AutoGrow

Xavier Humberg[1], Yance Mooso[2], Cody Ngo[3]

***Abstract*— In this document, a system for automatically growing plants is proposed, and the results of building it are exposed. This system automatically cares for the plants with little external influence. The system includes three different, confined systems, with each doing different, interconnected jobs.**

I. INTRODUCTION

The world is becoming increasingly automated. There are machines that will automatically play table tennis against any opponent, machines that automatically assemble and test cell phones, and even machines that automatically cook hamburgers; however, there is one industry that has not been fully automated: agriculture. Over the past century, more and more jobs typically reserved for the farmer have been industrially automated, like shucking corn or separating wheat, but only a handful of people have attempted to automate the more sensitive part of agriculture: caring for the plants.

This project aims to create a prototype for that very process. Though the prototype may not be 100% autonomous, with enough industry experience, this prototype could easily be adapted to accept water and fertilizer from outside sources, removing the necessity to refill the wells.

II. PROPOSAL

A. Design

The proposed system will be designed to grow plants with little external help. It will be able sense moisture in the soil and will provide water and fertilizer to a plant as needed. A light fixture above the plants will be a substitute for sunlight. Periodic checks will be required to make sure that when plants grow, they do not touch the light fixture. These checks will be done with a laser that will sweep the growing area and if the laser breaks due to a plant, the light fixture will be raised higher to accommodate the growing plant. If the levels of either the fertilized water or regular water reaches levels that are too low, an external notification in the form of a light and email will be sent, indicating to the operator that the well is in need of refilling. This will likely be the only area where external assistance will be needed. Periodically the system will collect data on the plants such as height and will be sent to the user via email.

This paper was submitted for review on December 15th, 2017.

[1]Xavier Humberg is a Computer Engineering student at the University of Utah, Salt Lake City, UT 84108 USE (e-mail: x@kuteradio.org)

[2]Yance Mooso is a Computer Engineering student at the University of Utah, Salt Lake City, UT 84108 USE (e-mail: ymooso@gmail.com)

[3]Cody Ngo is a Computer Engineering student at the University of Utah, Salt Lake City, UT 84108 USE (e-mail: u0831637@utah.edu)

Due to the nature of how the system will be designed, there will be multiple components that will need to communicate with each other to form a cohesive system. Each sub-system will first be built and operated individually to make sure they are working as intended before they are all combined.

B. Background

There are only a few commercially available products that automate parts of the plant-care process. The most consumer available is called an Aerogarden [1]. The Aerogarden is typically a small, automatic gardening device that doesn't use soil, but instead uses a special pod sold only by the manufacturer. There is a limited amount of space for seed pods, and the light that comes attached to the system must be manually adjusted to avoid having plants block it. The system, as it isn't fully autonomous, also reminds users (using an app) to feed the plants and add water to the system.

The most interesting part about the Aerogarden is it's water system. It has specific profiles for each plant, and only waters the plants when a tried and true method says so.

Another example of an automated system is FarmBot's "Drag and Drop Farming" [2]. FarmBot is a 3D-printer-style machine that uses interchangeable heads and tools to do almost every last part of the process – from planting seeds and watering plants, to self-weeding its garden. This system is geared towards a full-sized, raised, outdoor garden that doesn't provide its own lighting source. The system can be controlled using an app – hence the "drag and drop" claim.

C. Tasks

The proposed system will consist of six different sub-systems: a soil sensor system used to detect the moisture in the soil, a watering system used to automatically water the plants, a plant detection system used to automatically detect when a plant is too close to a light source, a light control system used to raise and lower the light source as needed, a network system used to send data to an external source, and a control system used to coordinate among and communicate with each system.

The soil sensor system will consist of a soil sensor, a driver, and a Zynq Board (Zybo)-Zynq 7000 Development Board [3]. The driver has six output wires, two wires connect to the soil sensor, one wire caries ground from the board, one wire caries power from the board, and the remaining two wires are used as communication lines between the board and the sensor – likely as an analog to digital system.

The watering system will consist of a pump, electronically controlled valves, a well (likely a bucket), a float sensor, and tubes extending from the well to the soil. This system will be

Fig. 1. The first working prototype of the plant detection system

controlled by the Zybo-Zynq 7000 Development Board [3], and will be directly triggered by a combination of timing circuits and the soil sensor system, with some plants being watered as soon as their soil reaches a certain threshold, and others being watered after a small delay (depending on what the plants need). These plants, and their respective profiles, will initially be hard-coded into the system. The float sensor will create an error code when the well empties to a certain threshold.

The light system system will consist of two fluorescent light units each with two T5 bulbs and a self contained ballast packaged on a frame built out of extruded aluminum.

The plant detection system will be mounted on the light system and will contain a laser scanner that upon an enable signal from the controller will scan the plant bed area to detect the height of the plants. When plants are detected due to blocking the laser and sensor unit, it will send a signal back to the main control system to raise the light system and will rescan until plants are no longer detected. The light system and plant detection system will be raised using two stepper motors actuating two lead screws that the light control system is mounted on using brass lead screw nuts and two linear rails with guide shaft bearings mounted on the frame next to the lead screw nuts. The plant detection system will be controlled using an Arduino Uno [4]. The Arduino Uno will also be used to raise the system using stepper motors to rotate the lead screws using a microstepping driver. The initial prototype of this system can be seen in Figure 1.

The network system will consist of a Particle Photon Wifi Microcontroller [5], the Zybo-Zynq 7000 Development Board [3], and possibly an external computer to receive requests. This system will receive error codes from the other components, which will be collected within the Zybo, and will, at least initially, send an e-mail to a set of pre-programmed addresses when something goes wrong.

The control system will consist solely of the Zybo-Zynq 7000 Development Board [3]. This system will control the timing and communication among the different systems, and will be in direct control of sending the error codes to the network system.

D. Stretch Goals

Upon completion of the above mentioned system, we'd like to extend into stretch goals. These goals extend into two categories: creating an app and replacing driver boards.

Much like FarmBot [2] and Aerogarden [1], we'd like to create an app for our system; though, this app wouldn't be nearly as complicated as FarmBot's. This app would replace e-mail alerts and offer a small amount of statistics tracking by noting when the plants were watered, as well as the height of the light.

We'd also like to replace the stepper drivers with those of our own design. If we made it to this stage, we would create custom circuit boards for the purposes of controlling the stepper motors in our circuit. As there will be two different stepper motors in use, we'd likely have to create two drivers for different power ratings, though duty cycles will likely be small enough that cooling shouldn't be an issue.

E. Components

We will be using several purchased, prepackaged components to make our subsystems and to comprise our full system. A few of the components are subsystems in themselves such as the stepper motor drivers and the fluorescent light units. For the initial project we plan to use these pre-made, purchased systems. If we have enough time, we would like to replace the stepper drivers with our own designed driver circuit and custom made PCB boards (See Stretch Goals for more information).

F. Resources

We will be using three different development environments to program each of our boards. The Zybo FPGA [3] will be programmed using the Vivado Design Suite from Xilinx [6]. The Particle Photon [5] will be programmed using its own web based IDE [7]. The Arduino Uno [4] will be programmed using the Arduino IDE [8].

Many complex components are controlled by drivers or from peripherals within the various microcontrollers. Please see Stretch Goals for which components we'd like to try replacing with custom-made components.

G. Task Interface

To coordinate our efforts and keep tabs on progress, we will be using Basecamp as a task interface. We will focus most of our use on the "To Do" section, and will update the information within the section regularly.

III. SPECIFICATIONS

A. Testing and Integration

Each sub-system will be tested individually. This will ensure that when the sub-systems are combined, the complete

proposed system will work correctly and will assist in correcting any problems found in each sub-system.

The soil sensor system will need to correctly detect the moisture levels in the soil. This can be tested in a small soil environment where varying amounts of water are poured into the soil and the output is measured.

The water system will need to be able to deliver either fertilized water or water to a given plant. Testing on this system will ensure that only one type of water can be delivered to a certain plant at a time. A test will also be done to make sure that the specified type of water is delivered to the correct plant location and to make sure that it is not watered more than needed.

The plant detection system should report when a plant is too close to the light. This can be tested by placing an object that will interfere with the laser and seeing if the system correctly reports that there is an interruption.

The light system must be able to raise its height when a plant grows too close to the system. To test this system, the stepper motors must be able to raise the light. From there, this system can be combined with plant detection system so that when it senses that a plant or object has triggered this system, the light system will respond by raising the light.

The network system will send data to an external source over the Internet. This system will be tested by first setting up the Photon Wifi Microcontroller to send an email. From there, additional tests can be conducted that will trigger an email to be sent based on either a given stimulus or data received externally.

The control system can only be tested once each sub-system is working as intended. The control system will be tested through the use of manual triggers on the Zybo-Zynq 7000 Development Board [3] that will trigger a response in a given sub-system. The sub-system should then correctly perform its intended operation. For example, pressing the light system switch on the Zybo should send a signal to the light sub-system to raise the light.

The test to see if the proposed system is working as intended will be planting seeds and letting the system operate on its own; thus, the plants should grow. Due to limited time and resources, small and quick growing plants will be selected such as herbs and beans. This will provide quick and observable results to ensure that the system is operating correctly.

B. Schedule and Milestones

We are aiming to get the plant detection and some light control sub-systems designed and prototyped by the end of April, with the rest of the light controller hardware, namely the z-axis and overall frame, prototyped by the end of May. We are also aiming at getting the full watering system prototyped by the end of May, alongside having a solid understanding of the various sensors and each microcontroller. From there, we'd like to create and solidify a communication protocol for the boards by mid-June. By the end of June, we'd like to get a full prototype of the network and control systems. From there, we would move on to stretch goals and design finalization.

Towards the beginning of the project, some milestones will be accomplished individually so that each member will learn about a component for each subsystem. This allows for parallel learning of a component for time management. Xavier Hummberg will be in charge of learning about the Zybo, Cody Ngo will be in charge of learning about of the Photon, and Yance Mooso will be in charge of learning about the Arduino. By the end of May, each project member should have a good understanding of how their component works. The end of May should also see a prototyped version of the light control system and watering system. Each project member is also expected to learn about the soil moisture sensor that will be a part of the watering system. These milestones will be a team milestones as the scope is much larger and a general understanding of the complete system is beneficial for each team member when moving forward.

From the initial understanding and prototypes, the team will be able to move forward to be able to create and solidify a communication protocol among the boards of each subsystem. This will be expected to be finished in the middle of June. Once this protocol has been established, a full prototype of the network and control systems is expected for the end of the month.

The final prototype for the entire proposed system is set for the end of July. This gives us time for the rest of summer and fall semester to fix any issues or move onto stretch goals.

C. Risk Assessment

Though our project may seem rather simple upon first glance, there is a surprising amount of risk involved in this project. The main problem we'll run into is the endstops on the laser scanning system. They require a decent amount of pressure to trigger, and can therefore desynchronize the laser scanning system (this has happened a few times already). The system will need to be perfectly synced to work in the way we plan. If this proves to be a problem, we plan to either swap out the endstops for another sensor or create a lever system to decrease the necessary force.

We also, currently, haven't found a good way to pump water into our system. Though we could do a gravity fed, the water flow-speed would vary based on the amount of water in the well, and refilling the well could lead to pouring water to the circuit. Many problems could also stem from improperly sealing the tubing to the bottom of the bucket. However, when it comes to pumps, most pumps we could purchase for this project are too powerful for the purpose of gently watering plants. This could lead to over-watering, drowning, or even blasting away plants. There are a few alternative methods to pumping water we could use if needed, including using air pumps in an enclosed system, returning to a gravity fed concept, and loop-back techniques for splitting tubes.

Also, there is an inherent danger working with water and electronics. Though most systems will be kept above the plant bed (and hopefully the watering system), systems could still get easily ruined.

TABLE I
PARTS LIST (PART I)

QTY	*Part*
As Needed	V-Slot Linear Rail 20mm x 20mm extruded aluminum
8	Delrin Mini V wheel
16	MR105ZZ 5mmx10mmx4mm bearings
4	F623ZZ Flange Bearing 3x10x4mm
As Needed	eSUN 3mm PETG Natural Filament 1kg
As Needed	Misc Nuts, Bolts, Washers
1	A4988 Stepper Motor Driver
1	TB6600 4A 9-42V Stepper Motor Driver
2 Units	Yescom T5 2ft Grow Light
1	MSP432 Launchpad
1	Particle Photon Wifi Microcontroller
1	Arduino Uno Microcontroller
2	Zinc Alloy 8mm Inner Dia 55x13x30mm Pillow Block Bearing
2	Aluminum 8mm Innner Dia 42x32x11mm SK8 Linear Rod Rail Support Guide Shaft Bearing
2	Aluminium Alloy D19L25 5x8mm Flexible Shaft Coupling
2	8mm Inner Dia L34.5mm SCS8UU Linear Bearing
2	8mm L500mm Linear Shaft Optical Axis
2	Stainless Steel and Brass L500mm D8mm 2mm Lead Screw Nut
2	Nema 23 CNC Stepper Motor 2.8A 1.26Nm(178.5oz.in) 23HS22-2804S
2	Nema 17 Bipolar Stepper Motor 3.5V 1A 13Ncm(18.4oz.in) 17HS08-1004S
48 in	1/4 in. x 48 in. Plain Steel Round Rod

TABLE II
PARTS LIST (PART II)

QTY	*Part*
8	250V 5A SPDT 1NO 1NC Momentary Hinge Roller Lever Micro Switches 3 Pins
As Needed	Crystal Clear Cell Cast Plexiglass Sheet
1	12v 30a Dc Universal Regulated Switching Power Supply 360w
1	Solid State DC Relay
1 Spool	Wire
1	Receptacle/Switch for 120v power
2	GT2 Pulley
2 Meters	GT2 2mm pitch 6mm wide Timing Belt
As Needed	wire heat shrink
4-8	Soil Moisture Sensors and Drivers
2	Float Sensors
1	Infrared Photoresistor
1 Pack	Zip Ties
1	Water Pump
4-8	Water Solenoid Valve
6 Meters	Vinyl Tubing
2	Water Tanks Reservoirs,Buckets,Containers
As Needed	Various Electronic Components (Capacitors, Resistors, LED's/Diodes,Power Regulators, etc...)

TABLE III
SOFTWARE AND HARDWARE

Software
Particle Web IDE (Build) [7]
Adobe Illustrator [10]
AutoDesk Fusion 360 [11]
Lulzbot Cura [12]
Lulzbot Taz 6 3D printer [13]

A large portion of this project will involve 3D modeling and printing. Some parts may be hard to build due to the limitations of 3D printing, while others may just require a large amount of work in a 3D modeling program. These are skills that come with years of practice, but can easily be created by somebody else if it becomes necessary. It seems as though most of our issues are already resolved here, but if parts break due to improper design, we may still need to consult somebody.

The last main concern is in the strength of the Z-direction stepper motor. Due to the sheer weight of the current light fixtures, we will need a much more powerful motor than for your x-direction scan. Currently, we're hoping the Nema 23 stepper motors [9] we ordered will do the trick. If not, we will have to create our own to keep the project within our price range.

D. Parts list

A full parts list for this project can be seen in TABLES I and II, and a full hardware and software list can be seen in TABLE III.

IV. THE COMPLETED PROJECT

In the end, the system ended up being very similar to what we had originally planned. A few decisions, such as using a more professional board and removing the FPGA, were made in the process that changed the direction of the project, but none were significant enough to have a major impact on the final product. In the end, our machine had taken a pepper plant from seed to fruit, and we're fully confident it will be able to continue growing the pepper all the way through into seed again.

Our final design consisted of three distinct, yet interconnected pieces.

A. The Watering and Soil Sensing System

The watering system was left as proposed, except that it was combined with the Sensor System since the two were so interconnected. The only major difference in the system was the use of an MSP432 instead of a Zybo, as we didn't need many of the benefits provided by digital circuitry in the FPGA. The system still consisted of a bucket, a float sensor, a pump, four solenoids, some tubing, four capacitive soil sensors, and a custom PCB.

The soil sensors proved to be troublesome throughout the semester. Several different resistive sensors were tested, and each one proved unreliable. As of writing this paper, we're still yet to have an issue with the capacitive sensor after installing it a month prior.

The PCB for the watering system (seen in Figures 2 and 3) was created to handle two pumps, each connected to four solenoids, for a total of eight watering lines. The PCB consisted of switch-transistors and a decoder meant to allow only one plant to be watered at a time. This allowed us to

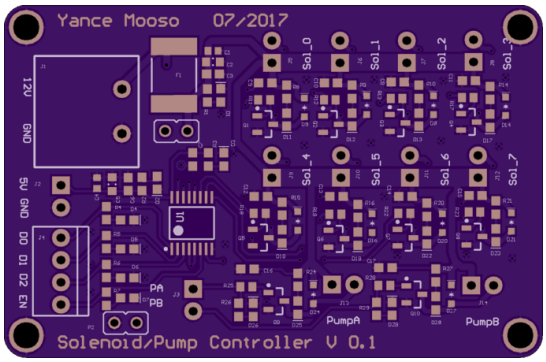

Fig. 2. A rendered version of the PCB for the watering system

precisely control the amount of water going into each plant, without having to worry about pressure loss.

B. The Light and Detection System

The light system proved to be the most challenging part of the project, from assembling all the moving pieces to assuring the scanning system was always aligned, the bulk of our project was spent working on this system. The first prototype of the scanning system can be seen in Figure 1. In this system, the steppers would pull the motor to the left, which would activate the laser. The gantry would then scan, keeping track of whether it lost contact with the light. If it did, it would move up and scan again until it completed a cycle without an interruption.

This method was far too slow and inefficient for plants with curved leaves (like drooping corn, for example), so we decided to have the system scan in both directions, and to move up as soon as a break was detected. This allowed for a scan to detect and fix a vast majority of the cases in the first one or two passes. However, moving the light vertically in the middle of scanning proved problematic due to the laser going slightly askew when moving up. This gave us, on more than one occasion, a false positive that forced the light to raise to its maximum before the laser finally aligned again.

To solve this problem, we switched from a laser to a clear LED, as those have some of the directional properties of lasers without the precision required by a laser detection system. This led to one final problem: because we weren't using the laser, our light sensor had to be configured to a point where the grow lights could trigger a false negative, making the machine think that the plants never approached the light. Our solution was to shut off the lights during scanning.

That proved difficult as well, as now we had to wire the 120 V AC from the wall into our system and have our board (which couldn't exceed a 12 V input) be completely isolated. We ended up using a solid state relay that the MSP432 was capable of driving on its own.

Once scanning was set up, we installed limit switches at the top and bottom of the movement system to ensure the machine didn't attempt to tear itself apart. The system was

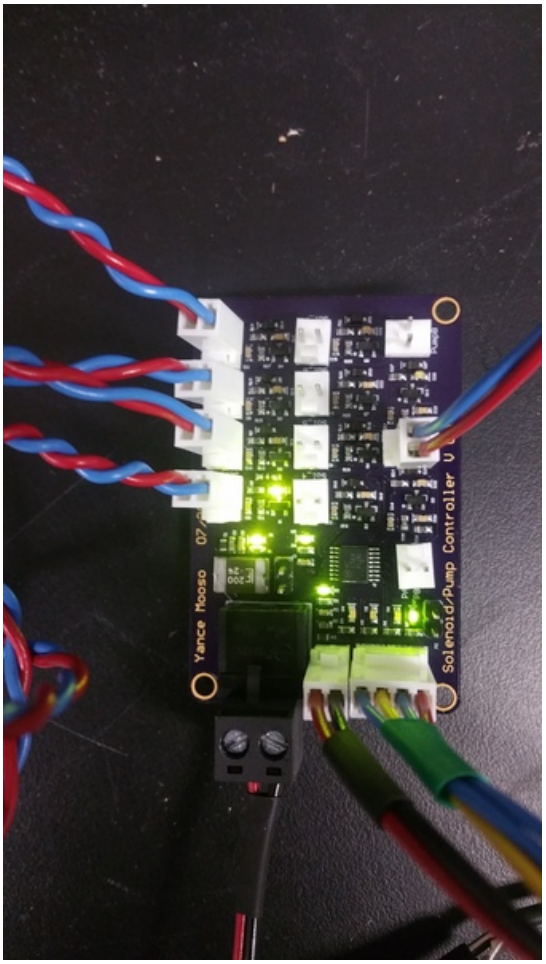

Fig. 3. The wired up PCB for the watering system

designed to scan upon hitting the lower limit switch, and turn off the light upon hitting the upper limit switch.

In the final version of the system, the light controller also requested the current time from the network system and ensured that the grow lights turned off between 8pm and 8am.

The gantry system also had a custom PCB (seen in figures 4 and 5) designed such that interfacing with the gantry system was trivial. It contained all the necessary components to wire the board to the stepper drivers, as well as the limit switches, and contained some LEDs for debug information if something went wrong.

Another custom PCB was created to allow easier control over the stepper motors. In our project, the two vertical stepper motors would always step in the same directions, while our horizontal stepper motors would always be opposite. To avoid having to control each of these motors individually, we created a PCB that used jumpers to determine whether parallel stepper motors would step in the same direction, or in opposing ones. This can be seen in Figure 6

C. The Network System

The network system was implemented as was proposed. Emails were sent to the user based on messages received

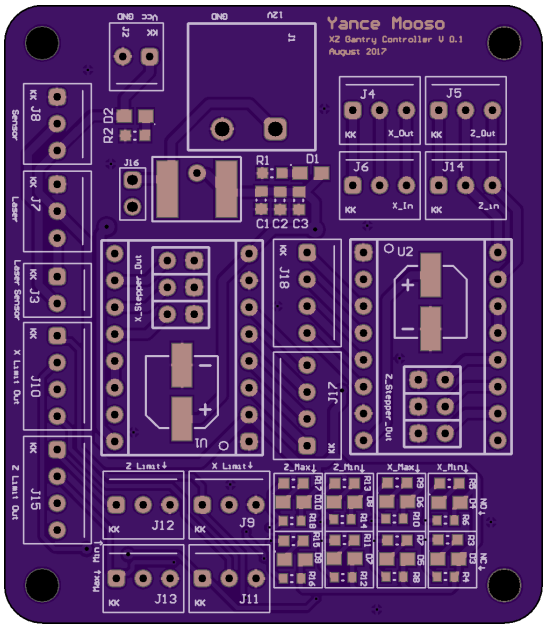

Fig. 4. A rendered version of the PCB for the gantry system

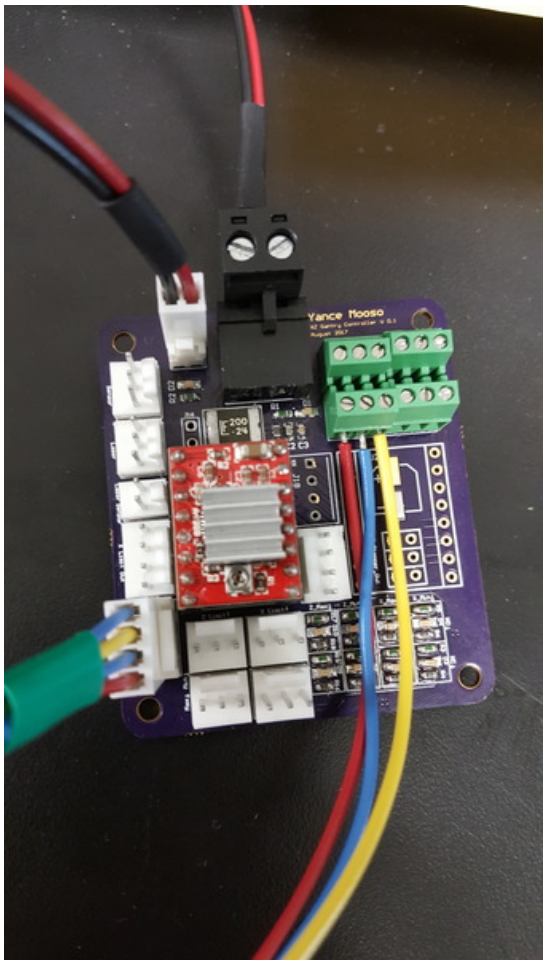

Fig. 5. The hooked up gantry controller

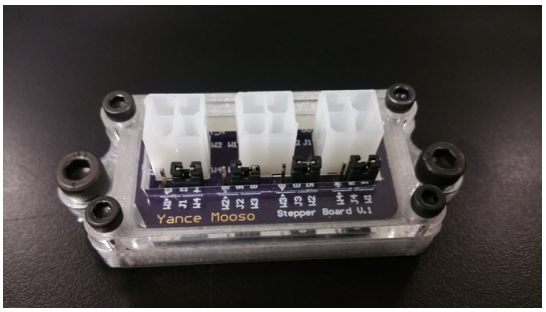

Fig. 6. The parallel stepper splitter PCB

from the other systems, such as when a plant was watered and if the water for the watering system needed to be refilled. As an additional feature, an email is sent at the end of each week that includes different information about the system such as how many times a plant was watered.

In addition to emailing the user about the status of the system, the network system was used as a real-time clock that helped in notifying when certain events occurred as well as triggering different events based on the time of day. An example of this kind of event is using the time of day to have the lights for the plants switch between on or off. The Particle Photon acquires this time from the Particle Cloud servers once it connects to the internet. As long as the Photon has access to a WiFi network, the time is able to be retrieved.

Communication between the MSP432 and the Particle Photon was achieved using a UART serial connection between the two devices. Integers representing different events were sent from the MSP432 to the Photon and were then processed to be able to send an email. In the opposite direction, time was sent from the Photon in 4 digits that the MSP432 had to process to be able to used for different functions.

D. Connecting Everything Together

As can be seen in Figure 7, almost all of our routing was done inside a box that was mounted to the back of the project frame. The box contained our 12 V power source, 5 V power source, solid state relay, custom watering system PCB, custom gantry system PCB, stepper motors, and power switch.

E. Demoing

Due to the slow growing nature of plants, a series of time lapse videos were made of the project prior to demo day. The videos showed various angles of the project over a 5-10 days period, each showing project behaviors like plants sprouting and growing, plants being watered, the light system raising as needed, and the light system turning on and off throughout the day. Corn was used as demo plants for its fast growing abilities that led to interesting time lapse videos. The videos were made by using a Raspberry Pi 3 connected to a camera that took a photo every 5 minutes over 5-10 days each. The videos were compiled as an FFMPEG at 30 frames per second. The videos were played during demo

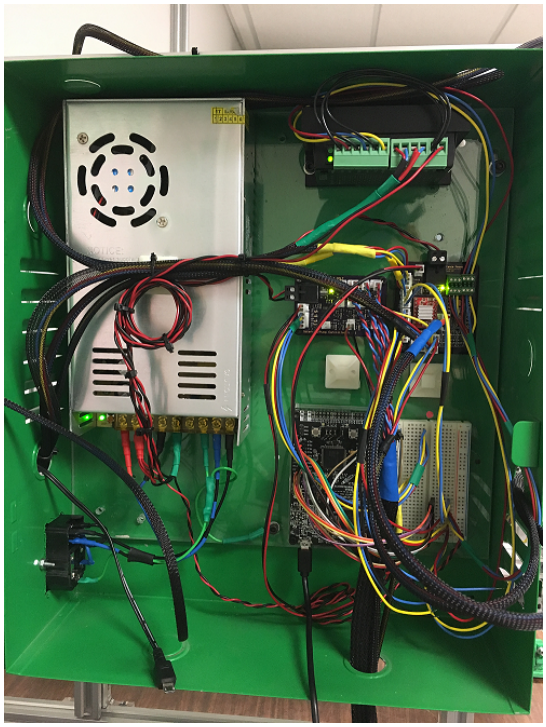

Fig. 7. The innards of the box that wires everything together

Fig. 8. The completed project including a timelapse

day on a monitor. To show individual functionality of the different subsystems of the project on demo day, an override controller was designed and built. The controller consisted of 4 momentary push buttons and two switches. The top left button was used to raise the gantry until the button was released, at which point the scanning system activated and scanned the current level to determine if it was safe for the plants. The middle-top button is pressed to tell the scanning system to find the top of the plants and positions the light gantry at a safe distance. The right button moves the gantry down and scans. The bottom two switches are used to represent 4 possible plant designations when using the momentary button on the bottom to selectively water a single plant for 20 seconds.

Despite being designed for demo day purposes, the override controller turned out to be a very useful device for day to day operation of the system.The override controller can be seen in Figure 9.

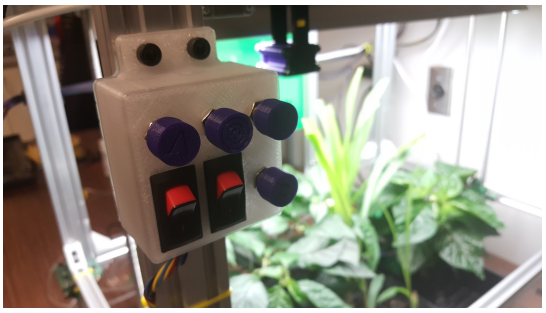

Fig. 9. Override control

Fig. 10. A test of using a snoot to restrict light pollution produced by the grow light on the plant scanning system

F. Risk Management

The potential risks outlined earlier in this paper discusses possible problems with the laser scanning system, the watering system, 3d printing, and the strength of the z-axis. These risks were alleviated with careful design and engineering practice.

The use of a laser diode on this system, coupled with a photo-sensor ended up being an issue.The laser ended up being too precise, resulting in false reads due to jitter in the device. We looked into using an IR emitter and IR sensor. We determined that IR had too much potential risk in false reads due to IR signals bouncing off of different surfaces. The method we decided on, was to directly replace the laser with a bright white LED that shined directly across the light to the photo-sensor. This lead to a problem with light pollution from the main grow light causing the sensor to never detect a break in the LED light. The first solution was to create a light snoot that restricted the angle at which light can enter the photo-sensor chamber(Figure 9), this still seemed risking so we solving this issue by having the grow light shut off before scanning and then turning on after the scan if permitted by the current time settings.

We were initially concerned with having high voltage electronics around water. This ended up not being an issue because all of the electronics were positioned higher up on the project, the watering system didn't have any leaking issues, and the plants were inside of a run off container for

collecting excess water for initial testing.

The 3D printed components on the project ended up being more than adequate for what they were needed for. There was very little stress on the components, laser cut acrylic was used where ever possible for ease of fabrication, strength and cost.

There was a concern about the strength of the Z-axis motors for lifting a heavy light system. We tried to avoid this issue by keeping the lighting and scanning system as light as possible, as well as having tight tolerances on the project frame to keep the lead screws straight and parallel. The smaller NEMA17 motors worked alright, but we ended up using larger and more powerful NEMA23 motors that are more than enough for their application.

V. Summary

In this paper, we proposed a system that was designed to grow plants with little external help. This was accomplished by combining multiple systems together, each in charge of a specific task such that one could not interfere with another. We worked through summer semester to complete most of the project, with fall being dedicated to bug fixing and software. Our milestones were much loftier, so we were unable to get the full prototype in the summer.

Management of team responsibilities and communication was made easier through the use of Basecamp, which allowed us report on our contributions to the final product as well as discussing the current state of the project. Overall, we learned many things about developing continuous systems, as well as several good wire wrapping, connecting, and routing procedures (thanks Yance!).

References

[1] *AeroGarden*, https://www.aerogarden.com/, Miracle Grow, accessed: 2017-03-23.

[2] *FarmBot, Open-Source CNC Farming*, http://farmbot.io, accessed: 2017-03-23.

[3] *Zybo Zynq-7000 ARM/FPGA SoC Trainer Board*, Digilent Incorporated, 2 2017, rev. B.

[4] *Arduino Uno*, https://www.arduino.cc/en/main/arduinoBoardUno, Arduino, 2014.

[5] *Photon*, Particle, 2016, v014.

[6] *Vivado Design Suite*, Xilinx, 2016, version 2016.4.

[7] *Particle Web IDE (Build)*, https://www.particle.io/products/development-tools/particle-web-ide, Particle, 2016.

[8] *Arduino IDE*, Arduino, 2016, version 1.8.2.

[9] Stepper online. [Online]. Available: http://www.omc-stepperonline.com/nema-23-cnc-stepper-motor-28a-126nm1785ozin-23hs222804s-p-108.html

[10] Adobe illustrator. [Online]. Available: http://www.adobe.com/products/illustrator.html?sdid=KKQML&kw=tgcontrol&mv=search&s_kwcid=AL!3085!3!155829415155!e!!g!!adobe%20illustrator&ef_id=WP@MiAAAADuVJDlG:20170425185614:s

[11] Autodesk fusion 360. [Online]. Available: https://www.autodesk.com/products/fusion-360/overview

[12] Lulzbot cura. [Online]. Available: https://www.lulzbot.com/cura

[13] Lulzbot taz 6 3d printer. [Online]. Available: https://www.lulzbot.com/store/printers/lulzbot-taz-6