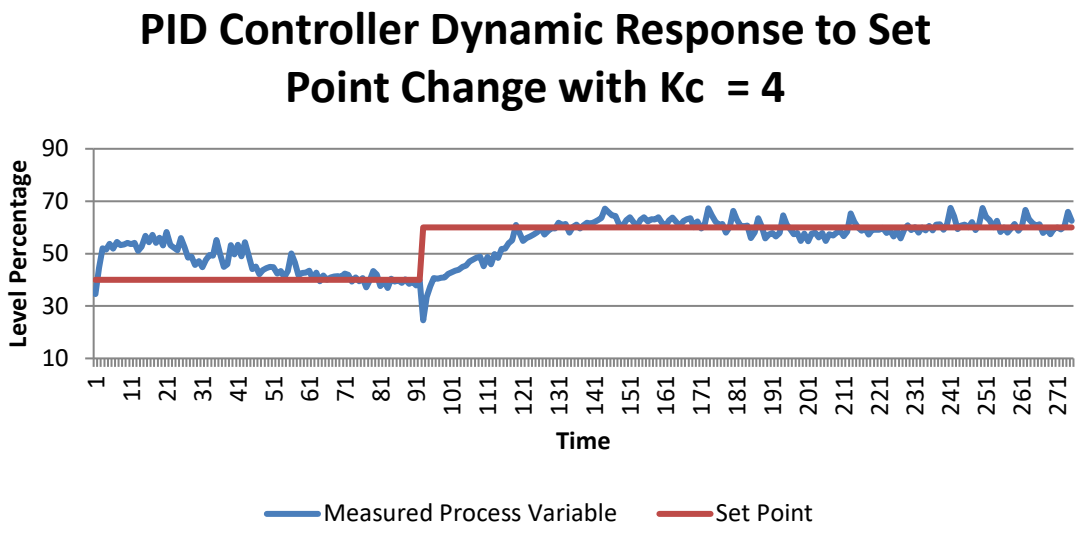

Figure 7: Sample PID controller response from a student group.

Table 3: Students response from survey of Level Control.

Statements	Strongly Disagree	Disagree	Agree	Strongly Agree
1. The Arduino Level Control experiment was easy to understand and perform.	4%	0%	32%	64%
2. The Arduino Level Control experiment will help me to become a more competent engineer in designing control strategies.	0%	14%	61%	25%
3. The goal set for my group was achievable with given amount of experimental time with the Arduino Level Control experiment.	0%	0%	36%	64%
4. The large-scale applicability of the Arduino Level Control experiment is readily identifiable.	0%	14%	36%	50%
5. The Arduino Level Control experiment improved my understanding of how a liquid level is calibrated in a chemical plant setting.	0%	14%	57%	29%

The students' feedback survey reflects the general experience of the students with the Level Measurement and Control experiment. The survey shows that 86 percent of the students find the experiment helpful for them to understand PID loop controllers in chemical plant setting. They also identify the experiment is easy to understand and perform within the class time although it requires understanding of the hardware and codes used for the microcontroller which they are not familiar with before the experiment. Some students mention in their comments that these tasks could be improved by giving more details in the laboratory description. To address those comments, we modified future lab classes to required students to hand draw a schematic of the experiment before performing the lab.

Conclusion

In this article, we described a level measurement and control experiment suitable for both undergraduate chemical engineering

process control laboratory and high school outreach activities. This experiment provides students with hands-on experience with level measurement and control using an Arduino microcontroller. The cost-effectiveness of this experiment allows us to conduct the lab for 35 school outreach program students simultaneously by using 10 desktop experiments.

The feedback from both sets of students was very encouraging. The undergraduate student feedback survey demonstrated excellent acceptance of the experiment and students reported that they improved their understanding of important process control concepts. In addition, this experiment seemed to really peak the high school student's interest and more than one student mentioned how programming with an actual cyber-physical system greatly increased their enthusiasm for programming and really seemed to enhance the motivation of some of the high school students.

References

1. Anon, "Arduino – Home" http://www.arduino.cc/, (accessed July 2012)

2. J. A. Romagnoli, A. Palazoglu, *Introduction to process control*, CRC press, 2012.

3. D. E. Seborg, D. A. Mellichamp, T. F. Edgar, F. J. Doyle III, *Process dynamics and control*, John Wiley & Sons, 2010.

4. S. Hidi, J. M. Harackiewicz, "Motivating the academically unmotivated: A critical issue for the 21st century," *Review of educational research*, vol. *70*. No. 2, Jun. 2000.

5. C. Ames, "Classrooms: Goals, structures, and student motivation," *Journal of educational psychology*, vol. *84*, no. 3, Sep. 1992.

6. P. Teikari, R. P. Najjar, H. Malkki, K. Knoblauch, D. Dumortier, C. Gronfier, and H. M. Cooper, "An inexpensive Arduino-based LED stimulator system for vision research," *Journal of neuroscience methods*, vol. *211*, no. 2, Nov. 2012.

7. B. B. Elmore, "CURRICULUM-A Freshman Design Course Using Lego NXT® Robotics," *Chemical Engineering Education*, vol. *45*, no. 2, 2011.

Biographical Information

M. A. K. Rasel received his B.Sc. in chemical engineering from the Bangladesh University of Engineering & Technology in 2005. He received his M.S. degree in chemical engineering from Lamar University in 2010. He is currently pursuing a Ph.D. in chemical engineering at Lamar University. His research interests include dynamic simulation, instrumentation, and control for process safety and reliability.

Yan Fang is a current Ph.D. student and research assistant of chemical engineering at Lamar University in Beaumont, TX. She received her B.S. (2010) in chemistry from Hebei Normal University in China and her M.E. (2013) in chemical engineering from Lamar University. Her research interests include dynamic simulation and process control. She is currently a teaching assistant of undergraduate process control laboratory.

Tracy J. Benson is an assistant professor of chemical engineering at Lamar University in Beaumont, TX. He received his B.S. and Ph.D. from Mississippi State University. His research interests include CO2 conversion strategies and biomass conversion. He currently teaches the momentum and mass transfer classes and the unit operations laboratory.

Peyton C. Richmond is an associate professor of chemical engineering at Lamar University in Beaumont, TX. He received his B.S. from Lamar University and his Ph.D. from Texas A&M University, both in chemical engineering. His research interests include industrial process control and undergraduate education. He currently teaches the capstone plant design classes and undergraduate process control laboratory.