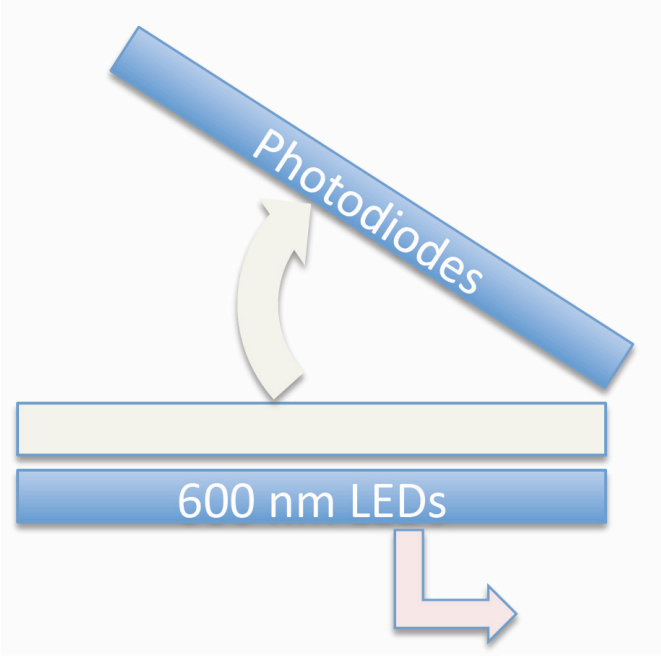
Turbidostat Control Design
MS3



By:
Evan Dreveskracht
Peter Harker
Maxwell Holloway

INTRODUCTION

For this milestone we are mainly showing our controller design and giving an update of our progress. In order to fully implement our mini plate scanner turbidostat, we are incorporating a PI controller into our system. This will be done by writing code with PI controller in C++ taking in plate scanner readings from arduino. Evolution of bacteria can be modeled by two first order differential equations. The first is with respect to population growth, and the second is with respect to nutrients being added to the solution. This is a non-linear system.

Using the two differential equations for population growth of bacteria and nutrients added to our solution, we come up with the open and closed loop simulation from a Simulink model of our system. This model includes the arduino and the liquid handler computer. We are then able to determine the response to Band-Limited white noise and saturation in Simulink. The noise in our system could come from non-uniform bacteria, excess light, light reflection, or any number of unknown variables.

Our original weekly schedule is progressing along fast enough; however the testing of the Optical Density readings from our plate scanner in week 6 is bleeding into week 7. During week 7 however, we will be sending our circuit board and plastic housing off to be fabricated. During this time we hope to catch up with scanner calibrations from week 6.

CONTROLLER DESIGN

NOTE: Some values have been changed from the presentation, because we found a value for the integrator gain that seemed to work somewhat better by reducing the percent overshoot.

PREVIOUS SIMULATION

As shown in figures 1 and 2, we have the simulations from Milestone 2 that show the uncontrolled system does not behave favorably for someone who wants a controlled population. Figure 2 shows the population graphs for the uncontrolled system with u equal to 0, 0.5, and 2, respectively. It can be seen from these graphs that the population does not reach

a steady state near these values (except for 0, which is easy to accomplish by just not adding any fresh media). In order to accomplish a system that will approach a steady state of a given input, we will need to create a controller with feedback. The following sections show the design process and the system behavior with the designed controller.

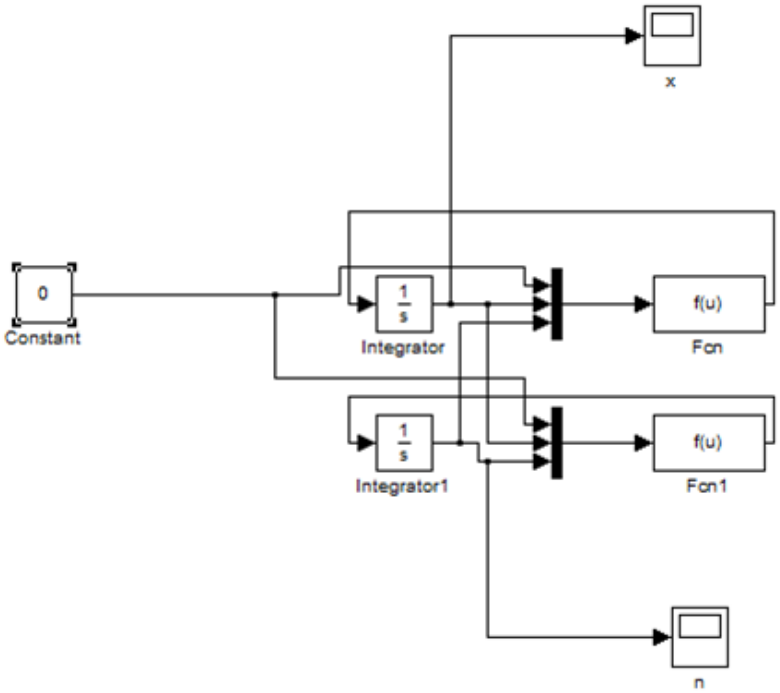


Figure 1 - Open Loop Model of System

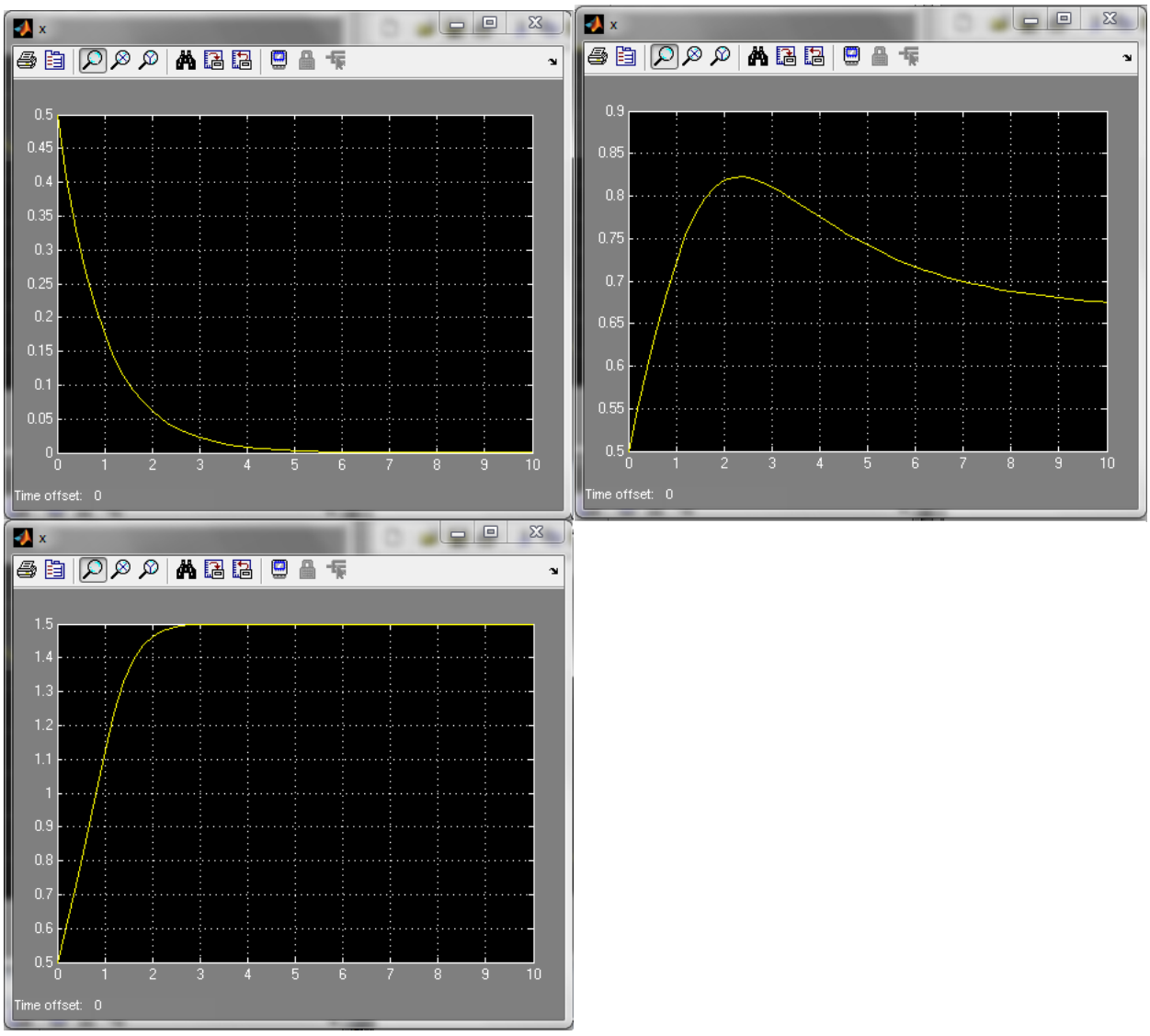


Figure 2 - Population Graphs of Uncontrolled System

When thinking of what kind of controller to implement, we first considered designing a PID controller and tweaking this design to achieve a system that had a quick settling time, a low percent overshoot value, a fairly low rise time, and low steady state error. We quickly realized that it would not be desirable to add a derivative term to our controller due to the effect of noise on the system. Though we probably could have gotten around this issue, we decided that having a large percent overshoot was not enough of a problem that we wanted to deal with the hassle of implementing a derivative term in our controller. For this reason, we decided to use a PI controller to control our system (as shown in Figure 3).

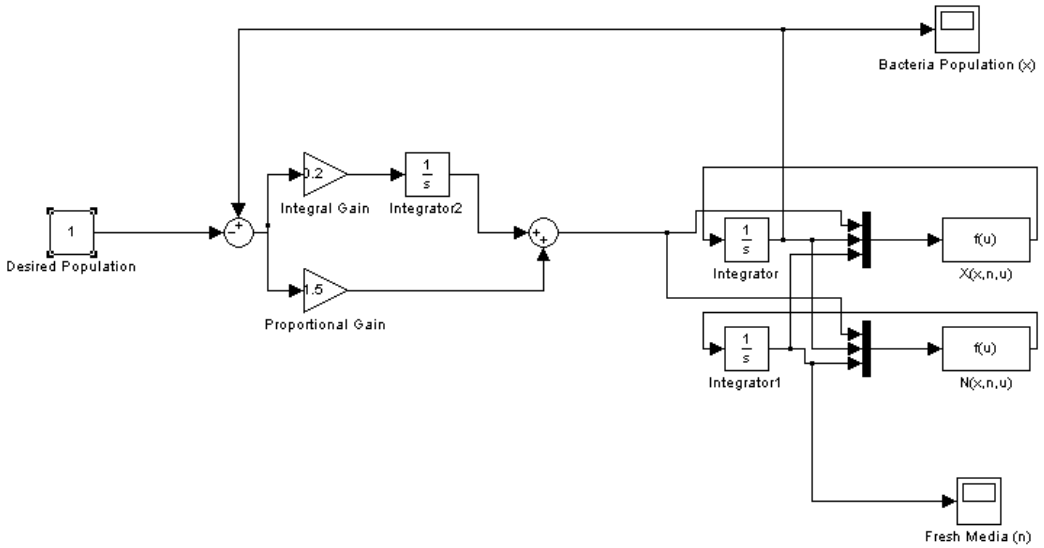


Figure 3 - Model of System with PI Controller and Feedback

The following figures show simulations of our system with our PI controller and feedback with the desired population set to 1 and 2.5 g/L. The controller seemed to act as we would have liked (which we will discuss in more detail in the design choices section) by approaching the desired population of bacteria within a reasonable time.

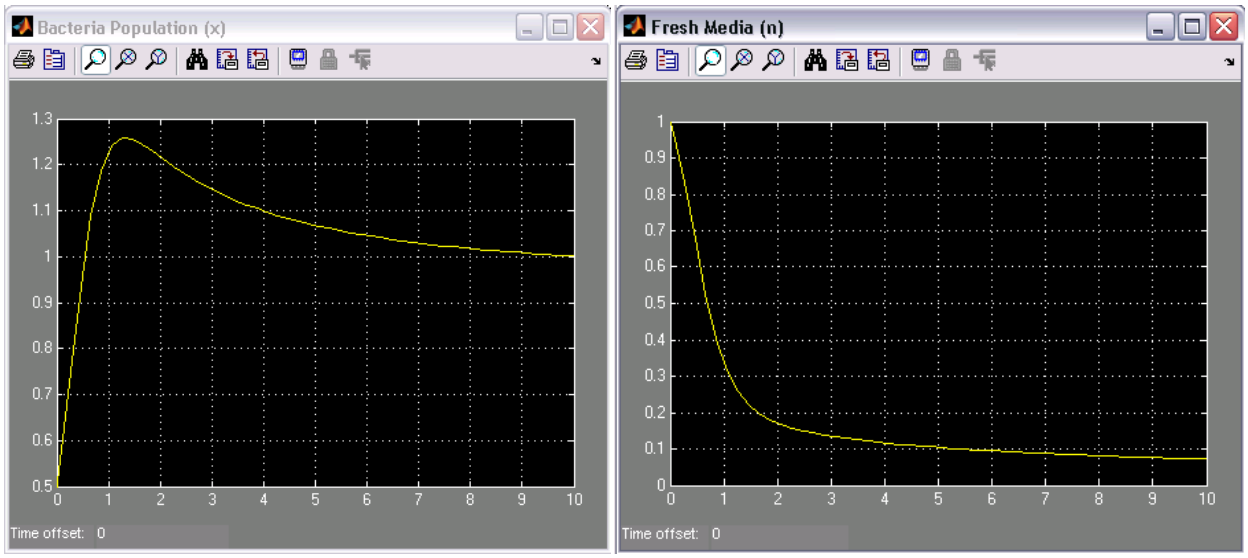


Figure 4 - Simulation with the Desired Population Set to 1 g/L

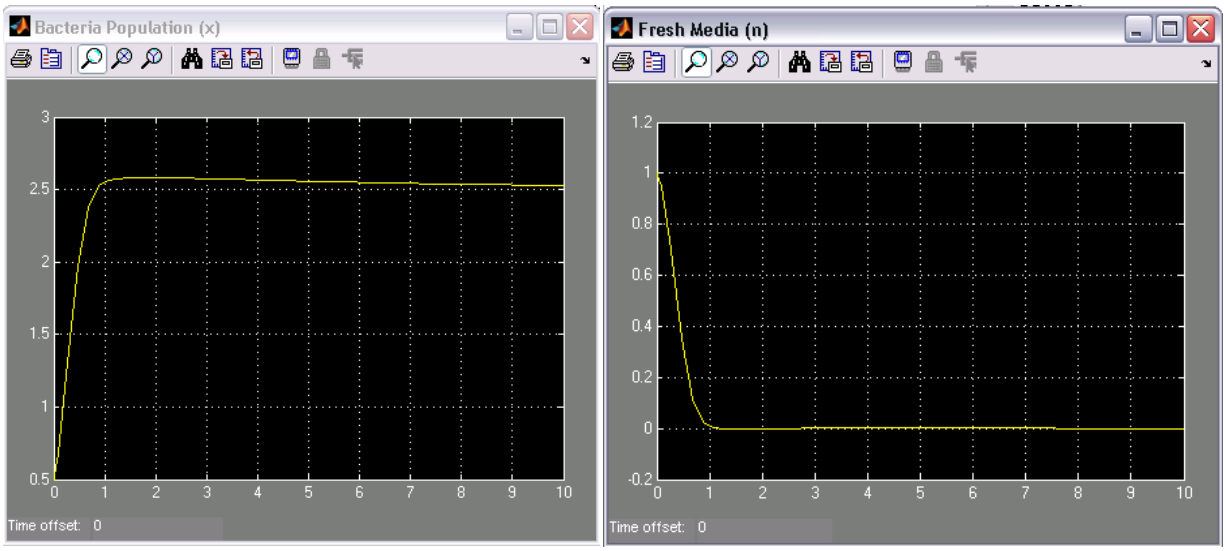


Figure 5 - Simulation with the Desired Population Set to 2.5 g/L

DESIGN CHOICES

When configuring our controller, we decided the main things we wanted to focus on were having a fairly quick settling time and low steady state error. We wanted the system to settle to steady state in around half of a day (12 hours) and that we wanted the final value to be within 0.05 g/L of the desired population. We also wanted relatively low gains that could achieve our goals without creating oscillation. When tweaking the proportional and integral gain, we found that 1.5 for both seemed to be a pretty good balance. If the proportional gain was lowered, there was a large amount of steady state error and if increased, the rise time began to become very high to the point where I don't think our actuator would be able to keep up. When the integral gain was raised, the percent overshoot and rise time increased and when the gain was lowered, the steady state error was increased. We decided to go with a proportional gain of 1.5 and an integral gain of 0.2, because these seemed to create a happy medium for our desired qualities of our system output.

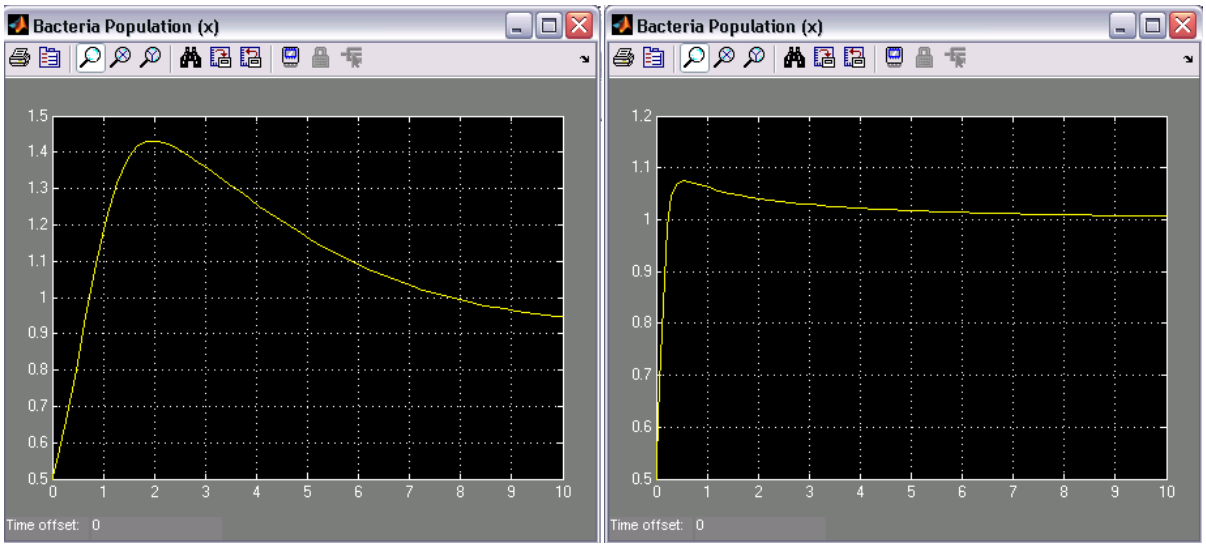


Figure 6 - Bacteria Population with Integral Gain Held Constant at 0.2 and Proportional Gain Varied between 0.2 (left) and 10 (right)

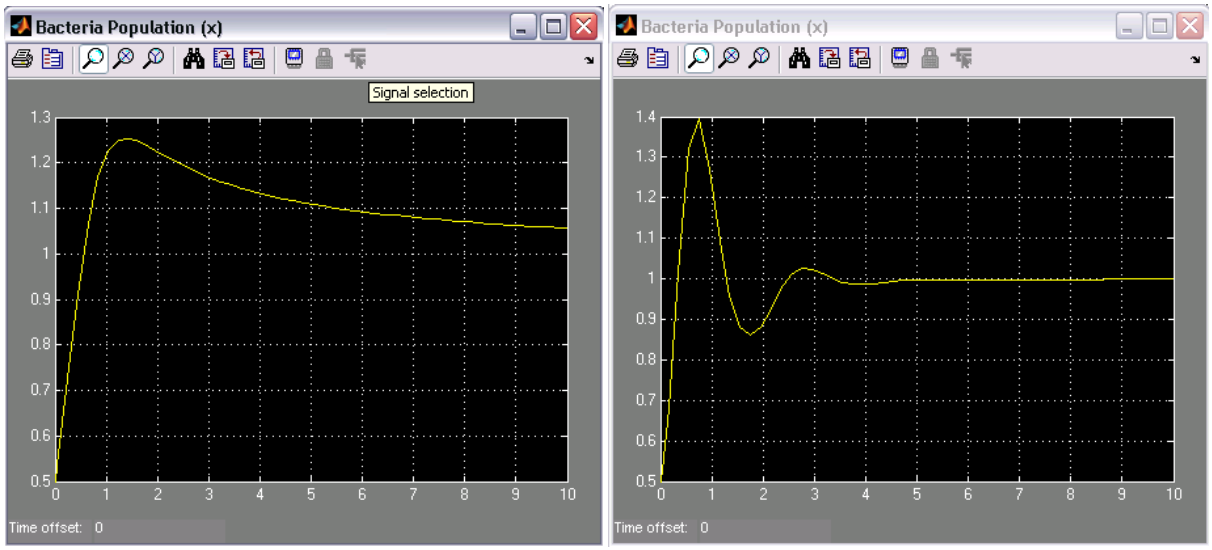


Figure 7 - Bacteria Population with Proportional Gain Held Constant at 1.5 and Integral Gain Varied between 0.01 (left) and 10 (right)

REALISM ADDITIONS

When considering realism in the system, we observed sensor noise and actuator saturation. When the sensor noise was set to a noise power of 0.00001 (with saturation held constant at 1), there appeared to be no affect on the system; however, when this was changed by an order of magnitude of 2, the system began to fall apart as shown in Figure 9. This seems to show that

sensor noise will actually be something that we will need to worry about. However, this should be controlled quite a bit by the current control of our LEDs. We may even need to implement a low pass filter to avoid allowing noise of the light to frequency signal.

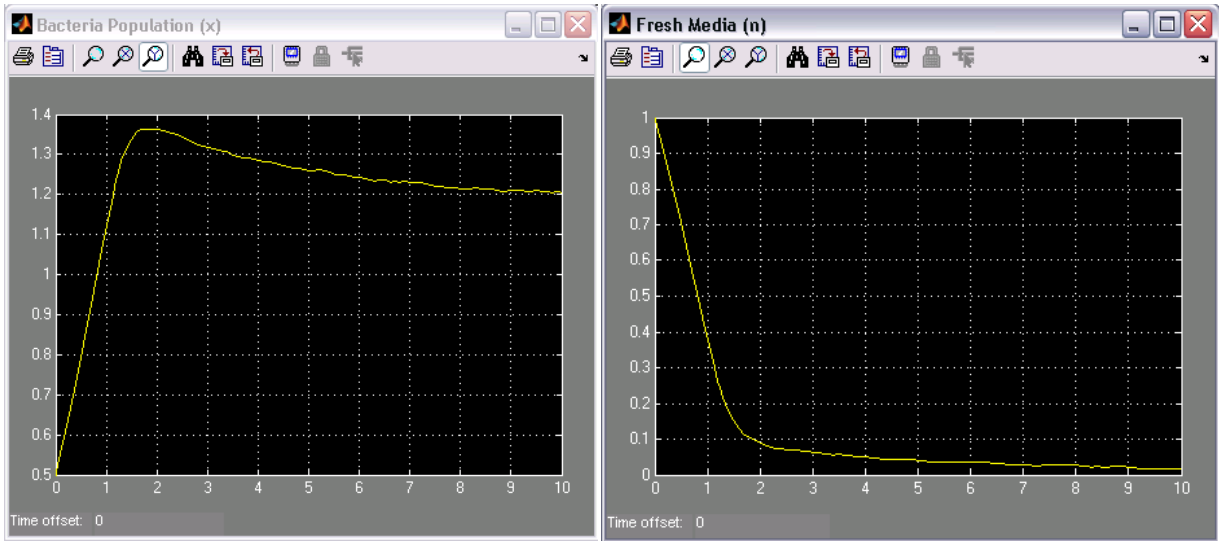


Figure 8 - System Modeled with Sensor Noise Set at Noise Power of 0.00001 and Saturation of 1

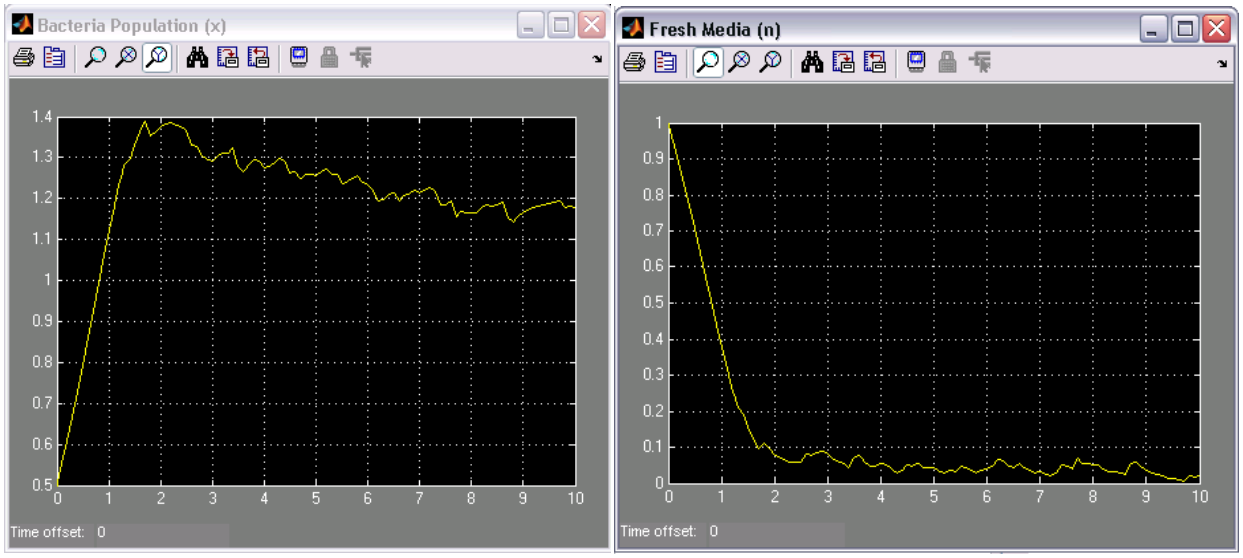


Figure 9 - System Modeled with Sensor Noise Set at Noise Power of 0.001 and Saturation of 1

When Saturation was modified from 1 and set to 0.05 (with noise power constant at 0.00001), the system would not approach an appropriate steady state value. This appeared to be because not enough nutrition could be added to dilute the solution and the population would stay way higher than it was supposed to (as shown in Figure 10). This was a very low saturation point and this is not likely something that we will run into because of its low magnitude.

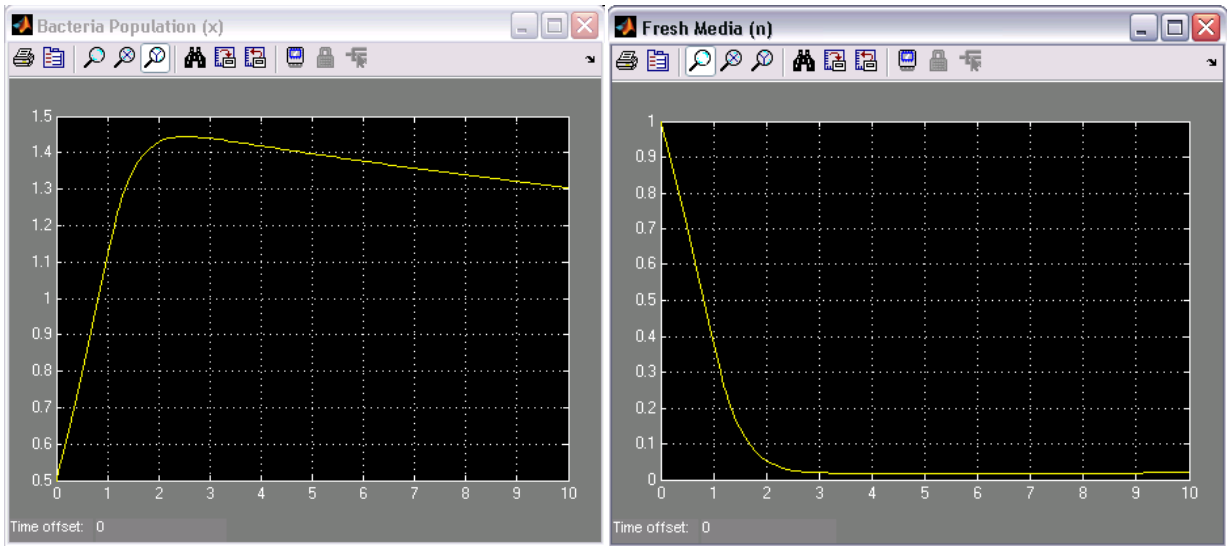


Figure 10 - System Modeled with Sensor Noise Set at Noise Power of 0.00001 and Saturation of 0.05

NOTE: It was brought up during our presentation that our Fresh Media values were higher than the saturation point and we had said that this was a mistake that we had made with the initial value of our media; however, it is alright if the Fresh Media is above the saturation point, because the saturation point deals with u (the output of our controller) and not n (the concentration of fresh media in the system).

Project Status and Planning

The key elements of our turbidostat can be broken up into 3 parts. They are the structural design, circuit design, and software implementation including our PI controller. Over the course of the past few weeks we have been focused primarily on the structural design portion which includes the plastic housing in which the plate will rest. We have also been concerned with the circuit design ensuring that each LED will be able to turn on separately without interfering another photodiode reading. In reference to our initial project plan we are a little behind. We hope to be able to accomplish as shown in the subsequent weeks.

Week	Goal	Action
6	Obtain OD reading from plate scanner	Calibrate plate scanner using blue food dye to learn output frequency relation to known OD
7	Circuit board and plastic housing sent off to be fabricated. Arduino code finished	Finish PCB layout. Make adjustments to plastic housing. Test arduino code.
8	Hardware assembly completed	Solder circuit components onto board. Combine plastic housing with board. Connect arduino and test output frequency (OD)
9	Interfacing hardware with liquid handler	Implement PI controller. Write code for liquid handler. Mount plate scanner on liquid handler.
10	System tuning	Make necessary adjustments to achieve more accuracy. Collect data for report.

Figure 11: Project plan from now until the end of the quarter

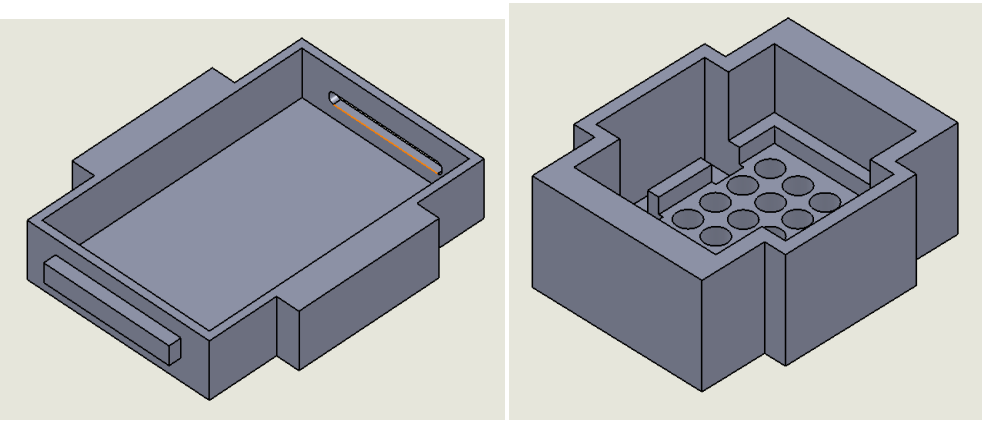


Figure 12: Base and body of plastic housing

We decided to delay the fabrication of the actual plastic housing design since we do not know the dimensions of the PCB, which will be embedded inside. We want to make sure that

everything will fit accordingly to save time and money. Since we don't have either of those parts yet, our goal is to create a prototype of our system using basic components with only a few plate wells for testing. This way we can learn right away the relationship between a known optical density in blue food coloring and the output frequency we obtain from the reading. We want to also learn output frequencies for different optical densities so we can know precisely how our system should respond to real bacteria. Once we learn more about the nature of the light to frequency converter, we then want to add the arduino code into the system to see if we can control multiple LED's with a clock input and obtain different bits of information with output readings. The circuit design however becomes more complicated as we add current regulators and shift registers in order to control each LED and obtain information from them independently.

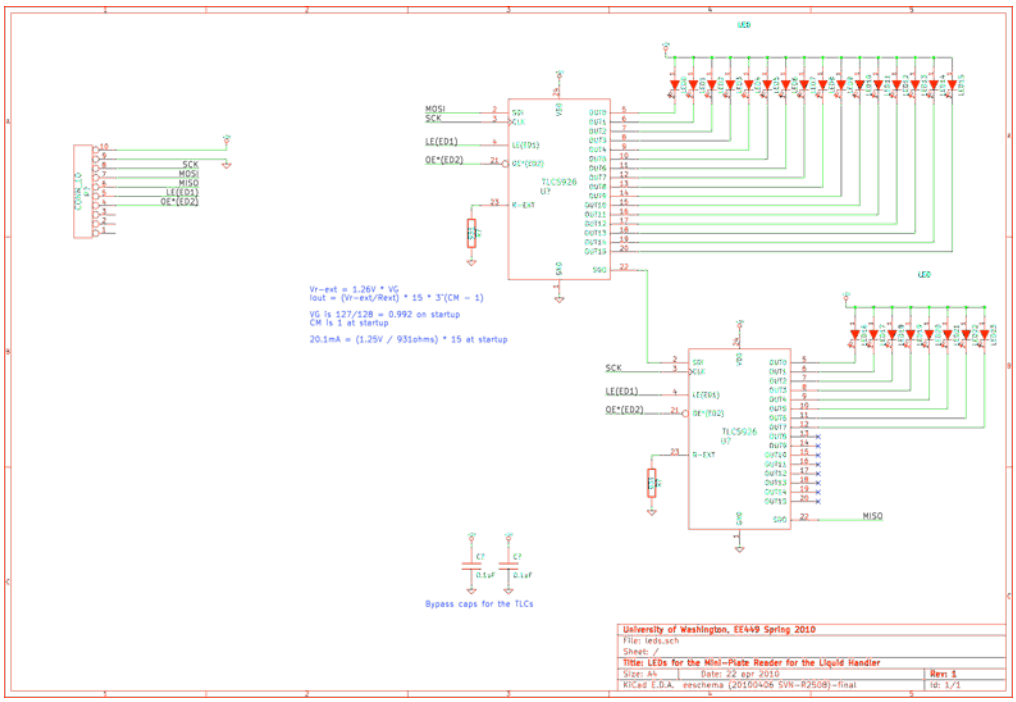


Figure 13: LED circuit diagram including current regulators

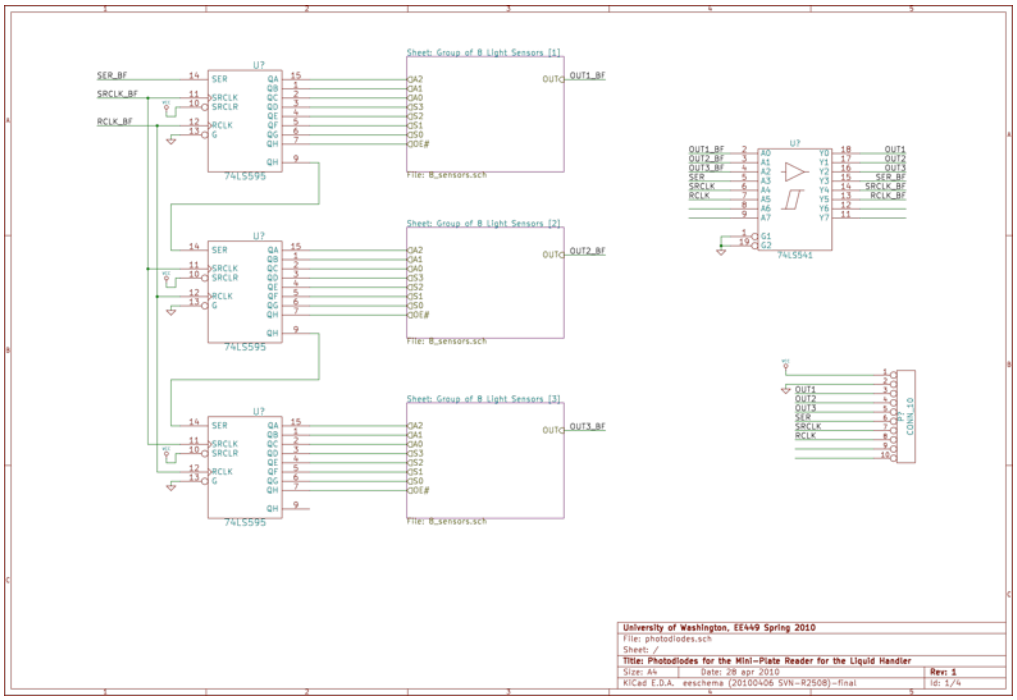


Figure 14: Photodiode circuit diagram with 8 bit shift registers

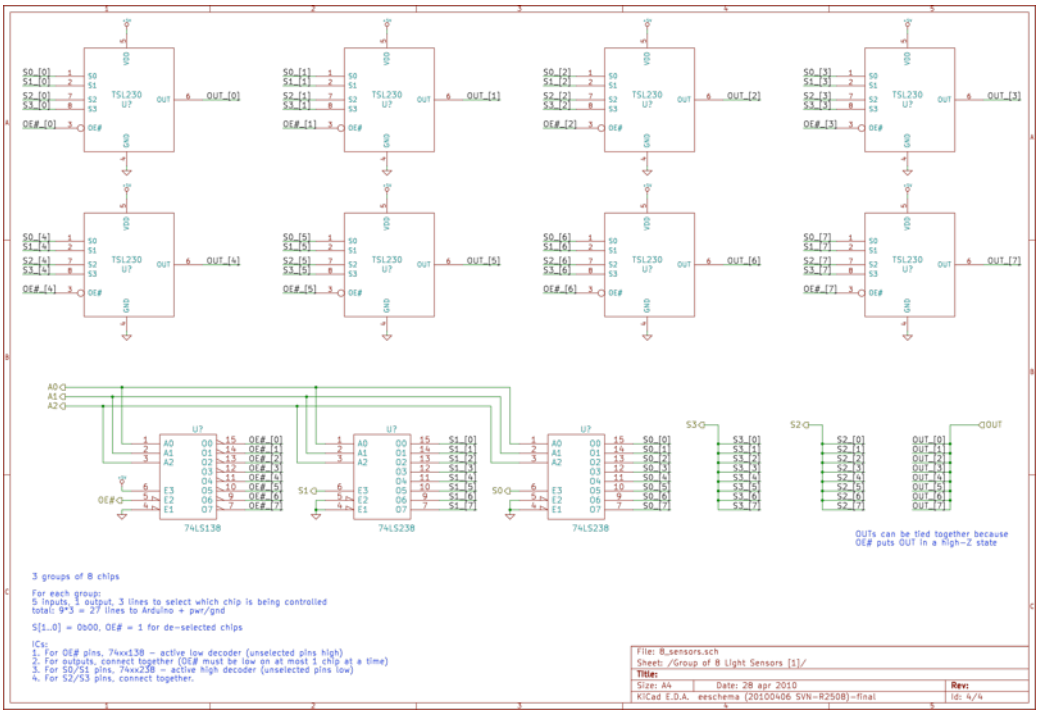


Figure 15: Photodiode circuit diagram with multiplexers and 1 group of 8 photo sensors.

Controller Implementation

One of the more challenging parts of this project will be to interface the plate scanner software with the liquid handler software. We will use an existing chemostat program as a platform to build our turbidostat on the liquid handler. The function of the chemostat program is to read in bacteria volumes and pipette and extract appropriate amounts in order to maintain a constant population. This is like our turbidostat open loop model. Since the chemostat does not have feedback code will be written separately in which the chemostat program will read in to pipette and extract correct amounts thus controlling the bacteria population. The actual PI controller will be written in C on the PC. This will take in readings from the arduino and send them through the controller to the liquid handler.

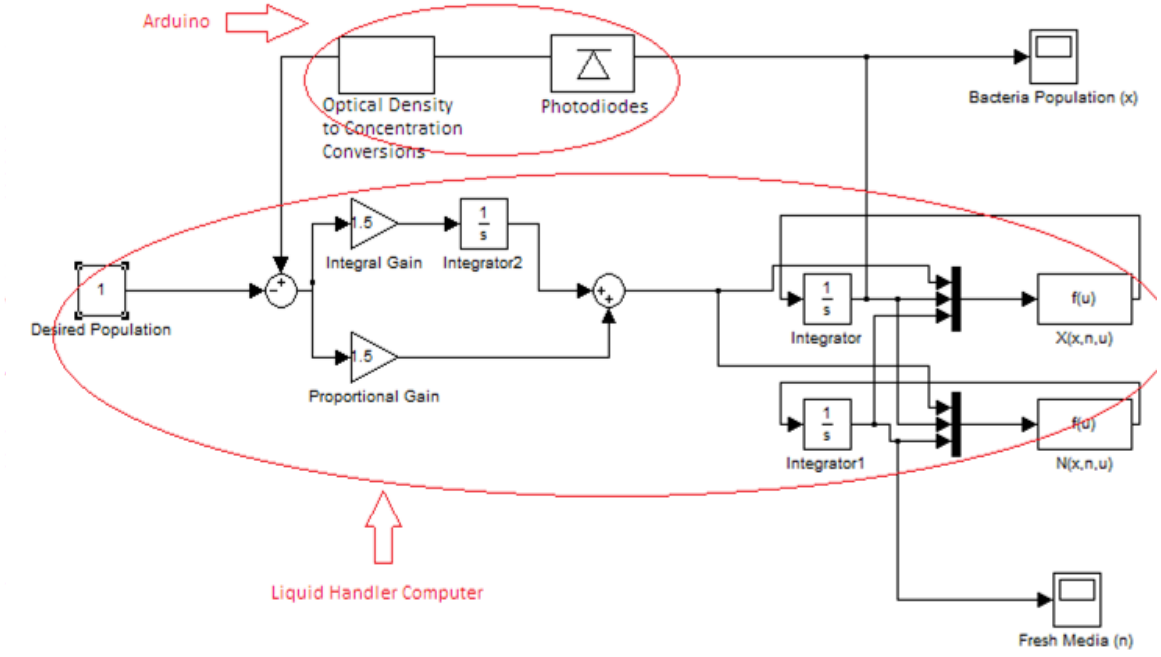


Figure 16: Controller implementation showing liquid handler and arduino portions

CONCLUSION

We have fallen a little bit behind schedule, but we plan on making up for this time by setting a goal of finishing the circuit board and enclosure design and sending these off for fabrication by the end of week 7. We also have a week at the end of the quarter that can be used as a bumper if we face any issues with the fabrication process. Overall, we have followed our schedule fairly closely.

We were able to design a PI controller that met some of our requirements that we set for ourselves. Our final controller accomplished a low settling time, a low rise time, low percent overshoot, and a low steady state error. The integral gain was set at 0.2 and the proportional gain was set at 1.5 to achieve these characteristics. These values were found by inserting the basic PI controller into our model, tweaking the values of the gain, and watching the respective output.

When testing for realism in our model, we found that sensor noise will most likely be something that we will want to worry about and that actuator saturation will probably not be a problem.

The saturation point was so low (0.05) for it to affect the system, that we do not think that we will run into this problem. With the sensor noise, it seemed like very low levels (noise power of 0.001) would really skew the output. We hope to remedy this by using current control on the LEDs and, possibly, by implementing a low pass filter on the output of our light to frequency photodiodes.

Lastly, we plan on implementing the PI controller in software on the liquid handler computer along with a conversion of the output of the photodiodes from optical density to g/L. The sensor interface will be implemented using the Arduino and this information will be sent to a program on the liquid handler computer that stores the concentrations in a file that is accessible by the liquid handler software which, in turn, dispenses the respective amounts to different wells.