1. Title of your project

SAAGE: Sensing and Actuation in Agriculture and Gardening Environments

2. Team members names, email addresses, and student IDs

Daniel Koohmarey dkoohmar@ucsd.edu A53274145
Nitish Nagesh nitish.n0212@gmail.com U08450730

3. The motivation for your project

In agriculture and gardening, there is often one main metric of success: crop yield. How many tomatoes will I have to harvest? At the agricultural level this dictates the success of a farmers business, who needs his/her crop yield to be high for a good harvest to pay back the debt taken on to plant the crop initially. At the gardening level, gardeners often dedicate hours of energy into caring for and raising their plants. They check up on their crops and work towards the healthy growth of the plant. In both contexts, crops must be monitored to determine health and corrective actions, which can help lead to a high crop yield. We aim to automate the previously manual and iterative process of monitoring with an on-site sensing system that can gather useful statistics related to crop growth. These statistics include the soil temperature, soil moisture, air temperature, air humidity, air pressure, and UV light (sunlight estimation). In addition, once data is gathered, linear regression techniques allow us to make predictive estimates as to the conditions the crop face, allowing for pre-emptive action to be taken in the event of unfavorable conditions. Examples of useful preemptive actions involve ensuring plants are receiving enough water and changing the irrigation system after determining dry periods of a day the soil moisture levels, or changing irrigation techniques if the soil temperature indicates it is freezing over. Given a farmer or gardener has the data related to the successful growth of their crop, one issue that still persists are rodents that may consume the crop. To combat these rodents, our system employs three actuated defensive systems triggered when motion is detected by a PIR sensor. A hypersonic emitter emits high frequency noise to repel small pests, while a speaker plays an audible noise of a predator, and a small servo motor moves a makeshift scarecrow while a large servo motor spins a scarecrow owl.

4. Related work

a. Discuss what others have already accomplished in this area - academic papers or commercial products are okay. Include specific citations.

We split the capabilities of our system into two main components, and compare them to commercial products on the market today (which we identify from Amazon):

Agricultural/Gardening Sensing:

Soil Moisture Monitors exist:

1. XLUX T10 Soil Moisture Sensor Meter (over 3000 ratings)
2. Diwenhouse Soil Moisture Meter Test Kit

A similar sensing system with multiple metrics (temperature, sunlight, moisture) exists with the Xiami smart plant tracker but requires a smartphone app and only acts as a passive monitor.

Other sensors often are only indoor such as the Govee Temperature Humidity Monitor.

Rodent Deterrence:

Various products rely on ultrasonic sound for rodent deterrence, found on amazon:

1. Neatmaster Dual Microchip Ultrasonic Pest Repeller (over 2000 ratings)
2. Seewinland Ultrasonic Pest Repeller
3. Bocianelli Ultrasonic Pest Repeller

Passive products such as scarecrows are also a popular option:

4. Gardeneer by Natural Enemy Scarecrow Horned Owl (over 2500 ratings)

b. Highlight what is interesting in your particular implementation relative to other's previous work

While sensing systems exist, they are mainly passive in nature. We aim to combine both passive and active elements that benefit crop production in an "all in one" system. Additionally our linear regression for prediction is not used in competing sensing systems and allows for preemptive fixes to poor growth conditions. Another limitation in existing systems is that they are designed to be indoors only, which we hope to address with a waterproof casing.

From the rodent deterrence perspective, systems are focused on indoor use cases, we plan on using the same underlying deterrence mechanism to deter outdoor rodents. We found the use of predatory bird such as an owl as a scarecrow was very popular on Amazon with over 2000 positive ratings. The reviews indicated this was an effective system for scaring off certain elements. A variation of the bird that added movement was equally popular. Once again, we would integrate these existing defenses along with the sensor data into one system designed for managing and protecting crop production.

5. List hardware components

a. Include manufacturer and model of any parts you use

We used the following hardware:

Sensor	Sensor Model	Manufacturer	Link
Temp/humidity/pressure sensor	Adafruit BME280 I2C or SPI Temperature Humidity Pressure Sensor	Adafruit	https://www.adafruit.com/product/2652
Waterproof temp sensor	Waterproof DS18B20 Digital temperature sensor	Maxim	https://www.adafruit.com/product/381
UV/Visible light sensor	SI1145 Digital UV Index / IR / Visible Light Sensor	Adafruit	https://www.adafruit.com/product/1777
Soil Moisture Sensor	Adafruit STEMMA Soil Sensor - I2C	Adafruit	https://www.adafruit.com/product/4026

	Capacitive Moisture Sensor		
Infrared motion sensor	PIR (motion) sensor	Adafruit	https://www.adafruit.com/product/189
Ultrasonic emitter	HC-SR04 Ultrasonic Sonar Distance Sensor	Adafruit	https://www.adafruit.com/product/4007
Large servo	Standard servo - TowerPro SG-5010	TowerPro	https://www.adafruit.com/product/155
Small servo	Continuous Rotation Micro Servo - FS90R	FEETECH RC Model Co	https://www.adafruit.com/product/2442

b. Discuss HW design choices

Hardware choices were largely financially driven, deciding to use the same online retailer to avoid shipping costs. We decided to use Adafruit as a popular retailer of components with documentation and examples available for a majority of components they sold, as well as a large component offering.

Due to the agricultural/gardening aspect of our project, our hardware sensors all retrieved data we thought relevant to the space. We collected the following information using the sensors:

Component	Communication Protocol	Units sensed
Waterproof temperature Sensor	1-wire	Temperature in ℉
Soil Moisture Sensor	I2C	Moisture level 200(very dry) to 2000 (very wet)
PIR sensor	GPIO	Motion detection
Temp/humidity/pressure sensor	I2C	Temperature in ℉, Humidity in %, Pressure in hPa
UV/Visible light sensor	I2C	UV index
Ultrasonic sensor	GPIO	Emits ultrasound in 40 kHz range
Small servo motor	GPIO	360° rotation

Big servo motor	GPIO	Intermittent rotation

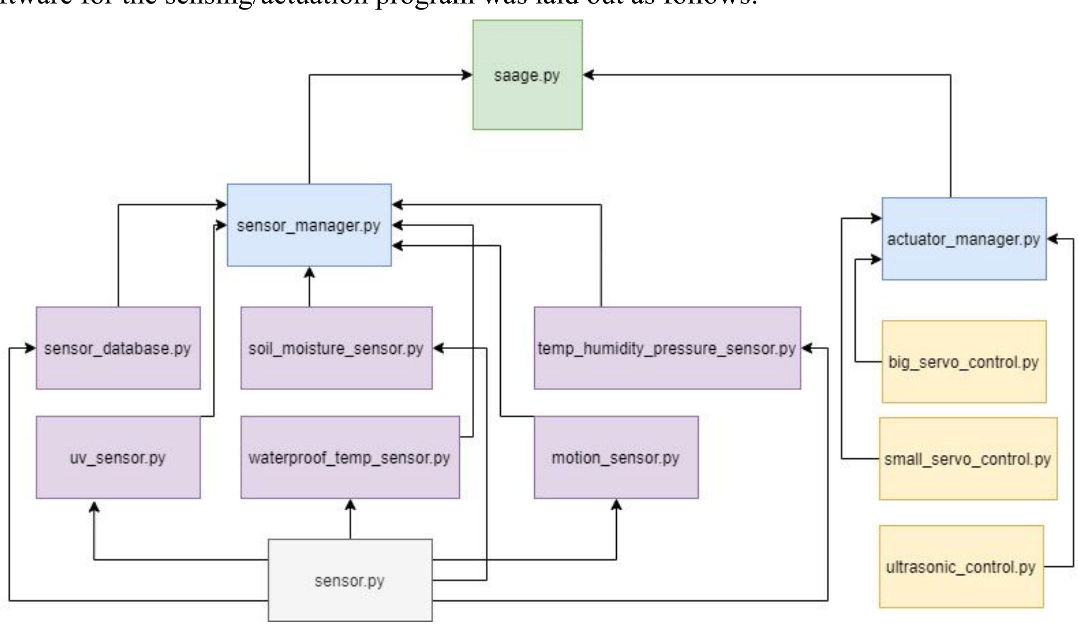
6. List software components and frameworks
a. Include release version numbers, etc

Language: python3, Web javascript libraries: Google Charts API

Python Module	Version	Use
sqlite3	latest	Sqlite3 database interface for data storage
busio/board	latest	i2c interface access
adafruit_seesaw	latest	Soil moisture sensor interface
adafruit_bme280	latest	Temp/humidity/pressure sensor interface
SI1145	latest	UV sensor interface
RPi.GPIO	latest	GPIO interface, soft PWM for servos
flask	latest	Web interface
sklearn	latest	Linear regression prediction model

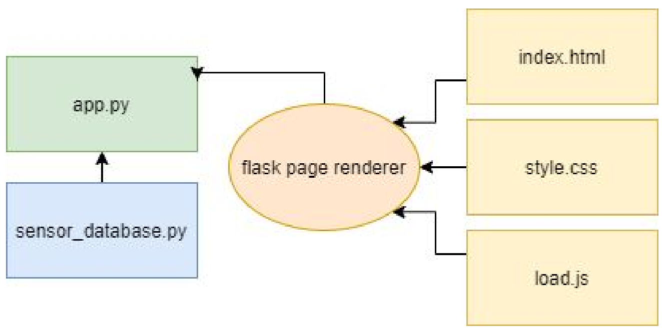
b. Discuss SW design choices

Software for the sensing/actuation program was laid out as follows:



The main program run was saage.py, which instantiated both the sensor manager (SensorManager from sensor_manager.py) and actuation manager (ActuatorManager from actuator_manager.py) which controlled their respective hardware. The actuation manager spun the servos and emitted ultrasonic sounds as a result of a callback from the motion sensor. The sensor manager would periodically collect data from all sensor objects it tracked, and a sensor object was created for each sensor type we wanted data from. Each specific sensor class had a get_data function which returned a dictionary containing key/value pairs of data. The keys returned in this dictionary matched the column names in the Sqlite3 table the data was stored in. The sensor manager would periodically collect all of the sensor data and write it to a local database file using the sensor_database.py SensorDatabase. The decision was made to store the data locally rather than on the cloud to keep the system “all in one” with no external dependencies (i.e server is running somewhere to accept the data). As a result of storing the data locally on the pi, the web server which visually presented the data was also locally running on the pi. This also improved reliability and security by guaranteeing access to the data locally. Performance concerns about running the server on the pi were low as responsiveness to a single user accessing the data was high. The majority of computation occured also in the client’s browser to render the charts, which would be on a stronger processing system.

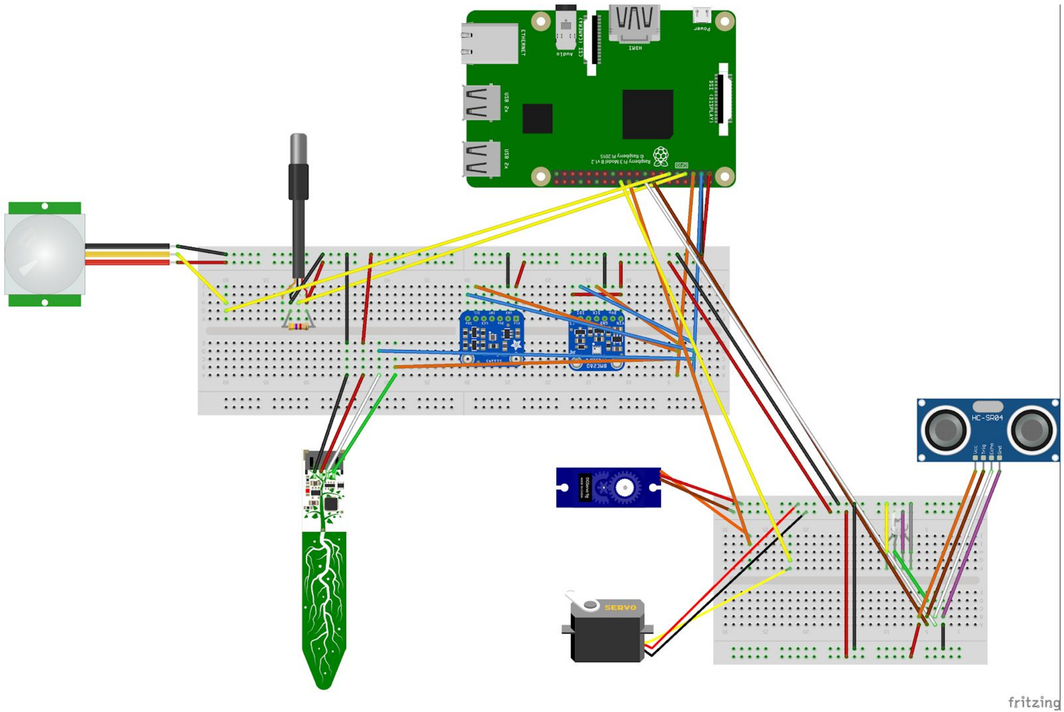
Software for the web server was laid out as follows:



The software followed standard flask development, with the use of a rendered template to insert python variables into html/js code.

7. Describe how you integrated HW and SW, discuss any issues and challenges you faced and how you solved them

To integrate, we first identified all of the connection requirements for the hardware to ensure there were no conflicts. This included ensuring unique addresses for i2c devices and pin availability for all devices. To track pin usage and manage configuration of the system when working disjointly as the system was built up, we created a fritzing diagram:



We developed and integrated each sensor one at a time and ensured the data was retrieved correctly before adding an additional complexity to reduce the number of unknowns in the system at a time. We encountered slight issues with the servos, because the code we initially needed relied on physical PWM but the raspberry pi was not capable of generating this signal. We quickly found the same could be achieved with a “soft” PWM in software. Identifying the correct duty cycle and controlling the servos as desired also required manual adjustments of the duty cycle and duration of movement. The sensors that came with vendor software support were easier to integrate as some complexity was abstracted. We also added an LED to our design to determine if the ultrasonic emitter was working, as it would be inaudible and we would not be able to guarantee it was working otherwise. We also faced an issue when we could not integrate the soil moisture sensor due to a missing JST PH 4-Pin to Female Socket Cable - I2C STEMMA Cable - 200mm we ordered after realizing we needed one. An additional hardware issue we encountered was that the uv sensor was obstructed from light if placed in the waterproof case. We tried to address this issue by cutting a hole in our waterproof box and covering it with clear plastic.

8. Experimental section

a. Describe your experiment setup and metrics of success (e.g. response delays, target accuracy, energy efficiency)

The experimental setup involved first testing each sensor value one at a time, and then testing collection from the system as a whole. Finally an entire end to end test was conducted to ensure the system as a whole was integrated correctly.

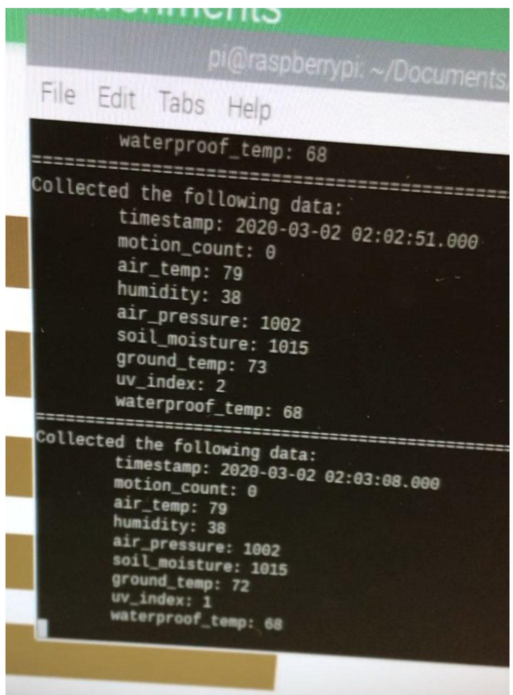
b. Outline experiments you already did, and discuss results to date

1. To establish the sensed data from the sensors was accurate, we evaluate each sensor as follows:
    a. Temp/humidity/pressure sensor: We check the reported values against values from the weather channel and ensure no significant deviation in values.

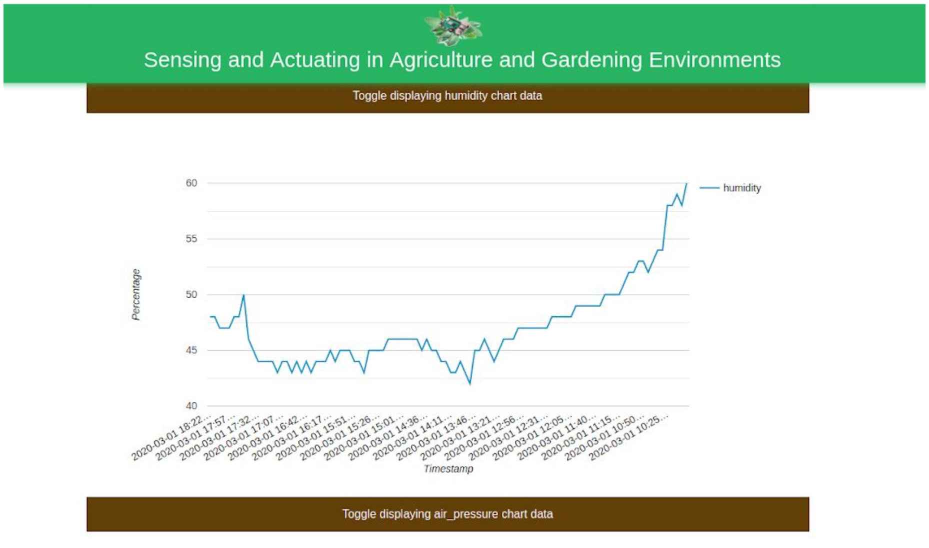
   b. Waterproof temp sensor: We check the reported value when the sensor is not contacting anything, and then put the sensors in our hands and expect the temperature to rise.
   c. UV/Visible light sensor: We evaluate the uv index when the sensor is exposed to sunlight, and then cover the sensor and ensure the uv index decreases when the sensor is covered.
   d. Soil moisture sensor: Place sensor in dry soil, place sensor in wet soil. Ensure different moisture level readings.
   e. To verify the web page is working, we check the chart values against what was collected from the sensor manager.
2. To determine if the predicted sensor values are accurate, ensure the predicted values lie within the range of sensed values fed into the linear prediction model.
3. To ensure the defensive mechanisms are triggered on motion, we will trigger the PIR sensor and ensure the actuators all trigger set periods of time in response. We will use an LED to verify the ultrasonic emitter is emitting.

b. Present results

From the individual sensor testing with a potted plant, we determined from the values gathered that all our system sensors were functioning as intended.



We then verified we could observe the recorded data from our web interface:



And also verified the predicted value was within the range of the sensed values:

We did observe inconsistency in the PIR motion sensor triggering motion events and actuation, as well as inconsistency in the big servo spinning. We also faced issues keeping the scarecrow on the big servo balanced

c. Discuss the implications of your results

The varying temperature in our gathered sensor data was somewhat expected, but indicated a limitation of the system because the air temperature was more reflecting the box temperature.

Our results show it is possible to have an integrated “all in one” system for a gardening or agriculture environment that does both sensing of growing conditions and active pest deterrence. In addition, it is possible to predict near term sensor data from calculated sensor values. The poor reliability of the PIR motion sensor also indicated to us a different sensor would be better for motion detection in the future.

9. Conclusion

a. What did you accomplish? How does this demonstrate key concepts in this class or further the development of embedded systems

We accomplished an integrated sensing/actuating system that runs on real world data. This demonstrates the key concepts in the class of the usefulness of ADCs, which allow us to obtain sensed data programmatically from physical values in the real world. We also fed our sensed data into a linear regression model to predict future values. This reflects the growing trend in memory and industry to support performance of computing vector operations at great speed, and having a processor with dedicated hardware to accelerate vector arithmetic workloads. We also learned of the usefulness of GPIOs in interfacing with a variety of sensors.

b. Discuss any unfinished parts of the implementations and/or next steps if you were to continue working on the project.

The negative results and unfinished implementations in our system are as follows;

- Solenoid valve for fertilizer: We didn't realize the solenoid valve required a 12V power source and didn't want a second power cord for the system, so decided to replace the solenoid valve with servo motors.
- Constructing a fertilizer system with just servos proved challenging to design mechanically, so we opted for an additional scarecrow for the sake of time. With more time we would look into using a screw like design to dispense the fertilizer.
- The UV sensor needed to be waterproof but also observe the sun, so we cut hole in the waterproof box but notice it still reports low uv index values due to the large amount of shade in the box.
- Google charts offline was not allowed/supported by the Google ToS, which prevented us from setting the pi up as a local access point to access the data directly. If we had more time, we could have changed to an offline graphing option, which would have allowed us to set up the raspberry pi as an access point with the web server providing all the information to a connected user without internet.
- Air temperature reading due to the waterproof box is warmer than normal from the electronic heat, this could be addressed if it was waterproof/external.

If we were to continue working on the project, we would consider adding support to upload the sensed data to the cloud, and adding offline mapping library so that the Raspberry Pi could be set up as an access point. We would also swap the PIR motion sensor for something more reliable.