Design Document:

Developing a Deep Learning Model to Automatically Detect Microscale Objects in Images and Videos

Senior Design Group: sddec23-10

Ethan Baranowski, Chris Cannon, Matthew Kim, Katie Moretina

Client/Faculty Advisor: Dr. Santosh Pandey

1 Introduction

1.1 Acknowledgement

The team would like to acknowledge the following people and groups for their contribution to our project:

- Dr. Santosh Pandey
  - For his guidance, leadership, and efforts facilitating our understanding of the problem and possible solutions.
- Yunsoo Park (PhD Student)
  - For his help in understanding existing object detection algorithms, identifying cysts in images, and sharing his general knowledge of the problem with us.
- Dr. Greg Tylka (Morrill Professor, Nematologist, Director of the Iowa Soybean Research Center)
  - For his detailed explanation of existing processes and methods and his tour of the lab where that happens on Iowa State's campus.
- Facebook's Research team
  - For their implementation of Mask R-CNN in their open source Detectron2 project, which will likely serve as a strong foundation for our project.

1.2 Problem and Project Statement

Every year 15-30% of soybeans are infected with parasitic cyst nematodes *[1]*, limiting their yield and forcing farmers to use pesticides. Farmers are currently faced with the issue of expensive and slow lab analysis that provide insight into how many cysts are on soybean plants. These lab techniques cost the farmers valuable time as they must wait for the results to apply the fertilizer necessary for the continued health of their soybean crops.

Our goal is to make a product that can detect and count microscale objects, which is cysts in our case. In computer vision, there are different algorithms that can detect objects. In similar fields, for example, there are some algorithms that can detect cars or people in images *[2]*. Therefore we want to implement an object detection algorithm designed for small objects to determine how many cysts are on the roots of soybean plants. We will also create a device to integrate image capturing with the machine learning algorithm, therefore, increasing productivity on the farm. Detecting cysts is the first step in eliminating them. Geneticists will be able to run tests to find if certain species of soybeans are more resistant to cysts, and farmers will know the amount of pesticides to use. This process will increase soybean productivity and decrease the overuse of pesticides.

1.3 Operational Environment

In our case, to maximize the advantages of not damaging the plant's root, we are considering a portable hardware solution intended for outdoor use, specifically on farm fields by the researcher, geneticists and farmers. Since our products are allowed to be used outside of the labs, the product must be designed to function without access to Wi-Fi or electronic plugs for charging. It should be able to handle various weather conditions, such as sunny, cloudy, or windy days. Additionally, temperature fluctuations should be taken into account, though they will not be extreme. The target environment involves soybean cultivation, which typically occurs between 68°F and 86°F.

1.4 Requirements

Requirement Type	Details
Functional Requirements	● Software(Algorithm) ○ Functional without internet access ■ Independent of software upgrades (no server) ○ Better than 50% accuracy for detecting cysts in images **(Constraint)** ○ Less than 5 second of processing time per image **(Constraint)** ○ Less that 1GB space taken by the application **(Constraint)** ○ Image processing independent of MATLAB ● Hardware (Image Capture) ○ High-enough image resolution to use for object detection ○ Portable (Wireless, can be used on a field) ○ Does not damage plants
Resource Requirements	● Processors/Processing time to train the algorithm ● Materials to build a prototype device ○ Cameras ○ Motors (depending on prototype design) ○ Misc building materials (aluminum, plexiglass, screws, wires, etc) ● Plants of certain age (3 week to 5 week maturity)
Qualitative Aesthetics Requirements	● Keep the device as small as possible ● Easy to transport

Economic/Market Requirements	• Create a final device that is affordable for farmers ○ Upper limit around $500 **(Constraint)**
Environmental Requirements	• Must not damage plants in any way • Provide an accurate number of cysts to avoid an overuse of pesticide consumption
UI/UX Requirements	• Farmer should not interact with product during image capture process • No formal training required to use the device

Table 1.4-1 Project Requirements

1.5 Intended Users and Uses

WHO has the problem?

Primary users: Industries and crop researchers.

Secondary users: Farmers.

WHAT is the problem?

Expensive analysis required to gain measurable insight into how many parasitic cysts are on soybean plants. Currently in the U.S, cysts nematodes are affecting about 15~30% of yields of soybean every year. That causes about $1 billion losses annually *[1]*.

Also, there aren't any methods that analyze the number of cysts directly from the root. Other methods require using technical equipment to crush the Cysts and analyze using the microscopic tools.

WHERE is the problem occurring?

The cysts attach to the roots of soybeans underground. In the United States, soybean production is most common in Iowa, Illinois, Nebraska, Minnesota, and Indiana. Also in the research lab, expensive tools are used with technical steps in order to count cysts.

WHEN is the problem occurring?

Before harvesting, during the growth of the plants.

WHY is it important?

Crop researchers can make more educated decisions about while developing cyst-resistant plants. Farmers can make more educated decisions about the proper care of their crops.

HOW will it be solved?

Machine Learning algorithms have made several developments that are tailored to identifying small objects in images. These improvements give a strong indication that a prototype device can be trained to recognize the cysts and provide accurate measurements to farmers.

Use Cases:

1) Uses for farmers
2) Uses for genetecists/plant pathologists
3) Uses for continuing developers/project maintainers

Figure 1.5-1 below shows a use case diagram for our final product:

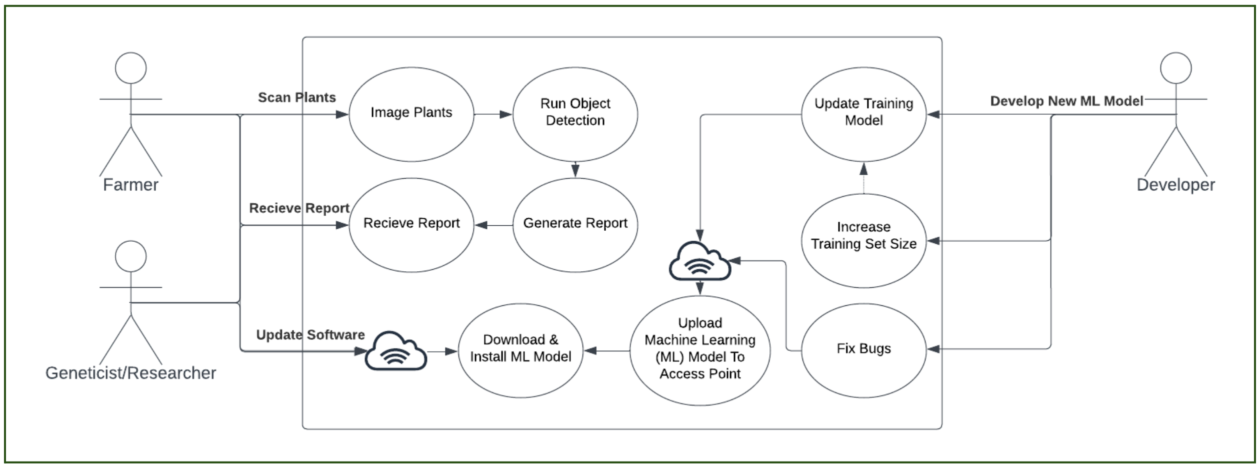


Figure 1.5-1: User Case diagrams, interaction between Researcher, Farmer and Developer

1.6 Assumptions and Limitations

Assumption	Justification

Only 2 pictures per plant will provide enough data for the counting algorithm.	Scanning both sides of the plant should capture every cyst in at least one image.
User will not have to interact with the product during its image capturing and counting algorithm phases.	Product will be mostly autonomous, with some user interaction for setting it up and running it.
Blue is the best background color.	Blue is the most direct contrast to the colorization of the cysts.
Python counting algorithm will be runnable on a Raspberry Pi.	
Raspberry Pi can interface with and control the LCD screen, the camera, and the motor.	Adapters and circuit board (may not be required) connections are available for each product.
Product can be used in the field.	Product is easily transportable and internet free.
Product will be used for soybean parasitic cyst counting only.	Image capture process is trained for parasites unique to soybean plants.

Table 1.5-1 Assumptions and Justifications

Limitation	Justification
1 plant processed at a time.	Counting process is limited to one plant with associated results. Multiple plants would derefence the results.
Cost to produce shall not exceed $500.	
Dimensions will be 12”x15”	Size of the base
Counting algorithm will exceed 50% accuracy but not exceed 80% accuracy.	Faster R-CNN algorithm has repeatedly shown high performance for small object detection, but the research has not indicated over 80% accuracy.
Battery life of 2 hours.	

Table 1.5-2 Limitations and Justifications

1.7 Expected End Product and Deliverables

The primary end goal for the Cyst Detector project is to develop and implement machine learning algorithms that can accurately detect small-scale objects, specifically cysts. The software will be designed to identify cysts within plant roots without causing any damage or requiring grinding of the roots. This cutting-edge algorithm will lay the foundation for future hardware integration, enabling the creation of a scanner that utilizes the algorithm to count the number of cysts present. Delivery date: Fall of 2023.

As an extension of the project, we will develop a hardware solution that acts as a scanner to utilize the machine learning algorithm for cyst detection and counting. The hardware will be designed with simplicity, cost-effectiveness, and automation in mind, allowing users to focus on other tasks while the device processes the count. Furthermore, the hardware will be suitable for outdoor use, portable, and able to operate without an internet connection.

A user-friendly software interface will be developed to facilitate seamless interaction between the user and the cyst detection system. This interface will allow users to scan plant roots with the hardware device and obtain cyst count results generated by the machine learning algorithm. The software will be designed to work efficiently both with and without an internet connection, ensuring flexibility and reliability in various environments.

Comprehensive documentation will be provided for both the cyst detection machine learning algorithm and the hardware scanner which clients can update and maintain afterwards as well. Documentation will include guidelines for installation, setup, and operation, along with troubleshooting tips and maintenance instructions. These resources will ensure that users can effectively deploy and manage the Cyst Detector system in their operations.

2. Specifications and Analysis

2.1 Proposed Approach

Our task is to develop a deep learning model that can find and count the number of specific microscale objects in the image. To address the problem, we have researched various object detection algorithms, including YOLO, Faster R-CNN, and Single Shot Detection. We have compared these algorithms to determine the best fit for our project. Additionally, we have explored labeling tools and started working on the labeling process. We have also identified existing algorithm implmentations that we may use as part of our project. Based on our algorithm implementation, we will develop hardware tools that are portable, which researchers can utilize outside of their laboratories.

Our project adheres to or is relevant to the following standards:

IEEE 268-1992

American National Standard for Metric Practice

Guidance for the application of the modernized metric system in the United States is given. Known as the International System of Units (SI), the system is intended as a basis for worldwide standardization of measurement units. Information is included on SI, a limited list of non-SI units recognized for use with SI units, and a list of conversion factors from non-SI to SI units, together with general guidance on proper style and usage. [4]

IEEE/ISO/IEC 32675-2021

- ISO/IEC/IEEE International Standard--Information technology--DevOps--Building reliable and secure systems including application build, package and deployment

Technical principles and processes to build, package, and deploy systems and applications in a reliable and secure way are specified. Establishing effective compliance and information technology (IT) controls is the focus. DevOps principles presented include mission first, customer focus, left-shift, continuous everything, and systems thinking. How stakeholders, including developers and operations staff, can collaborate and communicate effectively is described. The process outcomes and activities herein are aligned with the process model specified in ISO/IEC/IEEE 12207:2017 and ISO/IEC/IEEE 15288:2015. [5]

IEEE/ISO/IEC P24748-6

ISO/IEC/IEEE Draft Standard - Systems and Software Engineering -- Life Cycle Management

This standard:

- Provides requirements and guidance for use of the integration process and its relationships to other system and software life cycle processes as described in ISO/IEC/IEEE 15288:2015 and ISO/IEC/IEEE 12207:2017,
- Specifies information items to be produced as a result of using the integration process, including the content and format of the information items.

This standard provides a detailed presentation of system and software integration, considering:

- The related concepts of integration, such as interface, verification, and validation;

- The possible composition of a system comprised of any mix of products and services that can include hardware, software, humans, data, processes (e.g. review process), procedures (e.g. operator instructions), facilities and naturally occurring entities (e.g. water, organisms, minerals),
- The life cycle stages of a system at which integration may occur, and
- The context of the domain in which the system functions. [6]

IEEE/ISO/IEC 14764-2021

ISO/IEC/IEEE International Standard - Software engineering - Software life cycle processes - Maintenance

This International Standard establishes a common framework for software life cycle processes, with well defined terminology, that can be referenced by the software industry. It contains processes, activities, and tasks that are to be applied during the acquisition of a software system, product or service and during the supply, development, operation, maintenance and disposal of software products. [7]

2.2 Design Analysis

We have only recently entered the implementation phase of our project. With regards to our software plan, we researched the best algorithms for small object detection. Faster R-CNN was selected as our algorithm of choice. We have found an implementation of the algorithm that was successfully run on the Google Colab virtual machine. However, the results are not yet repeatable on a local machine because the environment Google Colab uses is not installable on local machines.

We shifted course to finish labeling our dataset for the training process. The implementation in Google Colab supports training the algorithm on a custom dataset. This is what we intend to do next as a proof of concept that our algorithm is trainable and implementable before modifying it to run locally. This follows the generic Computer Science tenet that functionality comes first; optimization comes after. An additional reason for pursuing this solution instead of directly resolving the issue is that Google Colab supports downloading the code as a Python file which can be run locally using terminal commands. As a result, the issues may be resolved with running Python files locally instead of through Google Colab. Thus, the solution to our problem will be easier to reach if we have a working algorithm to test the solutions.

With our hardware, we have only finished our systems-level design. This is because we required more research and discussion about the machine learning model to have an understanding of what hardware we would need. The mechanical components take the lowest priority as the mechanical design may change if we discover some challenges that requires changing the mechanical design. So, this approach avoids investing time into designs that may be changed not as a result of problematic design but new discoveries with the software and requirements to run the algorithm.

2.3 Development Process

Our project will be completed following the Agile methodology. This method will benefit us because of its iterative nature. Since the work we're doing is research and experimental development, we can assign tasks to be completed in weekly sprints, ensuring that we have deliverables each week that we can iterate on and improve, while also allowing us to make steady progress on some of the longer goals, like labelling our data.

Agile is a better fit than Waterfall because our project consists of several distinct components that do not depend on each other's completion to be started. We can work on the prototype, raspberry pi, software, datasets, and ML model simultaneously if necessary.

2.4 Conceptual Sketch

Our intended design is shown in the sketch below (Figure 2.4-1). It is a 3-D representation of the prototype we intend to develop. The description is broken down as:

Front: handlebar for user to carry the product.

Side: LCD screen for displaying parasitic cyst count. Rotating attachment point for arms. The rotating attachment point enables the user to fold the arms down for transportability. Additionally, the rotating attachment contains a hole through which the wires connecting the Raspberry Pi underneath the platform can be fed to the rest of the components.

Top: On the far end, there is a motor box which may be geared as necessary to produce enough torque to rotate the plant 180 degrees, flipping the plant. The motor uses a binder clip-like clamp. Binder clips have highly desirable properties that would make flipping the plant easy. These properties are a high pressure clamp which requires applied force to release. With this type of clamp, we will have full control over the orientation of the plant during operation with very little risk of losing our clamp grip during normal operation.

Bottom: The bottom is not portrayed in the figure. However, the bottom will be partially hollow exposing the Raspberry Pi (while keeping it off the ground) for the user to connect to later for data collection/management and charging the power source.

Note: The plant is portrayed with its stem in the clamp and the roots clearly unobstructed from the overhead camera.

Arms: The arms are PVC arms attached to the sides as well as attached to each other via a cross bar piece of PVC. PVC is an ideal material for this because PVC can be machined easily while having secure joint connections. This enables us to attach a camera mount to the underneath side of the crossbar in the ideal overhead position for image capture of the plants.

Blow-up Circle: The Blow-Circle in the figure illustrates that the Raspberry Pi can be wired to the rest of its components by sending the wires through the rotatable attachment point of the arms.

Finally, Figure 2.4-2 illustrates the block-diagram connections and interactions between the different components. The blue background illustrates mechanical components, while the green background illustrates electrical components.

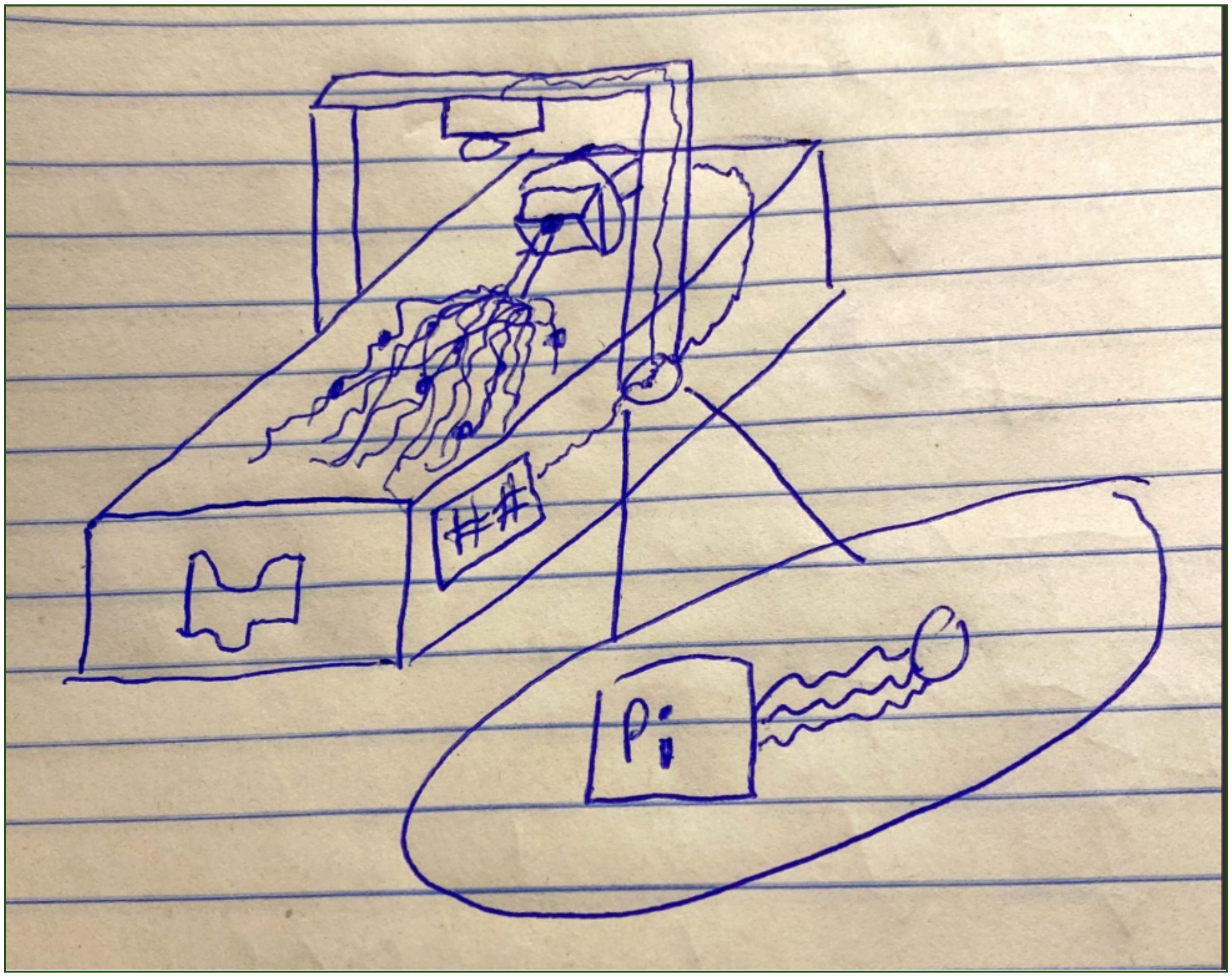


Figure 2.4-1: Conceptual Sketch of Soybean Root Scanner

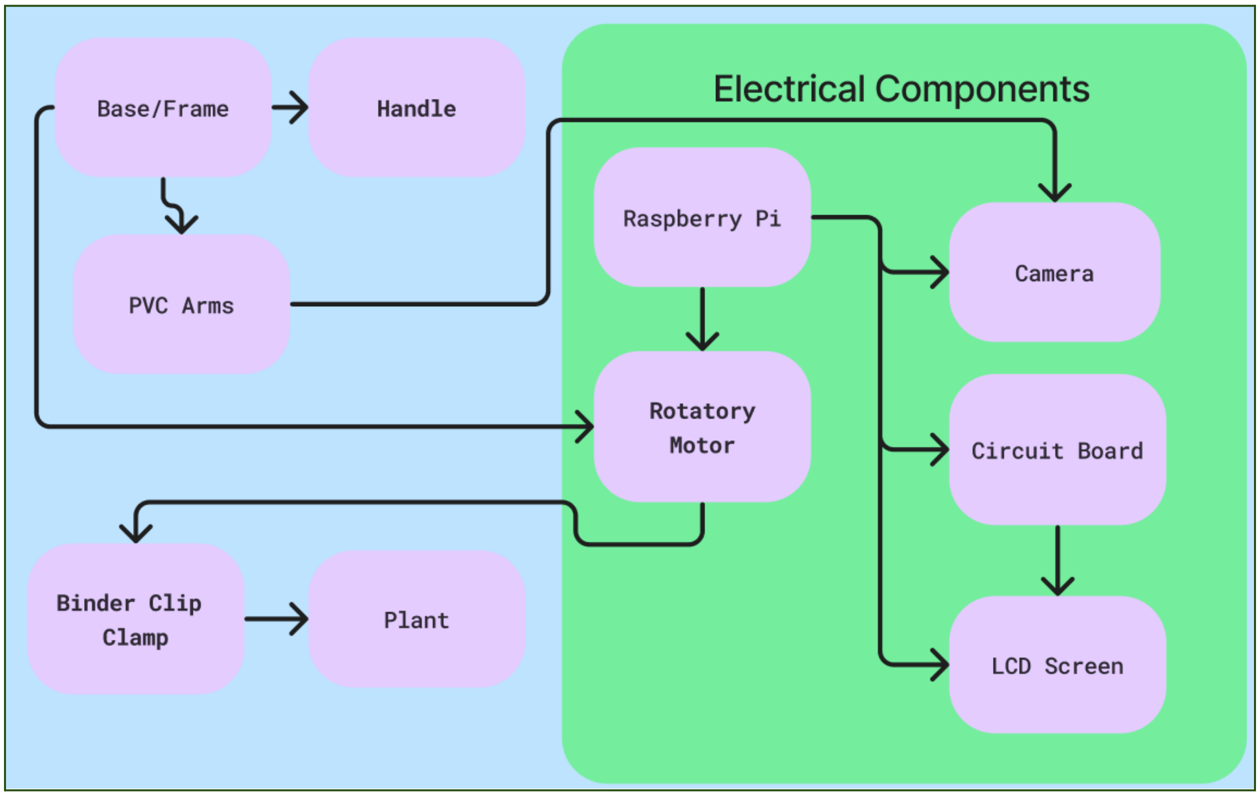


Figure 2.4-2 Systems-Level Diagram of Parasitic Cyst Detector

2.4.1 Mechanical

Plastic Base will operate as the housing and attachment point for the electrical and mechanical components.

PVC arms are attached to rotatable attachment points allowing for the arms to fold down during transportation. Handle is strictly for user to carry the product.

Binder Clips have the properties for securely holding onto the plant: high clamping capabilities with force applied to release the plant. Thus, the plant will not be easily released during rotation.

2.4.2 Electrical

Raspberry Pi acts as the control unit and communications center for the product. It will be coded with autonomous functions that the user can activate with buttons.

Primary function is as follows:

1) User interacts with Pi to start image capturing and counting process.
2) Camera will take first picture of the plant (User will have set up the product for operation prior).
3) Picture is sent to Raspberry Pi where the internal counting algorithm is run on the image.
4) Results of counting algorithm are stored.
5) The Motor receives the signal to rate the plant and rotates the plant 180 degrees.
6) Camera will take the second picture of the plant. (User will not need to do any additional setup prior)
7) Picture is sent to Raspberry Pi where the internal counting algorithm is run is run on the image.
8) Results of counting algorithm are stored and combined with previous results. Process will manage double counting but not prevent it.
9) Final results are sent from the Pi for display on LCD screen.

3 Statement of Work

3.1 Previous Work And Literature

The issue of soybean nematode cysts has been studied since the 1950s, with current processes very similar to the ones developed back then. The two current processes are discussed below, along with relevant technology and machine learning algorithms.

3.1.1 Existing Processes

From talking to Dr. Greg Tylka in Plant Pathology and Dr. Santosh Pandey in Electrical Engineering, we know there are a couple of processes for counting these nematodes that already exist.

1. In the **corporate** research environment, soybean samples are visually scanned by experts to get an estimate of how many cysts are present. This method is imprecise and requires a subject matter expert, i.e. lots of investment in training or finding appropriate personnel.
   - The strength of this method is primarily the simple workflow. A single person takes a sample, examines it, reports their finding, and then disposes of the sample
   - However, this process requires highly skilled labor, which could potentially be cost-prohibitive, as well as limit the quantity processed.
2. In the **academic** research environment, soil samples are sieved to filter out the appropriate size particles to capture cysts. Then, the cyst-sized particles are ground down to release eggs and create a sample. The sample must then be dyed and a prepared on a

microscope slide, and finally visually examined by either an algorithm or human researcher to count the number of eggs, discounting any dirt that remains in the sample.

- The advantages of this process are consistency and accuracy. The methods employed in this environment are known and frequently used, and the accuracy is significantly higher since they are counting the individual eggs.
- The primary drawbacks of this method are that it is both time-consuming and requires specialized equipment. This means that it isn't ideal for high-volume use cases.

Context: Eggs vs. Cysts

Eggs are different than cysts - each cyst contains 200-250 eggs. In process (2) above, cysts are ground down to release the eggs, which are then counted under a microscope.

This project will be detecting cysts.

3.1.2 Existing Technology

1. There was a company founded (Creative AI) to assist in process (2) above, which created the algorithm referenced in that process. A plant pathology professor, Dr. Tylka currently uses Creative AI to assist in his lab research. However, that company went out of business, and its founders are now working for other companies on unrelated projects.
2. NVidia is a big name in the AI market, and with a product line known as Jetson designed for embedded devices and usecases, it provides hardware and software designed for embedded systems using AI and deep learning. [8]
3. The Detectron2 project, published on GitHub as an open source project by Facebook's research division, contains an implementation of Mask R-CNN, an improved version of Faster R-CNN, discussed below. Included in their project are details on how to train a Mask R-CNN model with a custom dataset, and we are exploring using this implementation as the machine learning algorithm in our project.

3.1.3 Existing Algorithms

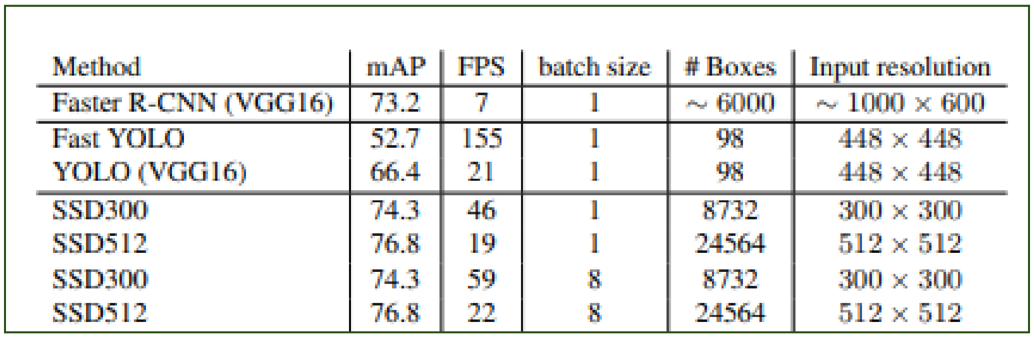
We did extensive background research on object detection algorithms in order to choose an appropriate algorithm for our use-case. In general, object detection algorithms have two functions: object detection, and object classification. Object detection is used to find the presence of distinct objects in an image, and a classification model is used to predict *what* an object is. Classification models have to be trained on certain object classes.[9]. In our case, we will be using an existing implementation of an object detection & classification algorithm to detect and count cysts in a image of a soybean root.

We are following the Faster R-CNN process for machine learning object detection. This process is slower in counting the cysts than the state-of-the-art for real-time object detection. However, with a slower process comes higher accuracy. We discovered that Faster R-CNN is a top performing algorithm for classifying objects, especially small objects[3]. Upon successful implementation, we expect to exceed our clients accuracy expectations by a considerable margin.

Below is a table comparing our chosen algorithm, Faster R-CNN, with other contemporary deep learning algorithms. Notably absent is the fact that Faster R-CNN is known to have a better accuracy with small objects in images.

Faster R-CNN [3]	You Only Look Once (YOLO) [10]	Single Shot Detector (SSD) [11]
5 FPS	20-150 FPS	20-50 FPS
Uses Region Proposal Network to generate regions containing objects for classification.	Uses Anchor Boxes to generate regions containing objects for classification.	Uses Anchor Box Pyramids to generate regions containing objects for classification.
Uses Convolutional Neural Network to classify objects.	Uses Convolutional Neural Network to classify objects.	Uses Convolutional Neural Network to classify objects.
Algorithm Training takes a considerable amount of time.	Algorithm Training takes a moderate amount of time.	Algorithm Training takes a moderate amount of time.
2 stage network- Additional stage is Region of Interest Pooling layer which filters out bad predictions improving accuracy and speed later.	1 stage network	1 stage network
73% mean Average Precision (mAP)	50-65% mean Average Precision (mAP)	75% mean Average Precision (mAP)
Capable of handling high resolution images.	Capable of handling high resolution images.	Capable of handling high resolution images.

Table 3.1-1 Object Detection Algorithm Comparison



Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	∼ 6000	∼ 1000 × 600
Fast YOLO	52.7	155	1	98	448 × 448
YOLO (VGG16)	66.4	21	1	98	448 × 448
SSD300	74.3	46	1	8732	300 × 300
SSD512	76.8	19	1	24564	512 × 512
SSD300	74.3	59	8	8732	300 × 300
SSD512	76.8	22	8	24564	512 × 512

Figure 3.1-1 Research illustrating the results of running several machine learning algorithms. Note that the algorithms were not run on the same data set for each case and are not indicative of small object performance.

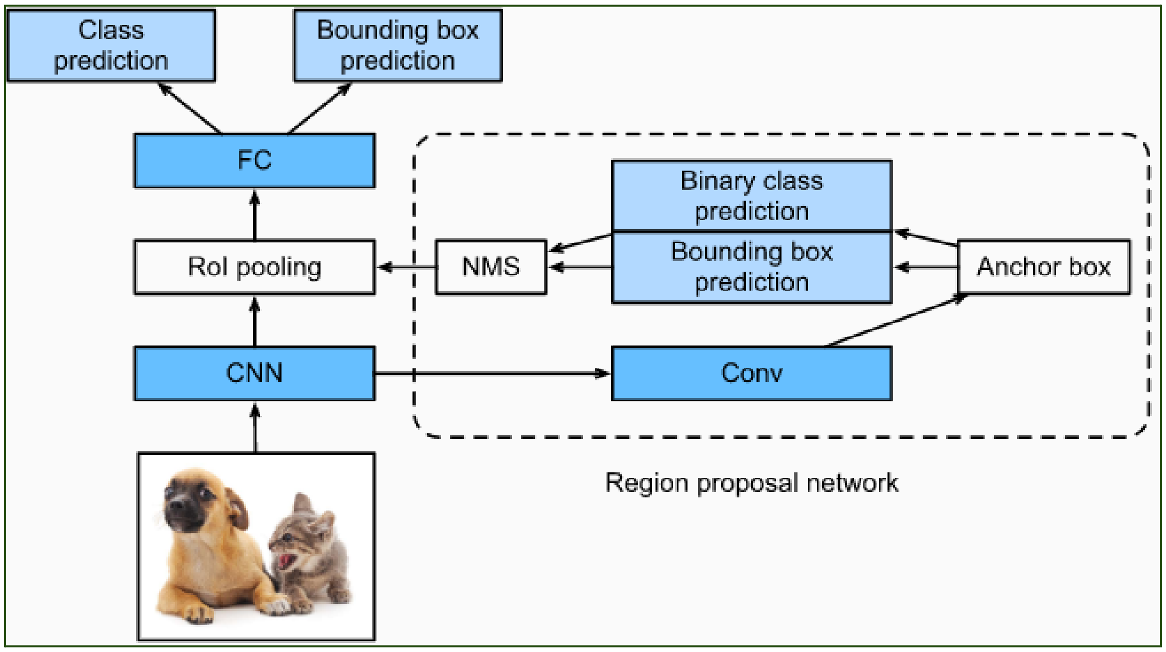


Figure 3.1-2 Model architecture of the Faster R-CNN model.

Fast Facts [9]:

- Not real time (as in not “instant” decision making but still it’s fast at <1 s.)
- Two-Stage
- Algorithm is recursive
- 3 main steps, illustrated in Figure 3.1-2.
  - Generating region proposals
  - From each region proposal, a fixed length feature vector is extracted using image descriptors

    - Feature vector is used to assign each region proposal to either background or object classes
    - Similar setup to a generic object detection pipeline
- Maximize the efficiency of GPU

Feature Extraction [3]

- Region Proposals are then used to extract feature vectors.
- Feature extraction is conducted using anchor boxes.
- Anchor boxes that cross boundaries are ignored during training improving feature recognition.
- Uses Pyramid of Anchors to improve performance.
- Feature vectors are extracted from regions using the ROI pooling layer.
- ROI pooling layer splits region proposal into a grid, max pooling is applied and returns a single value
- Feature vector is as large as the number of elements in the grid

Classification [3]

- The feature vectors are then inputs into a CNN classifier.
- CNN used for image classification.
- Classified using Fast R-CNN
- In Faster R-CNN, the creation of region proposals and object detection are performed on the same CNN. Which, algorithm for regional proposal and the algorithm for detection are combined for the faster detection speed.

Drawbacks [3]

- Multistage model, cannot train end to end
- Each region is fed independently to CNN for feature detection
- Impossible to run real time
    - "Real Time" is defined as near-instantaneous decision making (milliseconds)

3.2 Technology Considerations

There are a couple different technologies to examine when looking at our approach towards this project. For example, as mentioned above, there are two existing approaches to doing this sort of work. The first approach involves counting the number of cysts infecting a soybean plant by eye, and the second approach involves taking soil samples from farms and processing them in order to count the number of eggs.

When examining the first approach, the strengths are clear - it is a quick, easy workflow that is designed to work for large-scale operations. An expert can get an estimate of the number of cysts relatively quickly by eye. However, the drawback is just as clear - the accuracy. When speed, scale, and volume are the primary concerns, accuracy might fall by the wayside.

A possible alternative – and indeed, the approach for this project – is to use computer vision (a computer running an object detection algorithm on an image) to count the number of cysts on the plant. This is also scalable, as a device will be able to process a plant fully in under 10 seconds, and devices will be cheap enough to produce at scale. Where it might improve is the accuracy of the count.

Examining the second approach, we can also clearly identify the strength. Processing the soil sample and meticulously counting the eggs in solution ensures high accuracy and consistency. However, the tradeoff for this method is that it takes significantly more manpower - someone has to sieve the samples, grind down the cysts, prepare the solution, and run Creative AI's program on each solution.

Our approach reduces the manpower of this approach significantly. Granted, even with high accuracy, it may not be as precise as the laboratory approach (due to the variable number of eggs in each cyst). However, if we can achieve comparable accuracy with a mostly automated process, it will speed up the process significantly.

Context: Soil Sample Processing

The laboratory process may have different use cases than our solution, as it is designed to test soil samples, whereas our device will be scanning soybean plants.

3.3 Task Decomposition

This section of the document will break down the tasks that will be required to complete the project. We separate them out into two main tasks, for hardware and software, and go into detail from there.

1. Develop a deep learning model suitable for our data
    a. Research and choose a deep learning algorithm/model
    b. Label our existing data
    c. Implement our model in our environment
    d. Train the model on our data
    e. Validate & test the model
    f. Optimize the code for enhanced improvement
    g. (Optionally) implement additional models for comparisons.

2. Prototype a device to scan a soybean plant and run our model on it
    a. Set up a controlled environment for image capturing
    b. Develop scanner that can scan all sides of the plant
    c. Apply the machine learning detector to the scanned sides to accurately count of the parasitic cysts.
    d. Optimize the prototype to be user friendly and intuitive.

3.4 Possible Risks And Risk Management

Our project has some risks involved that require risk mitigation. Each of these risks includes the probability of the risk occurring and the plan we have in place to mitigate it. Our risk and mitigation plans are divided into our two main tasks, as well as general risks we might encounture during this project.

Task 1 - Developing a Deep Learning Mode

Risk	Probability	Mitigation Plan
Available data is not enough to train an accurate algorithm on.	0.8	Develop a proof-of-concept model and allow the project administrators to collect more data over time to improve the model.
Labeling tools are incompatible with algorithm implementation.	0.1	N/A
Our algorithm does not provide a sufficient amount of accuracy rating.	0.1	Since we have a relatively low goal accuracy (~50%), even with our limited dataset we should be able to achieve this.

Table 3.4-1 Deep Learning Model risks and mitigation plans

Task 2 - Developing a Prototype Device

Risk	Probability	Mitigation Plan
Our hardware does not have enough resolution for machine learning to detect.	0.5	Doing market research to find a high-resolution camera at an affordable price

		This may violate some of our requirements budget wise. For example, our budget is currently set at $500, however, in order to get higher resolution, expensive cameras might be needed.

Table 3.4-2 Prototype Device Risks and Mitigation Plans

General Risks:

Risk	Probability	Mitigation Plan
No finite ending	0.2	Good documentation will avoid lost progress and help advancements in our project
Less predictability, especially since no one has a strong background in this area	0.4	Spending time researching machine learning can help us anticipate issues we might have developing and working on an algorithm

Table 3.4-3 General Risks and Mitigation Plans

3.5 Project Proposed Milestones and Evaluation Criteria

This section lists the key milestones for our project, and how we will measure our progress and success in each. These milestones are roughly sequential, but progress will likely be made on multiple at the same time.

Milestone	Description	Metrics
1: Background Research	Progress is complete when we understand artificial intelligence, machine learning, and deep learning.	Progression through team training and onboarding tasks.
2: Algorithm Research	Progress is complete when the team members have sufficient understanding of machine learning algorithms to evaluate multiple different deep learning approaches	Number of algorithms researched & evaluated (3 per person).

Milestone	Description	Metrics
	and assess which one is best for our project.	
3: Algorithm Implementation	Progress is complete when a Faster R-CNN implementation/template is found on GitHub and is runnable on our computers.	Progression through developing a functional Faster R-CNN implementation.
4: Labeling Data (149 total images)	Progress is complete when all the data has all the cysts on the plants labeled.	Number of completed images.
5: Algorithm Training	Progress is complete when we have developed a model based on the implementation in milestone 3, trained on our soybean cyst data.	Progression through training set of soybean cyst data.
6: Algorithm Testing (50%+ accuracy)	Progress is complete when the algorithm sufficiently hits a high accuracy rating (For our purposes it is currently set at 50% subject to change).	Evaluation of results should show accuracy of model should be at least 50%.
7: Hardware Design	Progress is complete when the Hardware System-Level Diagram is properly developed and finalized.	Iterative development process with improving diagrams and ideas.
8: Hardware Implementation (50%+ accuracy)	Progress is complete when we have a working prototype of the image capturing device that can run the machine learning model.	Progression of creating the design from the previous milestone.
9: Hardware Optimization	Progress is complete when the hardware implementation is friendly and can be run by not an engineer.	Iterative development process dedicated to improvements over functionality.
10: Algorithm Optimization	Progress is complete once the counting accuracy of our algorithm has increased by a significant amount (5% or more).	Increase in accuracy of model.
11: Documentation (Website)	Document our design process, implementation, hardware & software designs, and provide next steps.	Progression of team managed website including documentation from each of the stages of the project.

Table 3.5-1 Milestone Descriptions and Metrics

3.6 Project Tracking Procedures

Our project will use agile project management to complete milestones across several sprints. We will have 1 week sprints because of the requirement for weekly reports and the fact that our team meets on Mondays and Tuesdays. Our Monday meetings will be sprint retrospective meetings and sprint planning meetings.Additionally, we felt that it was important - since we are doing research none of us are familiar with - to meet regularly and discuss our notes and work.

This style has worked well for us because we have multiple facets to our project with various stages of development. In this case, we have a research phase, a implementation phase, and an optimization phase. By subdividing these phases, we have had consistent group and individual assignments designed to promote group discussion and feedback from our client.

Agile project management also has a more iterative approach than the waterfall project management style. After our senior design project ends, we expect some improvements to be made to our project. By using the agile methodology, we can brainstorm the improvements that could be made after our initial machine learning model and accompanying device provided there is enough time.

We will use Gitlab to track our progress.

3.7 Expected Results and Validation

Our project's deliverables will include the following: A prototype soybean scanner, integrated with a Raspberry Pi (or other computing device), camera, and LCD screen, and a machine learning model trained on soybean cyst data to identify cysts on soybean roots in an image. The machine learning model will be integrated with the device and able to run on the Raspberry Pi.

We have a dataset that we will use to establish the accuracy of the machine learning model after training. According to our project stakeholders, if we can achieve 50% accuracy in our machine learning model, this project will have been a success. We cautiously hope to beat this benchmark significantly, but we have yet to have any results.

4. Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

The team recognizes that this is a complex project that involves a lot of work, and as such, we have developed a schedule to keep us on track and making progress. We will be using 1-week sprints, except during the summer, when we will be treating each month as a sprint.

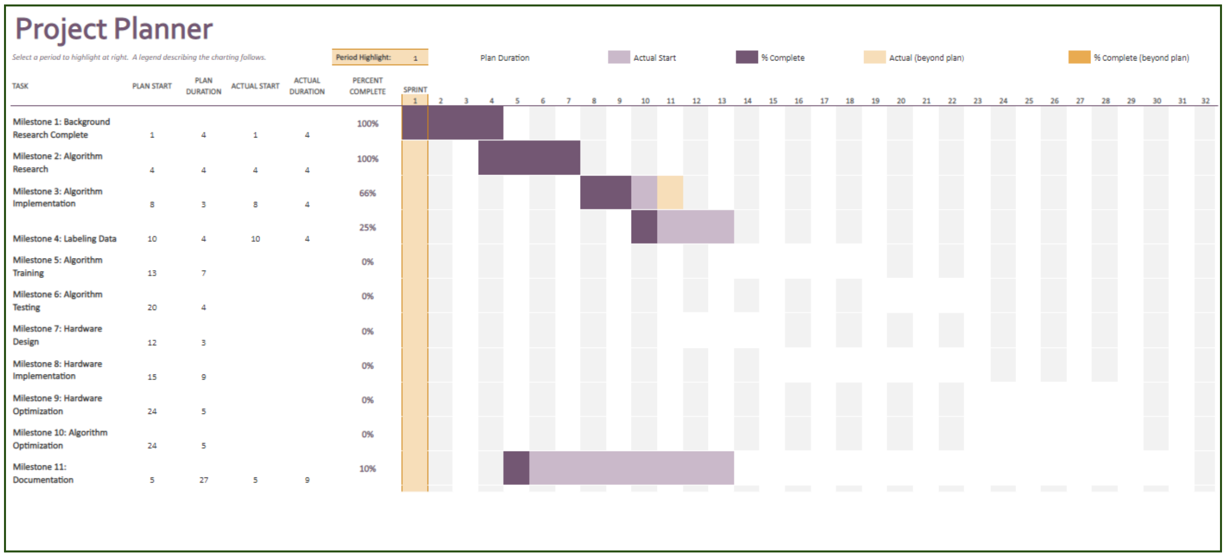
The overarching structure of our timeline is treating the first semester as planning, design, research, and preparation, and the second semester as time for implementation, testing, and optimization. As you can see below, by sprint 15, we should be done with the research and design. After that, we'll move on to training the algorithm we implemented and creating a prototype of the hardware.

Our agile development process allows us to make progress on multiple milestones at the same time, as we progress through the sprints. Below is a table summarizing when each milestone is scheduled to be completed.

Sprint Number (Out of 32)	**Milestone(s) Complete**
Sprint 4	Milestone 1: Background Research Complete
Sprint 8	Milestone 2: Algorithm Research
Sprint 12	Milestone 3: Algorithm Implementation
Sprint 13	Milestone 4: Labeling Data
Sprint 15	Milestone 7: Hardware Design
Sprint 20	Milestone 5: Algorithm Training
Sprint 24	Milestone 6: Algorithm Testing Milestone 8: Hardware Implementation
Sprint 28	Milestone 9: Hardware Optimization Milestone 10: Algorithm Optimization
Sprint 32	Milestone 11: Documentation

Table 4.1-1 Projected Milestone Deadlines

We have also included a Gantt chart showing our current progress and planned timelines for each milestone below.


Project Planner

Select a period to highlight at right. A legend describing the charting follows.

Period Highlight: 1

TASK	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Milestone 1: Background Research Complete	1	4	1	4	100%
Milestone 2: Algorithm Research	4	4	4	4	100%
Milestone 3: Algorithm Implementation	8	3	8	4	66%
Milestone 4: Labeling Data	10	4	10	4	25%
Milestone 5: Algorithm Training	13	7			0%
Milestone 6: Algorithm Testing	20	4			0%
Milestone 7: Hardware Design	12	3			0%
Milestone 8: Hardware Implementation	15	9			0%
Milestone 9: Hardware Optimization	24	5			0%
Milestone 10: Algorithm Optimization	24	5			0%
Milestone 11: Documentation	5	27	5	9	10%

Figure 4.1-1 Gantt Chart of current milestone progress

4.2 Feasibility Assessment

Our title "Developing a Deep Learning Model to Automatically Detect Microscale Objects in Objects in image and Videos" indicates along with the project proposal that we should be strictly developing a deep learning algorithm and training it to identify parasitic cysts in images. We project that we can develop and train the model as well as design and develop a prototype that will apply the model in the field. This is achievable because the counting algorithm model is not new but an application of preexisting models with different training.

We believe that the counting algorithm model is the highest piece of value for our product. Consequently, achieving a working counting algorithm model is our highest priority. We have done extensive research into understanding the machine learning algorithms available and selected one that is used in many applications and therefore should be implementable by us. This does take into consideration the challenge that no one on our team is specialized in machine learning or has any background experience in the field. We have also devoted a considerable amount of time into labeling our dataset for the training process. As a result, we also have developed the training dataset for our model to be trained successfully.

With considerable resources and time devoted to the machine learning algorithm, we did take away time from developing the hardware to support the model. We have done research into each of the components and found that our end product should function as intended.

However, not enough time has been invested into the hardware as yet to develop a prototype and test the intended functionality of the hardware. The lack of time available for this will continue to be a challenge in the future. And, if the components need to be ordered or get backlogged, this will hamper our ability to complete the hardware component of this project. Since the hardware application is an additional feature we as a team agreed to design and develop, we do not expect this to hinder our progress towards our original goal.

Summarily, the model will always be achieved and tested and well developed. However, the hardware prototype may not reach full development due to time delays and constraints that may be imposed on us in the future. Thus, the product is feasible provided that we can acquire all the necessary components within a reasonable amount of time.

4.3 Personnel Effort Requirements

Our project requires a significant effort by each member of the project team. We have divided our person-hours projection into the two main tasks of our project. Each subtask has been assigned the expected person-hours and an explanation of why we think each subtask should take the time it was assigned.

4.3.1 Task 1 - Developing a Deep Learning Model

Subtasks	Person-hours	Explanation
Research and choose a deep learning algorithm/model	20	This task is vital to our project and we cannot train and test our algorithm until we've chosen one we feel is best for this specific project
Label our existing data	40	We need to label the data set to train our algorithm. Unfortunately, this will have to be done manually and will take a considerable effort to find and label all the cysts in every image.
Implement our model in our environment	10-15	This task's time is dependent on if we can find an existing implementation that works on all our computers. If we have a template, the work will be significantly easier.

Subtasks	Person-hours	Explanation
Train the model on our data	40	On average, it takes a few months to train a machine learning model. However, it takes a while to run the algorithm, so person-hours will be less than a few weeks.
Validate & test the model	20	It takes less time to validate the model than to train it.
Optimize the code for enhanced improvement	10	Our professor who oversees this project has specified we are not going for a model with high accuracy. This means there will likely be less optimization needed. However, some optimization will be needed especially if the implementation we find does not specialize in small objects.
(Optionally) implement additional models for comparisons.	60	This is an estimation based on the person-hours for implementing, training, and testing one model. We anticipate, however, this would take less time because we have a better understanding of the process of machine learning
Software Documentation	50	The documentation should be detailed enough for someone to replicate the project based on the documentation alone. As such, it will take a lot of time to develop.
Total Hours:	190-255	N/A

Table 4.3-1 Software Person-Hour Estimate Breakdown

4.3.2 Task 2 - Developing a Prototype Device

Subtasks	Person-hours	Explanation
Set up a controlled environment for image capturing	10	A brainstorming/researching effort will be made to find an optimal image environment that helps the cysts stand out.
Develop scanner that can scan all sides of the plant	20	This portion will be building the scanner with the background environment. The prototyping will also need testing and revision. Thus, extra attention will be given to this task.
Apply the machine learning detector to the scanned sides to accurately count of the parasitic cysts.	15	To avoid double counting cysts, we will have to add additional functionality to the detector that can cross-reference multiple images. Due to this, this task should take less time than the algorithm.
Optimize the prototype to be user friendly and intuitive.	10	Our end goal is product a farmer could use - not an engineer. Thus, we need it easy to use with few errors. The time to accomplish this will largely depend upon how well the prototype was developed
Hardware Documentation	50	The detail that is going into the descriptive documents will be enough for anyone to replicate and reproduce our product. Consequently, the documentation is half the project due to its importance.
Total Hours:	105	N/A

Table 4.3-2 Hardware Person-Hour Estimate Breakdown

Total Projected Person-Hours: 295 - 355.

4.4 Other Resource Requirements

In addition to time and money, our project will require some additional resources. These resources are detailed below. The hardware resources are devoted to building our prototype. The software resources are devoted to creating the environment for the best results for our training and testing.

Hardware	Software
Camera	Computing Platform to train the algorithm.
Small Motor	Soybean Plant Root Pictures for dataset.
Power Source (Battery)	
Binder Clip/Clamp	
PVC	
Colorized Pastic Base Platform	
Raspberry Pi	
Handlebar	

Table 4.4-1 Hardware and Software Resource Requirements

4.5 Financial Requirements

If relevant, include the total financial resources required to conduct the project.

Item	Cost
Raspberry Pi 4	$45
LCD Screen	$10
Small Motor	$5
Binder Clip	$1
PVC (4ft)	$20

PVC 90 Degree Fitting x2	$5
Colorized Pastic Base Platform (3d Printed)	$25
Handlebar	$10
PVC Swivel Fitting x2	$10
Raspberry Pi Camera v3 (4k)	$40
TOTAL	$171

Table 4.5-1 Financial Estimate and Parts List

5. Testing and Implementation

5.1 Interface Specifications

Since our project consists of both software and hardware elements, there is a lot of interfacing between the two that will need thorough testing.

5.1.1 Hardware Interface

In order to create the image capturing device, we will need to integrate components with a raspberry pi. This includes a camera and an LCD screen to display the number of cysts on the soybean plant. We will verify the components we buy are compatible with a raspberry pi whenever we buy them.

We must also integrate our raspberry pi and all attached components with the image capturing environment. In order to test this, we will verify that the environment created is controlled and does not allow for variation that will impact the accuracy of our machine learning model. We will also verify our environment created does not damage the soybean plants in any way.

5.1.2 Software Interface

The machine learning algorithm, which is faster R-CNN, with the given picture taken by the raspberry pi camera, should be able to identify and return the value of how many cyst are in the soybean's root.

5.2 Hardware and software

We will be testing many of the hardware components through small programs we write on the raspberry pi. For example, to test the camera, we will take regular images to test the resolution and usability of the camera. We will also take pictures of the soybean roots before integrating it into the image capturing environment. The LCD screen will also be tested through a small program written on the raspberry pi prior to integration with the scanner.

5.3 Functional Testing

This section of the document will describe our approach to unit testing, integration testing, system testing, and acceptance testing.

5.3.1 Unit Testing

We will find an implemented Faster R-CNN algorithm to meet the time required for this project. One unit being tested within our implemented Faster R-CNN algorithm is the accuracy of the machine-learning model. We are trying to create a machine-learning model that detects soybean cyst nematodes with 50% accuracy. After training the data set, we plan to test it with other pictures of soybean roots and validate the number of cysts given by the model by manually labeling our images and seeing if our model is above our wanted 50% accuracy.

Context: Accuracy

Accuracy is defined for this project as having the output be within 50% of the actual answer. An image with 100 cysts could output a number within the range [50, 150] for the model to be considered 50% accurate. For an image with 200, we would accept output in the range [100, 300].

This is a **different metric** than the Object Detection industry standard also called accuracy, which refers to a combination of several metrics, including how confident the model is and how often it is right compared to ground truth data.

An important hardware component that will need to be tested before integrating the component into our system is the camera. The images our machine learning model is being trained on are very high resolution that we might not have with a camera that is compatible with a Raspberry Pi. A lot of market research is being done to find a high-resolution camera for our image-capturing device, but we will be taking pictures with the camera to ultimately determine what we plan on using for our final prototype.

Another important hardware component for this project is our Raspberry Pi. Currently, we plan to use a Raspberry Pi 4, but we will test this to verify it will have the memory to upload a machine learning algorithm on. We will also verify that running a machine-learning model is not too slow for the end users.

We will also have to test our image capturing environment setup. To do this, we will verify there is nothing in the image-capturing environment that would allow for inaccurate results. For example, if there is reflective material being used that could create bright spots in the photos. We would also ensure that our environment would not harm the plant in any way. We would set up the environment with a test plant and make sure nothing in our image-capturing setup harms the cysts or the soybean roots.

5.3.2 Integration Testing

After the user fixes the soybean on the board, the user should be able to press some kind of start button, then the signal will go through the raspberry pi, which will send signals to the camera, by plugging the ribbon connector of the camera module into the connector on the Raspberry Pi. After taking a picture of the different sides of the soybean, our machine learning algorithm, implemented in the raspberry pi, should receive and be able to use that picture. After processing is done in the machine learning part, LCD board connected with the wire as well, will display the number of the cysts.

The largest integration path in our system is between the machine-learning model and the image capturing device. We want to allow the end users to be able to capture images of soybean roots without needing to upload images onto a computer. The software and image capturing device integration testing will be the last step after verifying the accurate machine-learning model and uploading it onto a raspberry pi.

5.3.3 System Testing

The system-level testing strategy is going to be a composition of previous testing strategies discussed in this document. The overall system should all work together at a bare minimum, meaning the machine-learning algorithm should run on a raspberry pi detecting cysts on an image of a soybean root captured by our camera. In order to verify this, we will compare the system-level performance with the unit-level performance of the machine-learning algorithm to ensure they achieve similar results.

This will require we have results from previous unit tests ensuring the camera's resolution is appropriate for the machine learning model, results that give us a baseline for accuracy of our machine learning model, and results that confirm the raspberry pi can connect with all requisite devices and run the machine learning model.

Once we have those results, we can test the system as a whole, ensuring that the time it takes to process an image is reasonable, its accuracy is comparable to the baseline we established, and that the integrated process works from image capture to output of the number of cysts

5.3.4 Acceptance Testing

Requirement 1:

Our client requires at least 50% accuracy for parasitic cyst count.

To show that this requirement is being met, we will use a image with the parasitic cysts labelled. The cysts can then be counted rather easily by using the number of objects labelled. Then, we can use our product to generate another count of the cysts for comparison. Finally, to get accuracy, we take the ratio of our products results over the number of labelled cysts as a percentage, which is ideally over 50%.

Requirement 2:

Soybean that was used as a specimen should be undamaged.

To demonstrate this requirement is being met, we will show that the plant is not destroyed by our product during its use.

Requirement 3:

Our machine learning algorithm should run on the raspberry pi without error.

To demonstrate this requirement is being met, we will show that our product has appropriate error handling and the output is as expected.

5.4 Non-Functional Testing

Requirement 1:

Price of our product should be affordable, around $500.

To show that this requirement is being met, we will have itemized parts list that should demonstrate the total cost for all our parts is less than $500. (For this product to turn into a business, we would need less than $100 as it is recommended to sell a product for 5x its production cost).

Requirement 2:

Our product should be sturdy enough, therefore it is possible to reuse.

To demonstrate that this requirement is met, we will show our product does not break under normal stresses such as being dropped or having an object dropped on it.

Requirement 3:

Users should interact with our product with as few steps as possible.

To demonstrate this requirement is met, we will show our product is like a black box where the user only needs to provide the inputs, in our case the plant, push a few buttons to operate the product, and then receives the outputs. They should not need knowledge of the internal workings of the product to use it.

Requirement 4:

Our product should be easy to use

With a few steps, users should be able to use and get data using our product. To show that this requirement is being met, our client should be able to use our product with as few instructions as possible to take a picture of a soybean plant and generate the parasitic cyst count on the plant.

5.5 Process

We have not yet started the testing process, but have plans for how each portion of the project will be tested. Each hardware component will undergo unit testing and integration testing before the final product is created. We plan on breaking down our systems level schematic into smaller sections and preforming many tests.

5.6 Results

Since we are only half way done with our project, we do not have any testing results to show yet. Over the summer, we will be working on training our machine learning model and will have results either by the end of summer or early next semester. We have a plan to quickly get our first hardware prototype done early next semester and will have testing results to show.

6. Closing Material

6.1 Conclusion

This semester, we have focused on the research and preparation of our project. We started with a basic overview of machine learning, artificial intelligence, and neural networks. Then, we researched object detection algorithms, such as Faster R-CNN, You Only Look Once (YOLO), and Single Shot Detector (SSD). After deciding Faster R-CNN would be the best algorithm for our small object detection application, we spent a few weeks finding an implementation of Faster R-CNN that we could upload our data set on. While doing research, we have been labeling our data set and designing our image capturing device.

Our goals for the summer is to train and test our algorithm and potentially test a few different ways to process images to get a more accurate model. Some time before the fall semester, we will also finalize our list of parts to start a prototype of our hardware device. During the fall semester, we will finalize testing and training our machine learning model. We will also build, test, and optimize our image capturing device.

Our device will allow for easy and fast image capturing to find the cysts soybean roots. The current methods of finding cysts consist of sifting the cysts off the roots of soybeans and manually counting them. Our method provides a way to automate the process of counting cysts on a soybean plant and will help researchers, and eventually farmers, learn how to prevent soybean cyst nematodes.

6.2 References

[1] Brooks, Rhonda. “New Program Tackles the $1.5 Billion Bite SCN Takes out of U.S. Soybeans Annually.” *AgWeb*, 18 July 2022, https://www.agweb.com/news/crops/soybeans/new-program-tackles-15-billion-bite-scn-takes-out-us-soybeans-annually.

[2] Gamage, Omega. “Car Detection from a Drone: A New Angle.” *Medium*, ACCELR Blog, 10 Sept. 2021, https://medium.com/accelr-blog/car-detection-from-a-drone-a-new-angle-821cc6fff422.

[3] S. Ren, K. He, R. Girshick, J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” Cornell University ArXiv, 2016 [Online]. Available: https://arxiv.org/abs/1506.01497v3

[4] "American National Standard for Metric Practice," in ANSI/IEEE Std 268-1992 , vol., no., pp.1-80, 28 Oct. 1992, doi: 10.1109/IEEESTD.1992.114377.

[5] "IEEE Standard for DevOps:Building Reliable and Secure Systems Including Application Build, Package, and Deployment," in IEEE Std 2675-2021 , vol., no., pp.1-91, 16 April 2021, doi: 10.1109/IEEESTD.2021.9415476.

[6] "IEEE Guide--Adoption of ISO/IEC TR 24748-1:2010 Systems and Software Engineering--Life Cycle Management--Part 1: Guide for Life Cycle Management," in IEEE Std 24748-1-2011 , vol., no., pp.1-96, 3 June 2011, doi: 10.1109/IEEESTD.2011.5871657.

[7] "ISO/IEC/IEEE International Standard - Software engineering - Software life cycle processes - Maintenance," in ISO/IEC/IEEE 14764:2022(E) , vol., no., pp.1-46, 21 Jan. 2022, doi: 10.1109/IEEESTD.2022.9690131.

[8] “Nvidia embedded systems for next-Gen Autonomous Machines,” *NVIDIA Jetson: Accelerating Next-Gen Edge AI and Robotics*. [Online]. Available: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/. [Accessed: 22-Apr-2023].

[9] P. Sharma, “Image classification vs object detection vs image segmentation,” *Medium*, 21-Aug-2019. [Online]. Available: https://medium.com/analytics-vidhya/image-classification-vs-object-detection-vs-image-segmentation-f36db85fe81#:~:text=To%20quickly%20summarize%3A,multiple%20objects%20in%20the%20image. [Accessed: 22-Apr-2023].

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[11] L. Wei, D. Angeulov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. Berg, *SSD: Single Shot MultiBox Detector*, Dec. 2015.

[12] R. Gavrilescu, C. Zet, C. Foșalău, M. Skoczylas and D. Cotovanu, "Faster R-CNN:an Approach to Real-Time Object Detection," 2018 International Conference and Exposition on Electrical And Power Engineering (EPE), Iasi, Romania, 2018, pp. 0165-0168, doi: 10.1109/ICEPE.2018.8559776.

[13] “14.8. region-based cnns (R-cnns)¶ colab [pytorch] open the notebook in colab colab [mxnet] open the notebook in colab colab [jax] open the notebook in colab colab [tensorflow] open the notebook in colab sagemaker studio lab open the notebook in SageMaker Studio Lab,” *14.8. Region-based CNNs (R-CNNs) - Dive into Deep Learning 1.0.0-beta0 documentation*. [Online]. Available: https://d2l.ai/chapter_computer-vision/rcnn.html. [Accessed: 22-Apr-2023].

6.3 Appendix

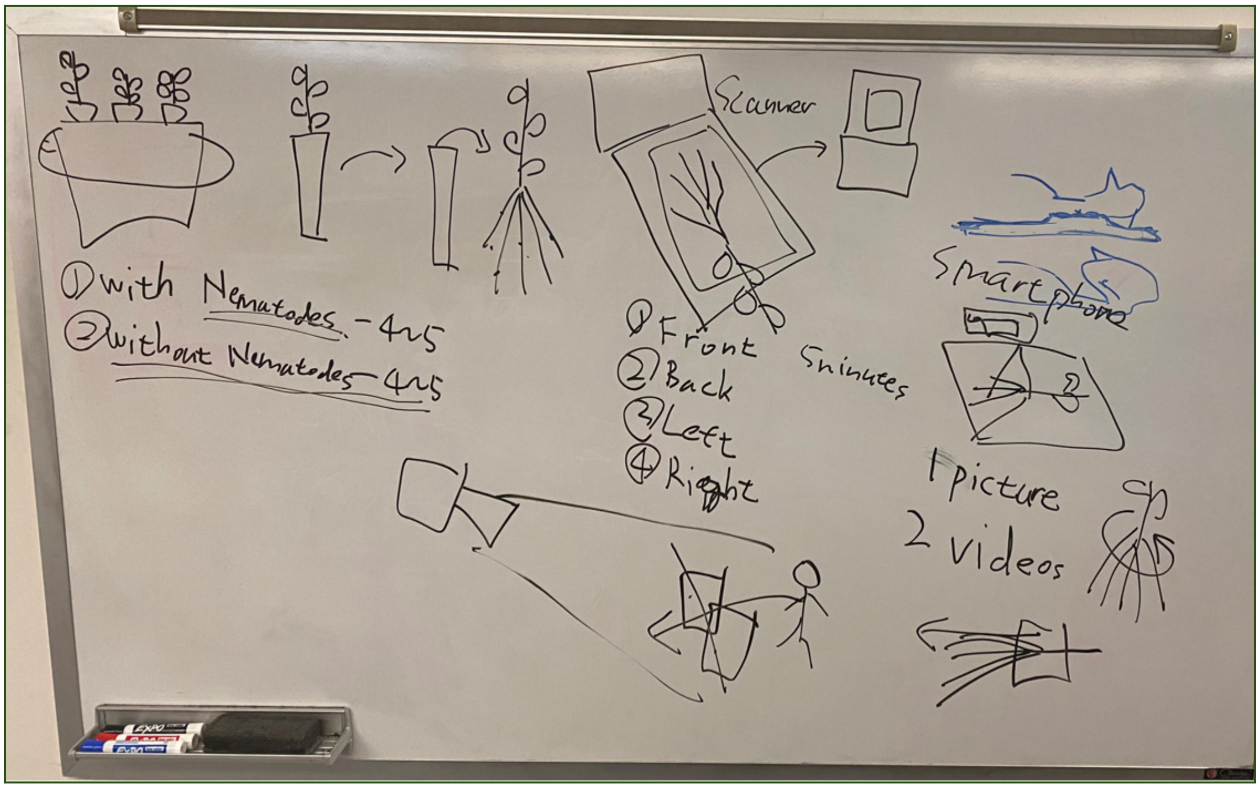


Figure Appendix-1 Current Image Capturing Phase by Phase Diagram

Background Research Compilation Document:
https://docs.google.com/document/d/1mITfIIbpWDMYVwMXOBxhz1OOZPWPzlSX5o5VofkQcLU/edit?usp=sharing

Algorithm Selection Research Compilation Document:
https://docs.google.com/document/d/1RJ1XpO6BM6Xi7iS9vB-CeDnzz7i8VdGND5rzunaQNCM/edit?usp=sharing

Faster R-CNN Research Compilation Document:
https://docs.google.com/document/d/1BST4wBlz1tL_HMQkdW3w6Xb2519I7nstuGS6UsmCbjQ/edit?usp=sharing

You must be logged in to your Iowa State account to access these documents.

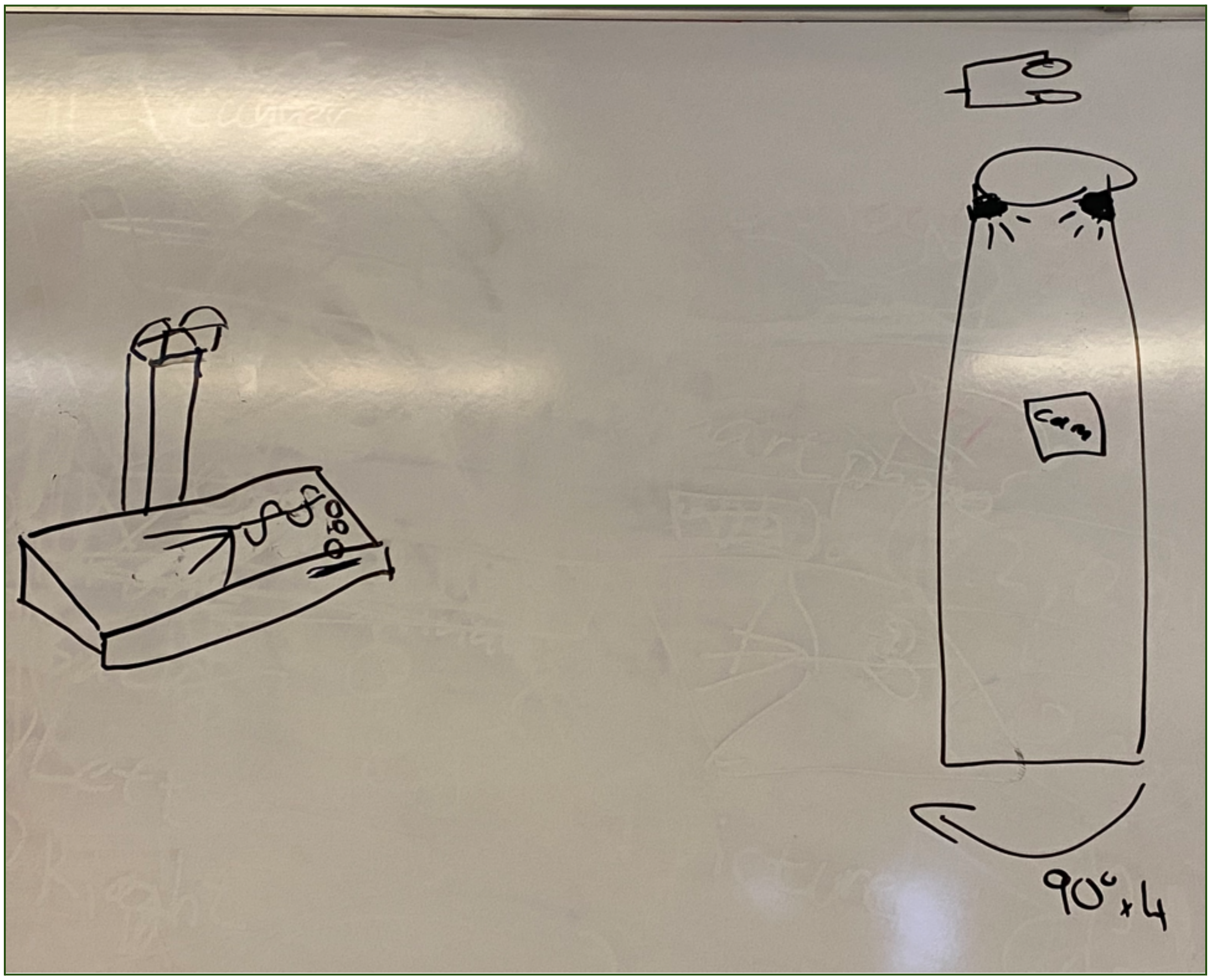


Figure Appendix-2 First Hardware Designs