Self Watering Plant

CSCE 462 Project Proposal

We are FARMERS!

Kerry Zeng

Curtis Green

Jingyu Lu

Hoang Trinh

Hunter Lukken

Department of Computer Science and Engineering

Texas A&M University

10/7/2017

1 Executive summary

The agricultural department is currently lacking a way to efficiently water individual plants. Our product solves this problem by implementing a proof of concept system that will reduce the amount of necessary human intervention by first automating the watering system and then establishing a link between the watering system, the server, and a moisture sensor. The server functions as both an intermediary between the distributed watering systems and the microcontrollers as well as a small storage unit to supply data to the centralized front end that can manage the moisture cutoff values and add extra timings as desired. Upon completion this project should provide a working proof of concept design that would assist the agricultural department by reducing the amount of necessary labor as well as giving them access to a more precise tool they can use within their own research studies.

Introduction

1.1 Problem background

The plants in the agricultural department's greenhouse are maintained manually by students. This includes manual watering, fertilizing, etc., creating many hours of menial work for the students. Additionally, during the holidays, there is a shortage of students, which makes caring for these plants much more difficult.

Needs statement

There is a need for automation in the current workflow so that the student's time is freed up for more important tasks. The department needs a way to automatically water the plants using a time based system or a moisture based system.

Goal and objectives

The goal of the project is to create proof of concept system that automatically waters individual plant samples whenever the soil in the samples gets too dry. This system will act like a demultiplexer for the water so that only certain samples of plants will get watered whenever the moisture levels get too low. Additionally, this project will have a web application to go along with it so the user can edit the settings and visualize the performance of the system. The objectives consists of 3 parts: costs, robustness, and precision. We have to make sure that the design costs do not exceed 500 dollars (100 dollars per person), not including the instruments provided to us. Also, since we will not be maintaining the system, the system must be able to run ad infinitum without user inputs; in the case that it needs maintenance, we must provide proper documentation so anyone can modify or maintain the system. Lastly and most importantly, we must have a highly accurate system. This device will most likely be used in experiments and studies done by the agricultural department so we must ensure that the sensors are highly accurate so that they are not the source of any confounding variables.

Design constraints and feasibility

There are several constraints that the project will follow, including future proofing, scaling, and retrofitting. The device has to be future proof because the people that will be maintaining it will not have an in-depth knowledge of our system. In order to do this, we have to have durable components, multiple fail safes, and good documentation. Another constraint is the scale of our project. Due to our limited budget and time, we can only create a proof of concept. This proof of concept will be a small scale version with the ability to water three "branches" of pipes, but we must design our system so that it has the potential to handle more or less branches if needed. Furthermore, our parts must be compatible with

the current system. We want our design to be practicable so retrofitting it to their current workflow would be ideal.

Literature and technical survey

There are already plenty of irrigation systems out there that do similar things, but most of them run on a time based system. Our project will allow the user to have a finer control over the moisture levels in the soil by constantly polling the moisture levels in the soil samples and watering them on a moisture basis. This can be especially useful in experiments that require a high precision of control over moisture level, for example moisture levels vs plant growth.

- Sprinkler System - The sprinkler systems are usually set by the user to run at a set time every single day. They spray water out from a central opening in a circular manner. This design works well for most people, but wastes a lot of water because it sprays water only on the surface of the soil and a lot of it gets evaporated. Additionally, it does not take into account any outside factors such as rain or evaporation.
- Drip Irrigation System - This type of system uses a system of pipes to deliver water underground to plants at a continuous stream. This uses a lot less water than the sprinkler system and is in general more efficient than the sprinkler system, but again it does not take into account external factors.

Our product will be a variation of the drip irrigation system in which we can control the moisture levels more finely. The product will respond to the readings of the moisture in the soil through sensors, and will water the plants through the opening and closing of the valves of the water pipeline when the moisture of the soil drops below a certain level. This call-and-response reaction is how our products differs and will provide a distinct advantage over others, because it will water with more stability than the drop irrigation system and require less manual effort than the sprinkler system.

2 Proposed work

2.1 Evaluation of alternative solutions

- Autonomous Robot

One alternative solution is to build a robot that can walk around and detect which ports are dry. The robot will have wheels, which allow it to move around, and arms. One arm should be able to hold the sensor and insert it into the soil to get data. The other arm of the robot should be to spring that can spray water into the pot when detected dry.

Pros:

- With the robot, we do not need to turn the the holes on or off manually.
- There would be no wires in the greenhouses which prevents leaking or electric shock.
- Can be used in other things as well and it is really flexible, even if we put more plants in, we do not need to change a lot of set ups.

Cons:

- Robots can be difficult and expensive to build, so it may not be practical.
- The robots may damage the plants when it is moving (some plants have vines).
- We may have to set the exact distance between plants so the robot knows where to stop and test.

- Water from a Stone:

This implementation involves a stone-shaped glass container that will water plants on its own over the course of the next four days. Unlike autonomous "waterers" we have seen, there's nothing hi-tech about this. It just slowly rations the water inside the container so you only have to fill it once to keep your plants nourished while you're away. To use Water from a Stone, simply fill it up with water and set it down on your planter, making sure it is positioned so the release hole is not obstructed.

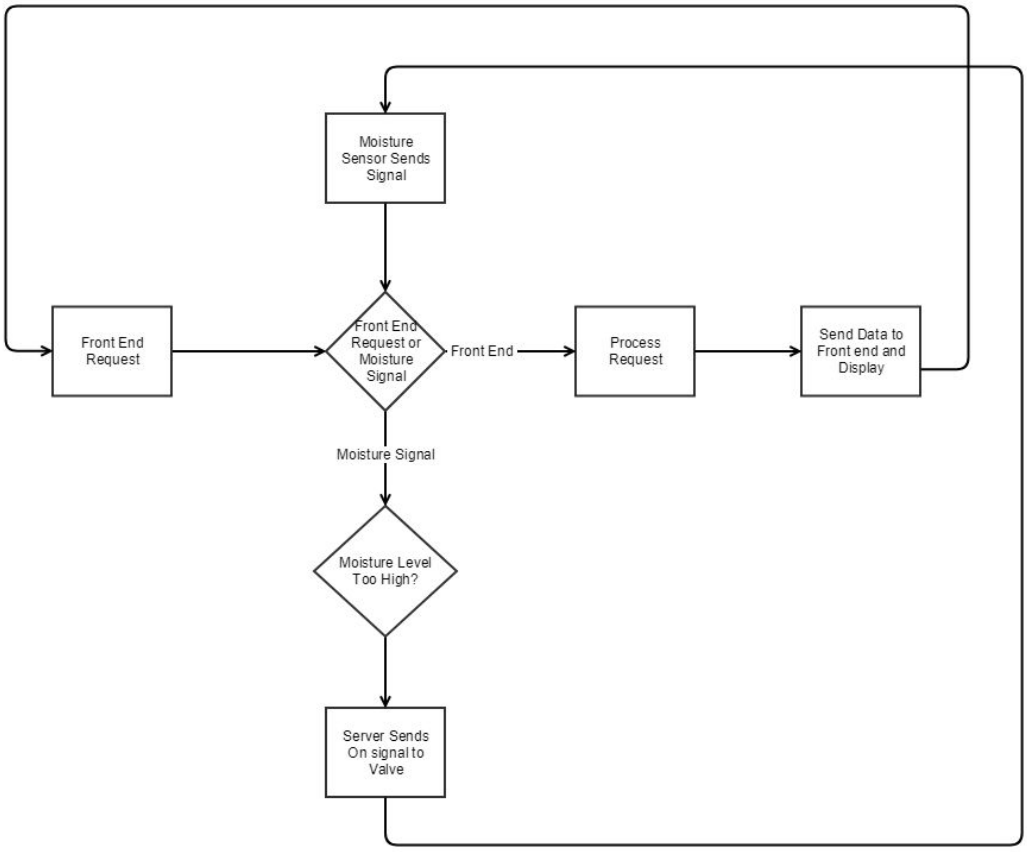
Pros:

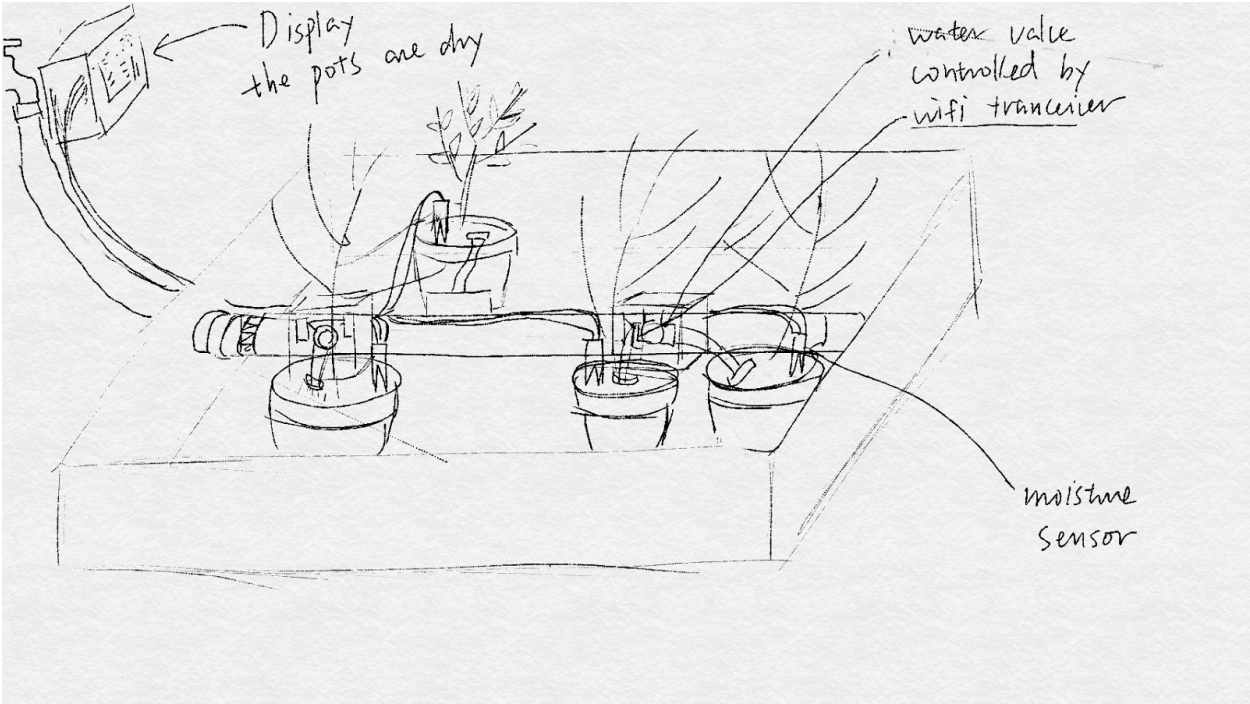
- Not as expensive as automatic devices
- Simple to install and utilize
- Easily replaceable when damaged.

Cons:

- Incorrect setup can lead to water either not dripping (your plants die of thirst) or flowing out too fast (they drown or something)
- Each implementation can only water the immediately surrounding area
- Can't be utilized in a large scale process

2.2 Design specifications





This product consists of 3 parts: the server, the user interface, and the hardware. The entire product will be wireless and we will use a server to communicate between the different layers. We chose this over a wired system because we intended for it to be used in a large distributed system so wiring may not always be possible.

The server is the control module for the entire system. It is in charge of processing requests from the front end interface and moisture sensors and it performs actions to handle those requests. The main methods that it will have is the algorithm to control water levels. It will poll the moisture levels every few minutes and output the water as needed in order to reach the desired moisture level. It will also expose a REST API for the front end to interface with. The main methods include the GET methods for displaying moisture levels and how much water it outputs. It will also expose some POST methods to modify the algorithm such as changing the moisture levels. More methods may be needed in the future.

The front end layer will serve as a way for the user to interface with the server without in depth knowledge of the code. This will allow the user to tune the algorithm parameters and view the performance metrics. This will most likely be through a web app so that the user can use it on any platform.

The hardware layer mainly consists of the sensors, water valves, and the microcontrollers for them. These components must be able to consistently send and receive requests from the server. Additionally, these sensors must be highly robust, easy to install, and reusable across different plant pots. The valves will control the piping system depending on the moisture needs of the plants. The piping system we will use will be similar to the ones used in the greenhouse at the agricultural department making the retrofitting of the valves easy. The entire device will be encased in a box, for easy transportation, but the box is unnecessary in larger systems.

Overall, we chose these different layers so that there is not a single point of failure for our system. Furthermore, it allows us to parallelize the different components and specialize in the different parts of the design.

2.3 Approach for design validation

With any product, there has to be a large amount of testing involved with it. With our product, we have many parts and thus extensive testing will be required to ensure that it performs up to standard. This part will most likely be the most time-intensive part of the project as stability and robustness are some of the primary goals. The tests that we plan on performing include performance testing, unit testing, and integration testing.

The first thing that we need to test for is the performance of our device. We will first need to figure out the proper location within the soil to place the moisture sensor so that it gets the most accurate readings. Additionally, we must decide on the most optimal place to secure the ends of the hoses. They must be close enough to the roots of the plant, but they can't be too close to the sensors otherwise the sensors could malfunction. This goes along with finding the best gates for our hoses and preventing any potential for water leaks. After figuring out the best placements for these materials, we must tune our algorithm for performance, stability, and scaling. For example, if our algorithm is used in a too large of a system, we can't poll for moisture levels constantly because it might overload the system. Additionally, the algorithm must evaluate how much water to output to each plant depending on the delta of the desired moisture level and the current moisture level. It also has to be precise enough that it does not overshoot the desired moisture levels because the device lacks the ability to remove water from the system.

Another thing that we have to do is unit testing the software component. This includes writing unit testing for the server functions such as sending and receiving signals, communication to the front end, and the logic behind the program. This part will be crucial in future proofing our system so that future developers can maintain or add more features to our code more easily. Additionally, we can do some front end testing through selenium if time permits.

The last and hardest thing we need to validate is the integration between all these different components. While each individual component can work on its own, the entire product may fail once we start "gluing" them together. There is no good way to test this other than doing it manually. One of us will have to watch the system perform and get the metrics, diagnose any problems, and patch any bugs that may appear.

3 Engineering standards

3.1 Project management

Kerry Zeng - Will be the team leader and be in charge of the technical reporting. He will also help in designing the software and the various means of communications between the front end and the server as well as the hardware and the server.

Curtis Green - Will be in charge of the interfacing with the hardware through software. He will also work on the server to send and receive signals from the hardware. He will also help write the algorithm to tune how much water is needed to reach certain moisture levels.

Jingyu Lu - Will be in charge of testing the hardware as well as the software. She will validate that everything is working correctly and provide performance metrics of the systems.

Hoang Trinh - Will be in charge of designing the hardware from the container it comes to the pipe system. He will also install the sensor and make sure that they are in the correct places for maximum performance.

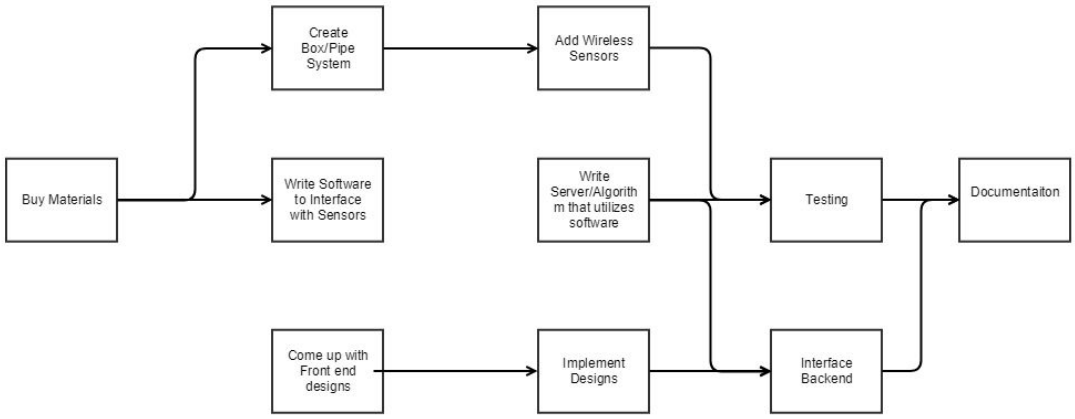
Hunter Lukken - Will be in charge of designing and implementing the user interface for the project. This will include a control page for the users to adjust the settings of the system and a metrics page for the user to monitor the performance of the system.

The team will meet during Friday labs and on Sunday to work on the project. Additional work will be done by individuals outside of those times on a per needs basis. The team will follow the AGILE methodology to keep the workflow moving consistently.

Table 1. Task matrix. Each task should have be assigned to exactly ONE person

		Team members				
Task	Description	Kerry	Curtis	Jingyu	Hoang	Hunter
1	Buy materials				X	
	Build container				X	
	Build pipe system				X	
	Add wireless solenoid valves			X		
	Add wireless moisture sensors			X		
	Write software for interfacing with valves			X		
	Write software for wireless moisture sensors		X			
	Validating Valves			X		
	Validating Wireless Sensors			X		
	Collecting Data of Water Output		X			
	Create server for accepting signals	X				
	Create server for sending signals		X			
	Assist in communication between frontend and server					X
	Write algorithm for turning the gate on and off based on moisture levels	X				
	Create Front End GUI for settings to configure settings					X
	Create Front End GUI to log activity					X
	Write documentation for maintenance	X				
	Writing assignments	X				

3.2 Schedule of tasks, Pert and Gantt charts



3.3 Economic analysis

The target market for this product would consist of the current agriculture production industry in any location, and would range from large scale production to production for experiments and self-reliant farmers

- This product has the potential to become fairly marketable because irrigation systems have been implemented on a large scale for a very long time. If our product proves to be efficient and affordable on the small scale, then this technology can be easily scale to large agricultural systems, which usually do not have an influx of technology, or much competition. Part of the reason for this is because the system they currently use works, and it would be risky to change this process at a large scale.

- We plan to construct our own system for our small demonstration. Hoses and pipes can be found from multiple vendors; however, the unique items we are using, such as the wireless sensors and wireless hose gates are available only at a limited quantity. This would make support for these certain items difficult. Maintenance that may be necessary includes upkeep of the pipes and hoses in case of water leaks and the functionality of the gates and sensors. Depending on the items we select, these possible issues may arise.
- Since our machine does not demand speed or quickness in its implementation, our component tolerances are solely limited to their functionality. We feel that the gates for the hoses/pipes should last a long time and would need to be replaced after years of wear and tear. The most vulnerable item we have to wear-and-tear is our wireless moisture sensors, which may wear out or stop sending wireless signals after a while.
- Regulations for general irrigation mostly revolve around preventing harm to the plants and preventing infection of public water sources. Since most systems have already been setup to abide by these regulations, we will not have to worry about compliance. We hope to test the system continuously to avoid water leaks, and expect the system to either work or not work based on the water being retained. This same process will help us test the accessibility of wireless signals sent by the moisture sensors and the reliability of the water hose gates.

3.4 Itemized budget

Item	Cost	Quantity	Total Cost
Wireless Moisture Sensors (5 pack)	$24	5	$109
Wood	$7	3	$21
Water Solenoid Valve	$11	3	$33
Pipes	$8	1	$8
voltage converter	$7	3	$21
wifi transceiver	$7	3	$21
Misc	$50	1	$50
Total			$199

4 References

Wood:

Supplies, B., Composites, L. and Boards, A. (2017). Shop (Common: 1-in x 4-in x 8-ft; Actual: 0.75-in x 3.5-in x 8-ft) Primed Pine Board at Lowes.com. [online] Lowes.com. Available at: https://www.lowes.com/pd/Common-1-in-x-4-in-x-8-ft-Actual-0-75-in-x-3-5-in-x-8-ft-Primed-Pine-Board/50004800 [Accessed 8 Oct. 2017].

Pipes:

Fittings, P., Fittings, P. and Pipe, P. (2017). Shop Charlotte Pipe 2-in x 5-ft 280 Schedule 40 PVC Pipe at Lowes.com. [online] Lowes.com. Available at: https://www.lowes.com/pd/Charlotte-Pipe-2-in-x-5-ft-280-Schedule-40-PVC-Pipe/3133043 [Accessed 8 Oct. 2017].

Wireless Moisture Sensor:

Amazon.com. (2017) [online] Available at: :https://www.amazon.com/Springfield-91746-Digital-Moisture-Freeze/dp/B0037BNHLS [Accessed 8 Oct. 2017].

Backup Wireless Moisture Sensor:

Sensor, W. (2017). Wireless Water/Moisture Sensor | Wireless Sensor Tags. [online] Store.wirelesstag.net. Available at: https://store.wirelesstag.net/products/wireless-water-moisture-sensor-2-0?variant=20055138884 [Accessed 8 Oct. 2017].

water solenoid valve:

Amazon.com. (2017). [online] Available at: https://www.amazon.com/dp/B00KKIH1YK/ref=nav_timeline_asin?_encoding=UTF8&psc=1 [Accessed 8 Oct. 2017].

wifi transceiver:

Amazon.com. (2017). [online] Available at: https://www.amazon.com/dp/B00VWVDOFO/ref=nav_timeline_asin?_encoding=UTF8&psc=1 [Accessed 8 Oct. 2017].

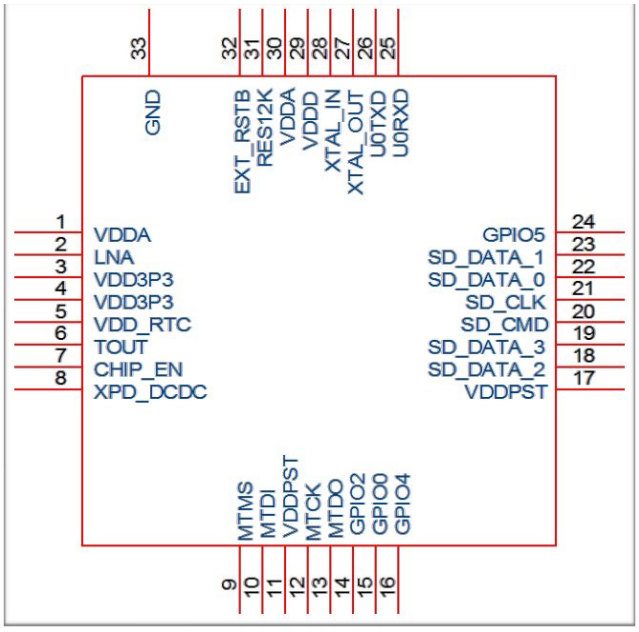
voltage converter:

Amazon.com. (2017). [online] Available at: https://www.amazon.com/dp/B00GBUSLTA/ref=asc_df_B00GBUSLTA5206424/?tag=hyprod-20&creative=395033&creativeASIN=B00GBUSLTA&linkCode=df0&hvadid=193992629021&hvpos=1o8&hvnetw=g&hvrand=9096677220414562646&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9027906&hvtargid=pla-312676080475 [Accessed 8 Oct. 2017].

5 Appendices

5.1 Product datasheets (optional)

wifi transceiver:



This part can receive wifi signals from our device and send control our valves through the GPIO ports.

5.2 Bio-sketch (one paragraph)

Include a brief bio-sketch that describes your strength (software or hardware) and project related experiences.

Kerry Zeng- I have experience in developing web applications. I specialize in the back end writing REST API's, RPC, and unit testing.

Curtis Green- Some experience scripting for game mods in C#/++, Python, Ruby, and Lua; as well as general development of through C/++ and javascript.

Jingyu Lu- working experiences in the greenhouses and had experience building auto robot car.

Hoang Trinh- I have experienced data analysis using C++ and Java. I specialize in analyzing and giving algorithms to problems.

Hunter Lukken- I specialize in software, and have worked on multiple web application projects using different frameworks including Java, Angular, C#, and more. I have also coded multiple school projects in C++. I have experience with multiple database tools and data analysis as well.