Western Engineering

MSE 4499 — Mechatronic Design Project

Camera Based Auto-Steer Solution for Agricultural Tractor

Final Report

Nicole Garcia
James Melisek
Kajal Patel
Jasper de Regt
Faculty Advisor: Dr. M Naish
Prepared for: Dr. A. Samani
Word Count: 7499
Date Submitted: April 10, 2018

Executive Summary

This report describes the details of the final design and the prototype of the camera based autosteer for agricultural tractor system. The concept generation and selection from previous reports will be briefly summarized. Additionally, the final design will be divided into three subsystems including the software, camera mounting, and steering system. The details of these subsections will be elaborated on in terms of design, analysis, prototyping and testing. Finally, recommendations for future work on the project will be discussed.

List of Tables

List of Figures

Problem Definition

Farmers currently lack the equipment or technology to automatically steer tractors in post emergence row crop applications, instead relying on manual steering. By automating the steering of the tractor, the operator is better able to focus on all other aspects of the field operation. Our team will design a marketable mechatronic solution for safely navigating a tractor through crop rows, reducing operator effort in post-emergence row crop applications.

Background Information

Tractor operators are required to complete several tasks in parallel while steering the tractor. This design project will focus on the tasks that are relevant to post-emergence row crop applications. Scientifically, post-emergence is defined as all the growth stages in a plant that occur after the plant has emerged from the soil [1]. Row crop is an industry term that refers to rows of plants being spaced far enough apart for the vehicle to drive between the rows, typically a row width of greater than 20 inches [2]. A specific set of field operations, such as spraying, side dressing, and scuffling, occur in row crops post emergence. The operator performs this multitasking while the vehicle is in motion. As a result, the user is prevented from operating the tractor in a safe and effective manner [3]. Currently, the only solution for autosteering a tractor is the GPS auto steer system. However, GPS autosteer is not suitable for post emergence row crop operations because it is not accurate enough to avoid damages.

State of the Art and Emerging Technologies

GPS based autosteer is a current technology that is widely used on farm tractors. GPS autosteer use is somewhat limited in post-emergence row crop operations for two reasons. For one, GPS is not inherently repeatable. GPS signals drift due to changing ionosphere and atmosphere conditions. It is not possible to save a GPS location today, and return to the exact same position using GPS a day later, unless further technology is used. To correct for this drift requires ground referencing and communication hardware. This hardware is offered by GPS autosteer manufacturers but is expensive, costing in the range of ~$20,000 dollars. Farmers typically forgo this option due to expense.

A second reason that GPS autosteer use is somewhat limited in post-emergence row crop operations is that a GPS system has no way of knowing where the crop rows are if the rows were planted whilst steering by hand.

No autosteering technologies outside of GPS based solutions are commercially available in the agriculture industry. Some specialty tractor mounted row crop implements, especially scufflers, are available with a hydraulic mechanism that is able to shift the implement sideways in relation to the tractor. This mechanism is sometimes actuated through a vision system. An operator steers the tractor to drive between the rows with some large tolerance, and the vision system shifts the scuffler so that it works close to the crop rows with some small tolerance. Our solution is somewhat similar to this technology, but is fundamentally different in that we steer the tractor itself. Our solution could replace this side shifting technology.

Scope

The scope of this report will include a description of the design work done during the course of the project. The early stages of the design process will be reiterated, including objectives, constraints, concept generation, evaluation and selection. All major design decisions that were taken, leading up to the final design, will be discussed.

This report will also provide a detailed description of the analysis that was completed to validate the design as well as the prototype. The report will be divided into three main sections, corresponding to the main components of the system; the software, the steering and the camera mount.

The scope of the project will be limited to the three components mentioned. These elements will interact with each other through a Raspberry Pi and Arduino in both the final design and the prototype. Realistically, a custom controller should be designed or selected for the system, however, this component is not within the scope of the project.

Objective and constraints

The following section will condense the general design objectives and constraints of the system. Subsystems that make up the components of the project will also have to meet these general objectives.

Objectives

- Robust: The steering mechanism should operate properly in different lighting and weather conditions
- Safe: Autosteer should be able to be disengaged when desired by the tractor operator
- Reasonably priced: The design should be marketable at a competitive price

- Accurate: The system must accurately detect row crops
- Reliable: Should behave as expected constantly throughout its use
- Universal Fit: Retrofittable to various types of tractors
- Easy to use/install: Able to be installed by average farmer
- Long lifespan: The system's lifespan should be comparable to that of tractors

Constraints

- Safe: The design must satisfy CSA model certification
- Reasonably priced: This solution must be competitively priced. Similar systems from Trimble Inc. can cost from $3,000 to $5,000 USD
- Accurate: Must drive accurately within 3 inches to ensure damage is prevented to crops
- Ease of use: Can be installed onto a tractor by a typical farmer or agricultural mechanic in a time span of 8 hours or less with basic hand tools
- Universal fit: Must be compatible with John Deere 6000, 7000, 8000 series and 6R, 7R, 8R series tractors as well as CaseIH, Optum, Puma, Maxxum series tractors
- Water/Dust Resistance: Must adhere to at least and 'Ingress Protection' standard of IP65, to ensure safety from dust and accidental water damage

Concept Generation

The design problem encompasses two main sub-sections. In the preliminary stages, the team had broken the design into the navigation and steering components of the project.

Navigation

A summary of the design concepts for the navigation are provided in the table below:

Concept	Description
Machine Vision	Camera-based system used to capture real time footage of the crop rows as the tractor navigates through the field. Image processing algorithms would be implemented to detect the crop row.
Laser Sensor	Several laser sensors would be mounted at the front of the tractor and pointed downward towards the crop lines in order to detect the presence of an obstacle that needs to be avoided.
Encoder Mapping	Placement of encoders on the wheels of tractor in order to map the field. This solution would require the tractor to drive through the field once after the crops have been planted to store the coordinates of the field relative to a home position.

Table 1 The table summarizes the concepts generated for the navigation system.

Steering

A summary of the design concepts for the steering are provided in the table below:

Concept	Description
Belt Drive on Steering Column	The steering column would be directly actuated using a belt drive system with a rotary encoder for positional feedback. The motor would turn the column according to the feedback from the sensor.

Friction Wheel Drive System	A friction drive wheel controlled by a motor would run tangential to the tractor's steering wheel. Using frictional force, the drive wheel would physically turn the steering wheel.
Hydraulic Cylinder Valve	A solenoid valve would be connected to the tractor's existing system in order control the hydraulic cylinder of the tractor which would then adjust the steering.
CAN bus	The CAN controller can take control signals from the navigation system. This would directly control the steering of the tractor.

Table 2 The table summarizes the steering concepts generated for the system.

Concept Evaluation and selection

The concepts generated for each subsystem were evaluated using a Pugh Matrix. The evaluation of the concepts is provided below.

Navigation

The main objectives considered were compatibility with other tractors, ease of installation, cost effective, robustness, safety, and reliability. The concept selected was machine vision. This solution is robust; image processing algorithms can be easily modified to fit our needs. Moreover, machine vision allows for feature recognition or colour thresholding which can be used to ensure that only crops are identified as targets, while all other objects are ignored. A specialized camera is not required, as a webcam would provide enough contrast, and further enhancements can be done via image processing. This makes this solution cost effective.

Steering

The relevant objectives considered were ease of installation, cost friendliness, robustness, safety and reliability. The selected concept was the friction drive system. This solution is easy to install as it is

not invasive. The solution is safe; pinching would not be an issue as the contact force required to steer the tractor is very low. A spring-loaded hinge would allow the motor of the system to be easily pushed away. Note that although this was the selected concept to be designed, the team recognized that for tractors already equipped with steering systems such as CAN bus or hydraulic systems, our navigation component could easily be modified to interface with these steering components and would not require an additional one.

Software

Algorithm

The design problem required a robust algorithm that could be used with real-time footage to steer a tractor for post-emergence row crop. Fundamentally, this was a data-processing problem and the performance (measured by efficiency and accuracy) of the algorithm were the key design components. The performance factors are further discussed in the 'Algorithm Efficiency' and 'Software Validation' sections of this report. The algorithm was used to process each frame of the video footage and accurately detect the row of corn that the tractor is following as shown in Figure 1. The selected final design of the algorithm implements colour-based thresholding. After testing the performance of the algorithm for accuracy and efficiency, it was determined that colour-based thresholding met the requirements of the design problem.

Figure 1 Row of corn detected by color-based thresholding algorithm that produces a vertical histogram and finds the column with the greatest number of pixels.

In the algorithm, each frame of the video footage is converted into HSV colour space, which is an alternative to RGB. There is a higher contrast in the HSV images than in the RGB. This observation is evident in the comparison between Figure 2 and Figure 3. For this reason, thresholding is easier and more accurate in HSV.

Figure 2 Original frame containing the image in the RGB color space. It is clear that the brown pixels in the background are difficult to differentiate from the green pixels of the corn.

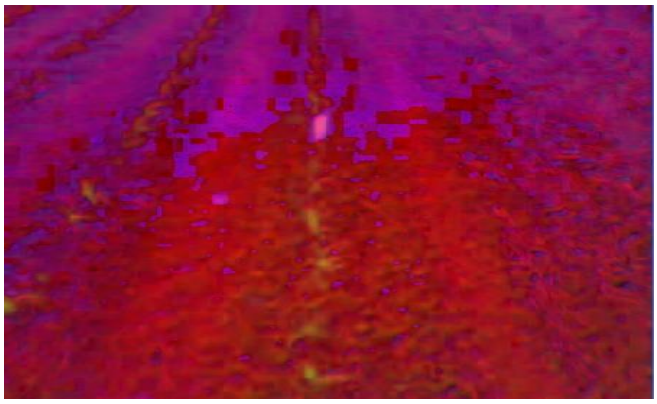

Figure 3 The image is converted into HSV colors space. The green pixels of the row of corn are easy to differentiate from the pixels in the background.

Thresholding in HSV colour-space is based on hue, saturation and value. The hue is determined by the absolute or pure colour. The saturation is determined by how much white the colour contains [4]. A pure or solid colour is considered to be fully saturated. Finally, value is determined by the brightness of the colour. A visual representation of colours in the HSV colour space is shown in Figure 4.

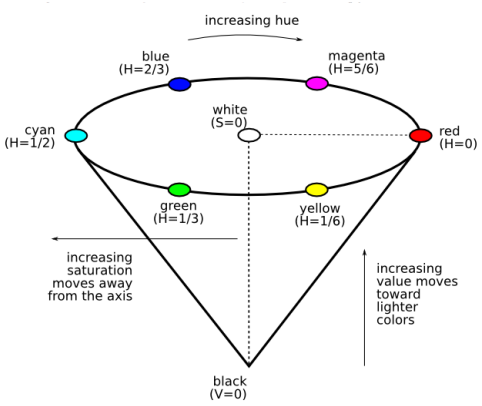

Figure 4 A visual representation of colors in the HSV color space. The image was taken from: http://infohost.nmt.edu/tcc/help/pubs/colortheory/web/hsv.html.

The threshold values for the algorithm were determined by testing several consecutive images of row-crop taken during the day. The resulting values are summarized in Table 3. The testing interface used to determine these values is shown in Figure 5. The interface is used to dynamically adjust the images based on the threshold values indicated by the sliders. Essentially, the sliders constrain the hue, saturation and value within a minimum and maximum value. The images in the figure show the before and after thresholding results of one of the test files. Evidently, the row of corn was distinguished from the background using the threshold values that were determined.

Parameter	**Value**
Hue Max	24
Hue Min	6
Saturation Max	226
Saturation Min	197
Value Max	184
Value Min	68

Table 3 The table summarizes the threshold values that would be used during the day time.

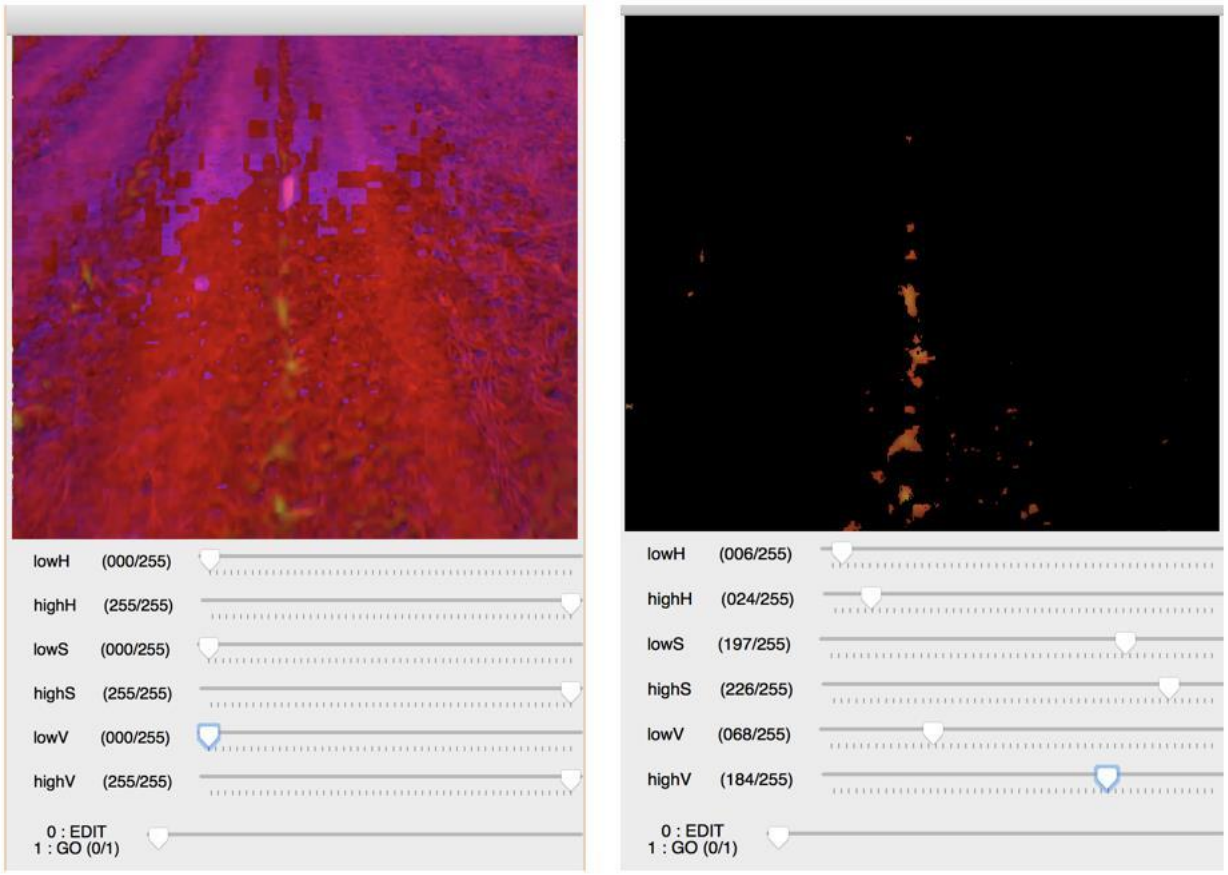

Figure 5 The above images show the Testing Interface for the project used for finding threshold values. The image on the left shows the original image without applying thresholding. The image on the right shows the image after thresholding. It is clear that only the row of corn is visible and easily detected.

After thresholding the image, a binary image is generated based on the pixels that are still visible after the threshold is applied. These visible pixels are considered to be the 'relevant pixels' because they represent the pixels of the corn. A vertical histogram of the binary image is computed to find the total number of 'relevant pixels' in each column of the image. Next, the column that contains the greatest number of 'relevant pixels' is determined to be the row of corn that should match the center of the tractor. The error is computed as the difference between the x-axis coordinates of the detected row of corn and the actual center of the tractor which is then sent to the controller to adjust the steering.

Efficiency

Different algorithms were implemented and tested for efficiency. The results are summarized in Table 4. The test concluded that colour-based thresholding using the InRange method was the most efficient and hence was selected as the algorithm to be implemented in the final design.

Algorithm	Description	Execution Time (s)
Python "Threshold Method"	Color thresholding was done on a HSV image.	0.047
Canny Edge Detection	Finds and highlights target blobs.	0.078
Python "InRange" Method	Color thresholding was done on a HSV image (different python method than threshold).	0.039

Table 4 Summary of different algorithms tested for execution time

The average reaction time of a male to a visual stimulus is approximately 0.25 s [5]. This is how often a tractor operator would be able to look at the GUI, check for any errors and perform the corrective measures needed. Therefore, to meet our objective of reliability, the algorithm should have an execution time of no more than 0.25s. The following timing tests were performed using a computer with a processor of 2.4GHz. First, the execution time of just the image processing algorithm was measured using the time() function included in Python. Time stamps were placed at the beginning and end of the code fragment corresponding to the image processing algorithm, excluding any display functions. The result is summarized in Table 5. Although the performance of the algorithm was validated in this initial test, the results were not representative of our program as a whole. The program not only has to process an image, but it also has to display it back on the screen, which would increase the execution time. A second test was performed, this time with time stamps placed at the beginning and end of the source

code, including the display function. The results obtained were below 0.25 s and are summarized in Table 5.

	Execution Time
Isolated Image processing algorithm	0.0009 seconds
Image processing algorithm and display functions	0.0369seconds

Table 5 Table summarizing the results of the algorithm's execution time

After the timing tests performed, it is clear that the program meets our timing constraint for the processor used. For this reason, we will continue to test our software prototype on this device.

Graphical User interface and analysis

The final user interface will contain 4 screens, a Home screen, a Basic Settings screen, a Saved Settings screen and an Advanced Settings screen. As the program starts, the user is taken directly to the Home screen, shown in Figure 6. This screen has a power button, a settings button and a go button which activates the serial communication of the error signal to the steering controller.

Figure 6 Snapshot of the Home screen of the GUI. Buttons top to bottom: Power, Settings, Go

By selecting the settings button, the user is taken to the "Basic Settings" screen. Here, the user is able to select between 4 different weather conditions including sunny, cloudy, evening and rainy, see Figure 7. Each selection comes with pre-set threshold for the respective condition. The purpose is to reduce the amount of manual thresholding done by the user. Due to limited data collected, only the values for the sunny condition have been determined for our prototype, however manual thresholding is still available for all remaining conditions.

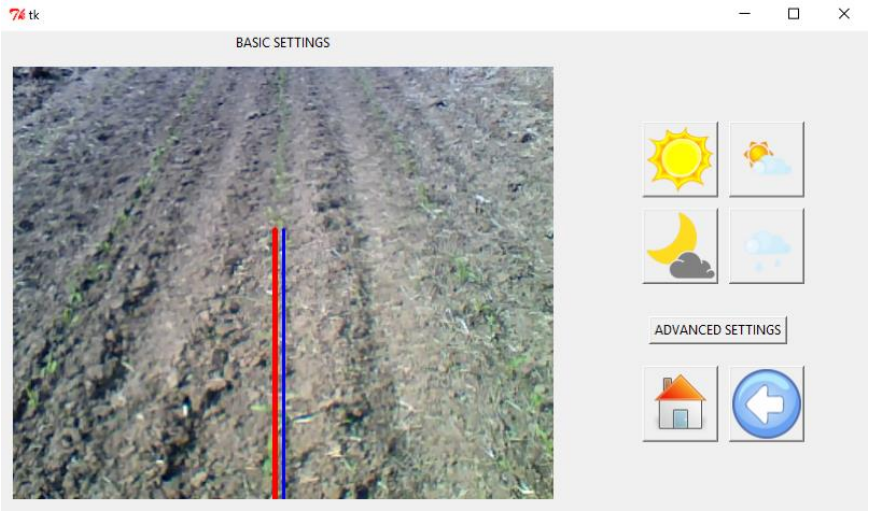

Figure 7 Snapshot of the Basic Settings screen. Buttons top to bottom: Preset settings (sunny, cloudy, evening, rainy), advanced settings, home and back

In addition, the program also has a saved settings screen (Figure 8) and an advanced settings screen (Figure 13). The user can access the "saved settings" screen by selecting "advanced settings" on the Basic settings screen seen above. To make a customized setting, the user can select "Make new" on the "saved settings" screen seen below. These two screens work together so the user can set and save their own threshold values for situations where the "basic" settings are not sufficient. The sliders can be adjusted to dynamically change aspects such as hue, saturation and value. Once the user is satisfied, these values can be saved and will appear in the saved settings screen on a list to choose from.

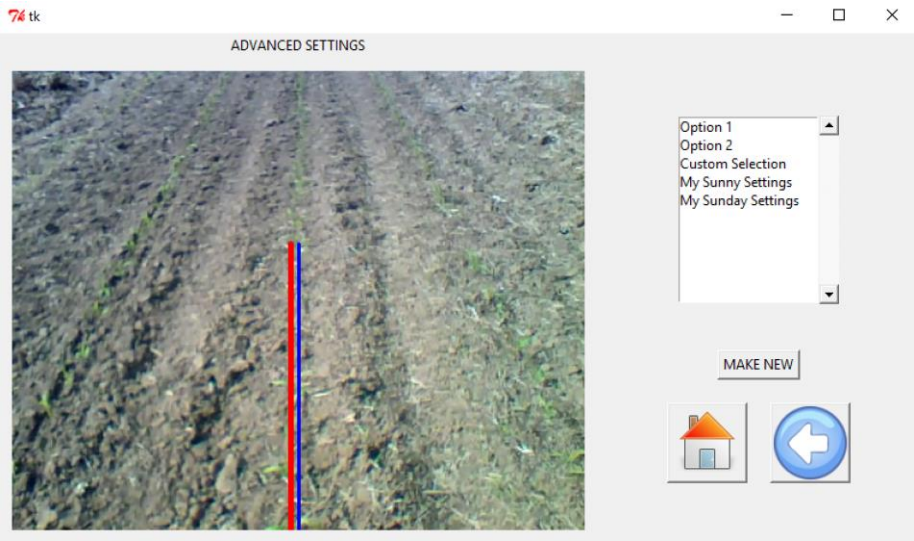

Figure 8 Snapshot of the Saved Settings screen. Widgets top to bottom: List of custom settings, make new, home and back

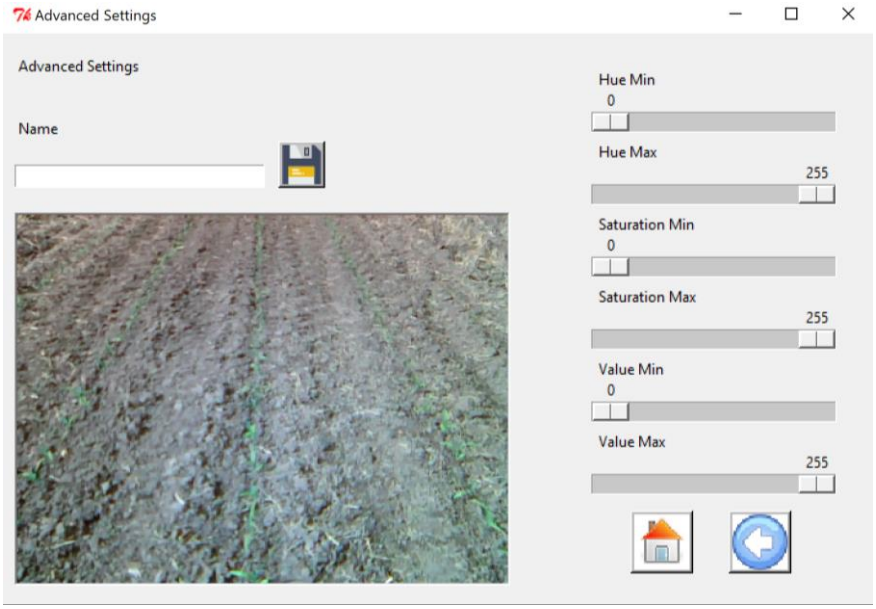

Figure 9 Snapshot of the Advanced Settings screen. Widgets top to bottom: saturation, hue and value sliders, home and back

A flow diagram has been provided below in Figure 14 to give an understanding of how the different screens of the GUI will interact with each other and use the algorithm.

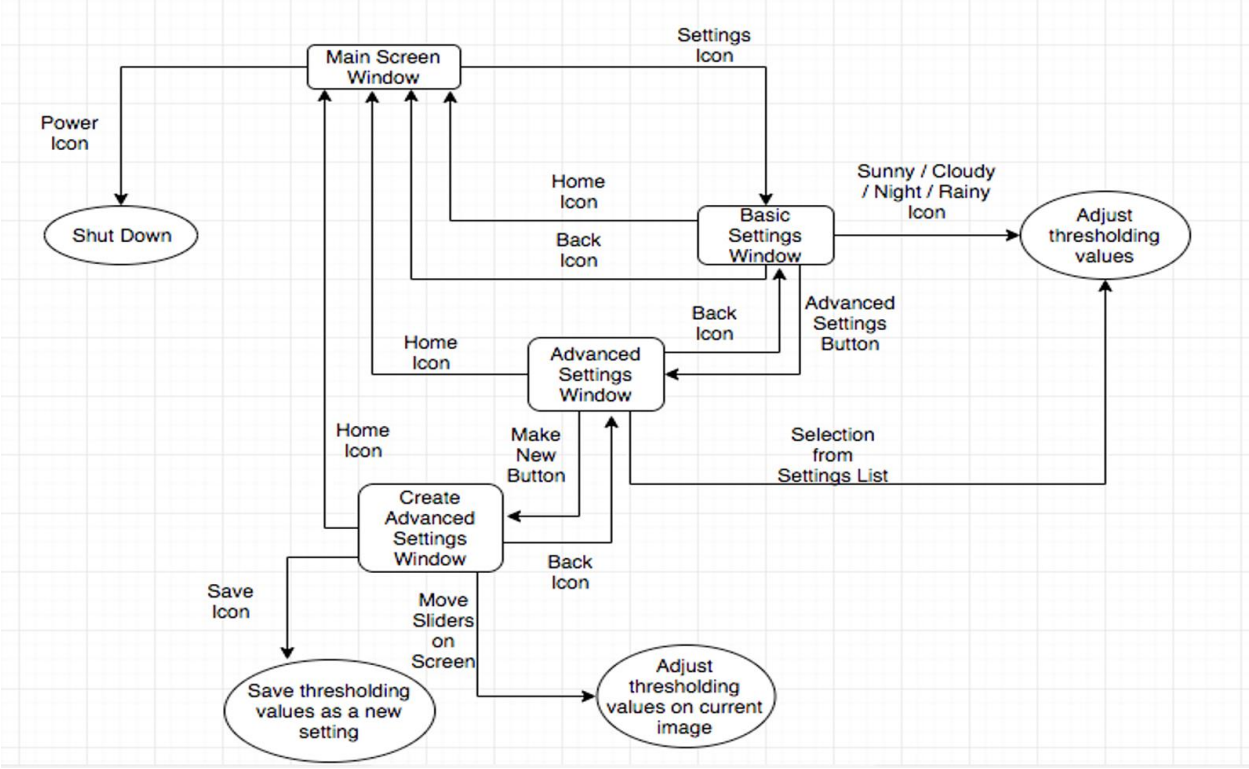

Figure 10 Flow chart of the GUI showing the interactive process the program

The GUI design was validated qualitatively. In order to satisfy our ease of use objective, the layout of the screens and all the widgets within them was made ergonomic and easy to follow. According to Microsoft, the minimum recommended button size for a touch screen with a display of 150 pixels per inch is 40 pixels by 40 pixels [6]. Based on this constraint, all of our widgets were designed to meet this requirement. In addition to button sizing and spacing, labels and/or images were added to make it more intuitive. For the layout, buttons were intentionally placed on the right-hand side of all the screens in order to make them easily accessible.

Prototyping

Initially, prototyping was conducted in MATLAB using the image processing toolbox because it was a familiar platform to verify that computer vision was a valid solution to the design problem. The initial algorithm was implemented in MATLAB using adaptive color-based thresholding to detect the row crop. In this implementation, a red box indicated the center line with respect to the center of the tractor. It also constrained the part of the image that is of interest to the algorithm. A dashed blue is also drawn to show the location of the detected row crop. Some of the sample outputs of the MATLAB prototype is shown in Figure 11.

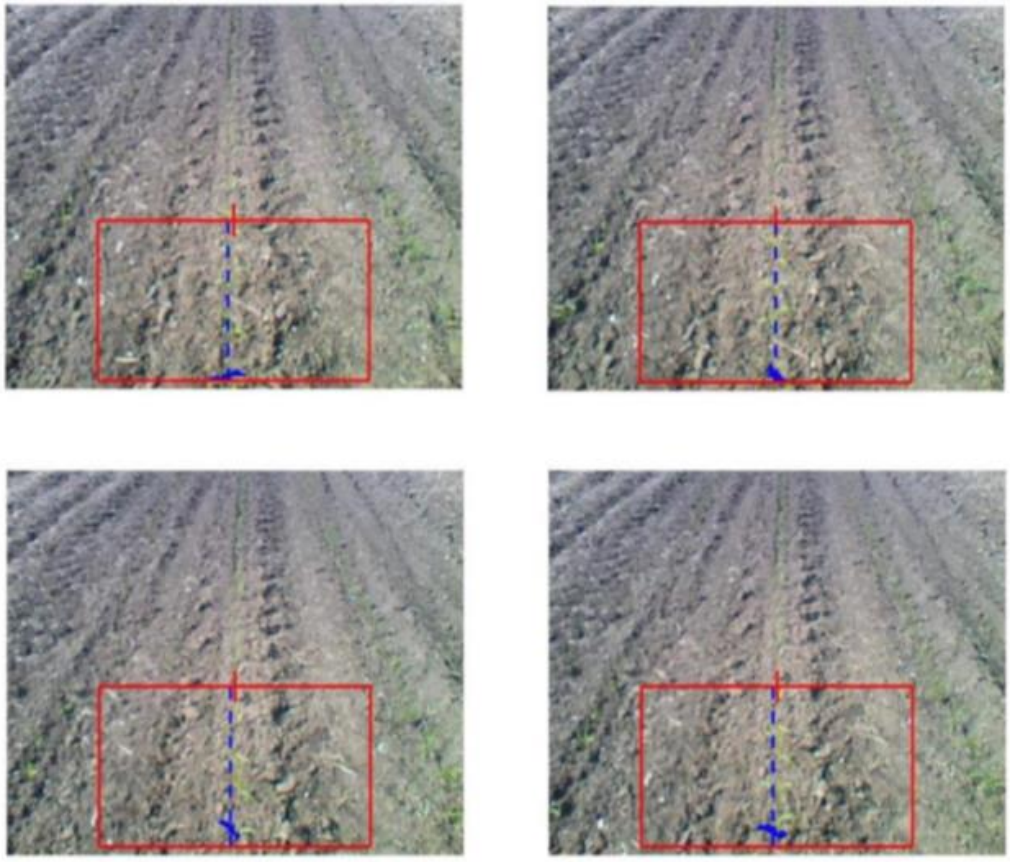

Figure 11 MATLAB adaptive thresholding prototype

Since the initial MATLAB prototype exhibited some errors due to noise and only processed about 20 frames/second, the team experimented with different image pre-processing techniques for contrast enhancement including histogram equalization and masking as shown in Figure 12. It was noted that these techniques were computationally heavy for a real-time application, causing the algorithm to slow down. Furthermore, it was unnecessary because large errors could be filtered out using averaging. However, the MATLAB prototype was able to validate that using computer vision to detect rows of corn to steer the tractor was a feasible concept.

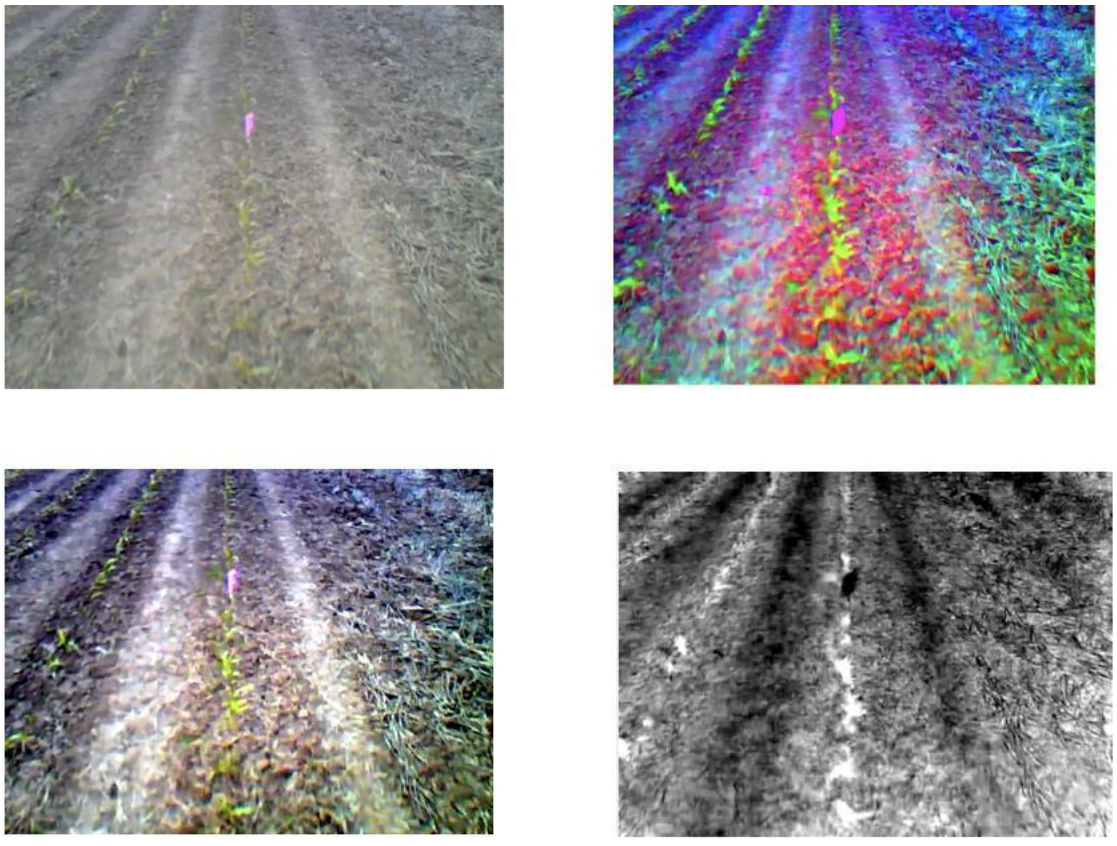

Figure 12 The top left is the original image. The two colored images resulted after using contrast enhancement techniques. Finally, the black and white image is a masked image.

This algorithm was then implemented in Python with OpenCV along with other algorithms discussed in the 'Algorithm' and 'Efficiency of Algorithm' sections of this report to better understand how to optimize the algorithm based on efficiency and accuracy. As previously discussed, it was

concluded that using simple color-thresholding in the HSV colour-space was the most effective approach to the design problem due to results concluded in the 'Efficiency of Algorithm' section of this report. The final prototype was implemented on a Raspberry Pi using Python with OpenCV for the image processing and TkInter for the GUI. Python was used because it comes pre-installed on the Raspberry Pi and it is a quick and easy prototyping language. The library for the GUI was TkInter because it comes pre-installed with Python and doesn't take extra memory resources on the Raspberry Pi. Finally, the image processing algorithm for the prototype was implemented with the OpenCV library because it is well documented, open source and compatible with the Raspbian operating system. The final design of the user interface for the prototype is discussed in the 'Graphical User Interface and Analysis' section of the report.

Prototype Cost

The following table summarizes the costs incurred by the team in validating the software component of this system.

Software Prototyping costs		
Item	Quantity	Cost (CAD)
Raspberry Pi 7" LCD display	1	$89.20
Raspberry Pi 3 Model B Motherboard	1	$53.33
TOTAL		$142.53

Table 6 Prototyping cost of the software component of our project

Software Final Design

The final design implements the algorithm described in the 'Algorithm' section of the report. The software for the final design would be implemented using the C++ programming language because it is efficient and it is compatible with the OpenCV library. The final design would also replace the Raspberry Pi with another controller that may be custom designed or off the shelf to make the product cost effective. The design and selection of the board is out of the scope of this project. Furthermore, the user interface would be modified to improve the aesthetics and a database would be implemented to store the user threshold settings in the backend of the software. This was not done in the prototype because it would require more memory resources than offered by the Raspberry Pi and was not needed to verify the functionality of the system.

Steering Actuator

Concept Introduction

The proposed steering mechanism is able to steer the tractor by actuating the tractor's existing steering wheel. The steering mechanism consists of a DC motor, a foam friction wheel attached to the DC motor's shaft, and bracketry. The bracketry allows the foam wheel to contact the tractor's steering wheel. When power is applied to the DC motor, torque is transmitted to the steering wheel through the foam wheel by friction. A view of the steering mechanism may be viewed in Figure 13.

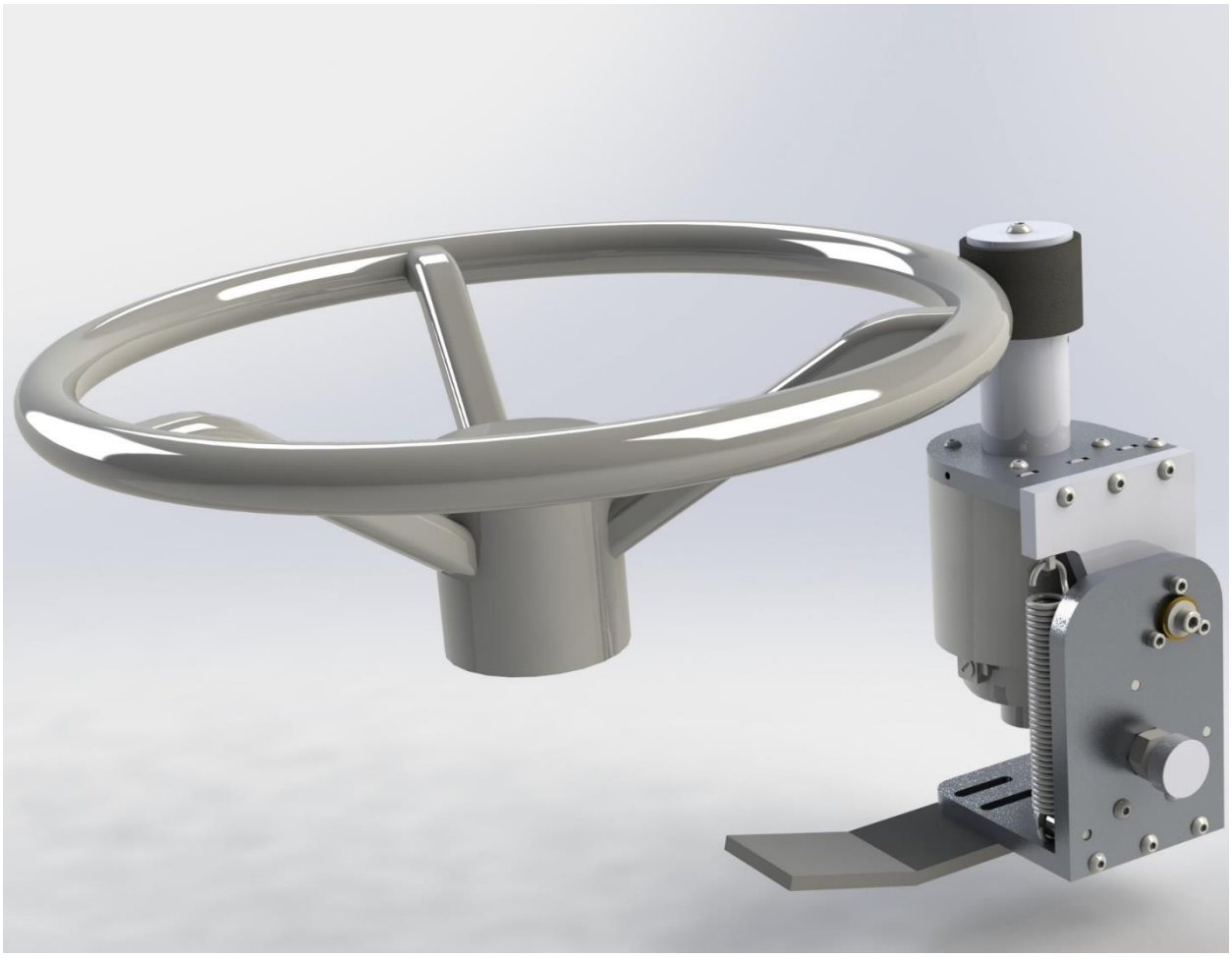

Figure 13: CAD render of the “operation” state of the steering actuator assembly

The bracketry serves two purposes and is designed as two separate assemblies. The first purpose of the bracketry is to allow for attachment to any tractor, serving as a foundation to the remainder of the mechanism. Tractor steering columns vary by manufacturer and model series, each series requiring its own unique foundation bracket design. This project does not aim to cover the ~30 different foundation brackets required to cover every popular model of tractor. Instead the group will utilize a premade bracket by Trimble installed on a John Deere 7410. The foundation bracket and its orientation relative to the tractor’s steering wheel may be viewed in Figure 14.

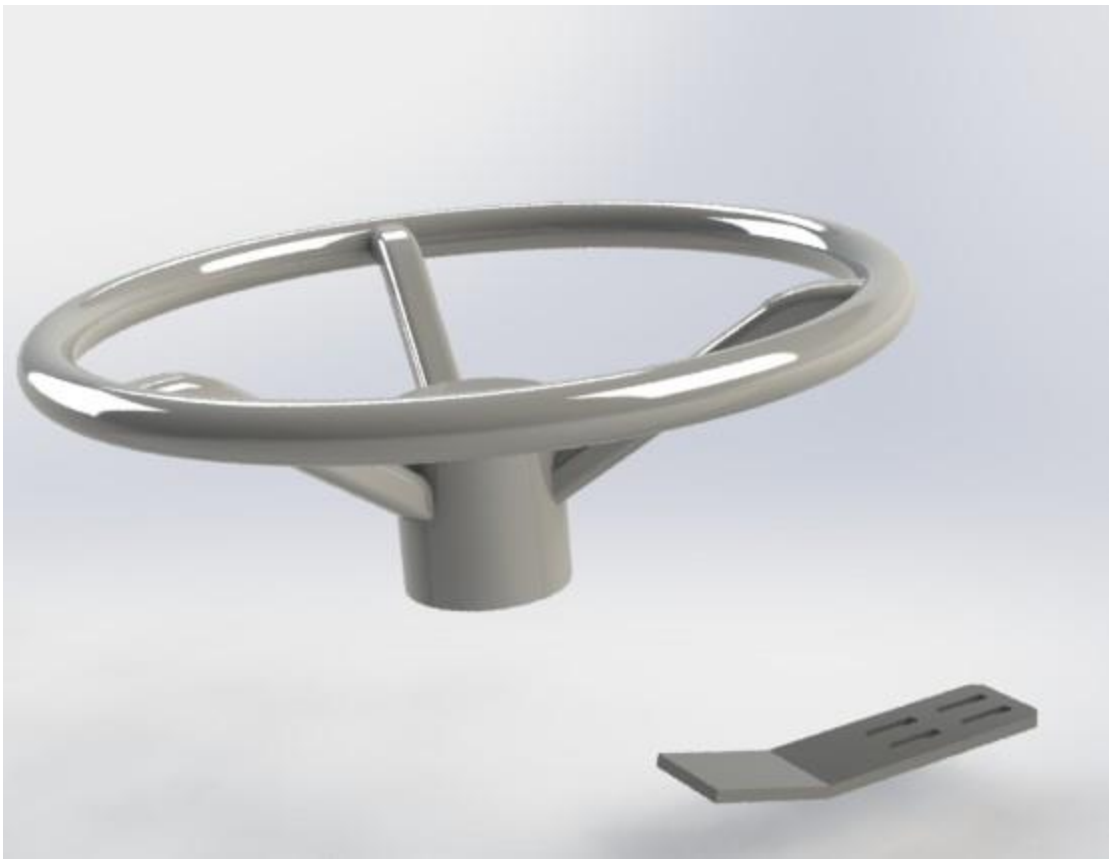

Figure 14: CAD render of the foundation bracket of the steering actuator assembly

The second purpose of the bracketry is to allow the motor to pivot between two states. The first state, the operating state, which may be viewed in Figure 13, maintains contact between the foam friction wheel and the steering wheel. The second state, the storage state, which may be viewed in Figure 15, locks the foam wheel away from the steering wheel.

In operating mode, a spring maintains contact and pressure between the steering wheel and the foam wheel. This pressure is constrained by two values. The pressure must be sufficient to overcome the steering wheel's frictional losses, but small enough to not harm an operator's fingers caught between the steering wheel and the soft foam friction wheel. The exact spring tension required will need to be derived experimentally.

To move the motor to the storage state, the operator can simply pivot the mechanism out of the way where it will lock into place through use of a spring-loaded pin. To bring the motor back into operating mode the operator can pull on a readily accessible round button.

The pivoting bracketry is of a singular design, able to be mounted to any foundation bracket.

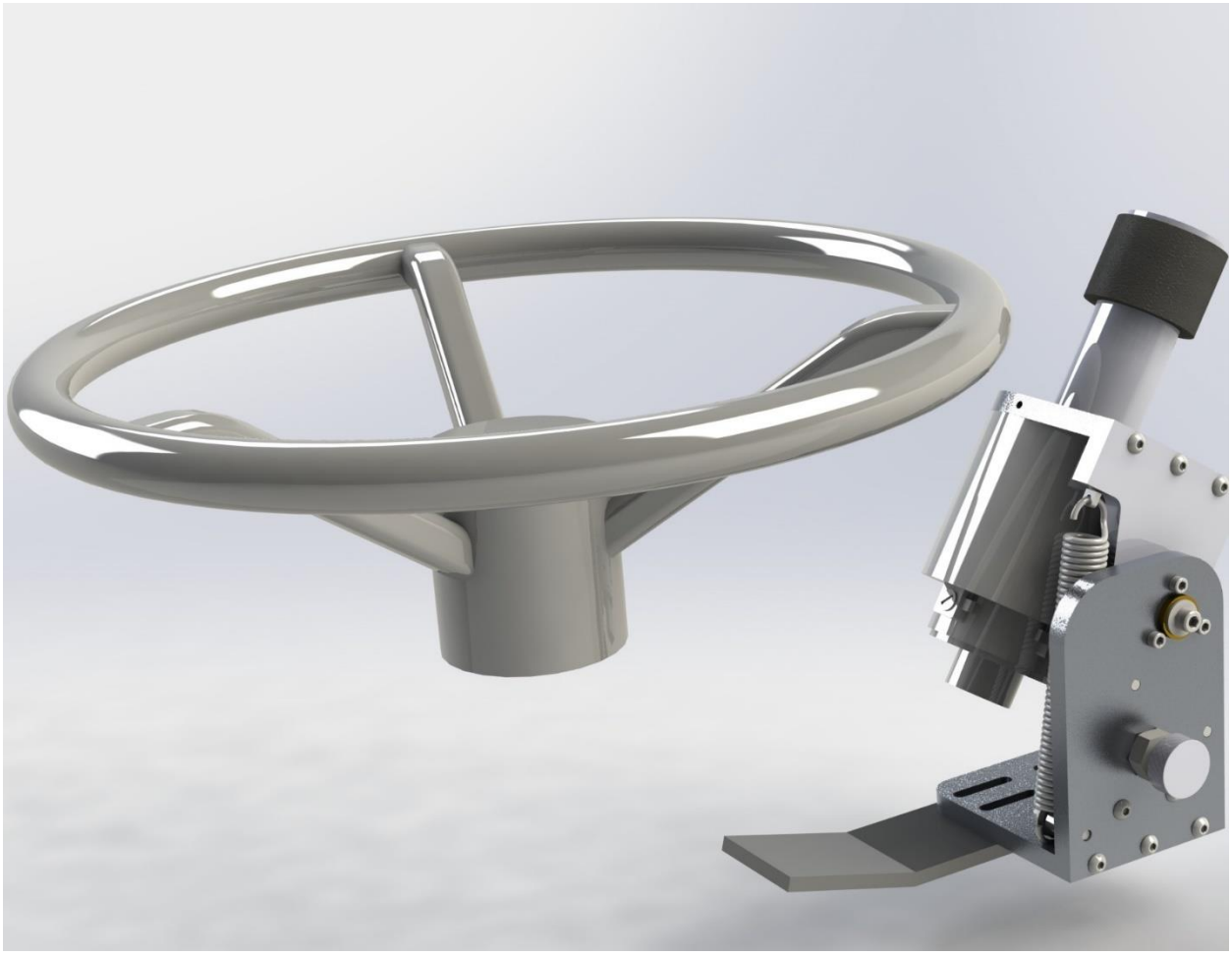

Figure 15: CAD render of the “storage” state of the steering actuator assembly

The DC drive motor is equipped with an internal encoder. If the steering mechanism is active and an operator physically grabs and turns the steering wheel, a microcontroller will sense a disparity between desired position and actual position, causing the system to dis-engage from automatic steering mode and into standby mode.

Calculation – Spring Force

The steering system's functionality is highly dependent on the force provided by the extension spring. The spring force was carefully chosen as a factor of multiple variables.

The friction wheel to steering wheel interface is impractical to shield, as adding any shielding would add a dangerous possible pinch point between the steering wheel and shield. To prevent injury of an operator's hand caught between the steering wheel and friction wheel, the friction wheel is able to pivot out of the way with a limited amount of force. This contact force was reasonably designed to be no greater than 10 lbs., setting an upper limit on the spring force. If an operator's hand is caught between the friction wheel and steering wheel, the operator would have no injury or pain. To further prevent the chance of injury the friction wheel was chosen to be a soft pliable polymer with no sharp edges; a urethane with a soft durometer rating.

The upper force to be exerted by the spring can be calculated using the maximum contact force of 10lb through a sum of moments calculation as shown below. Required dimensions were pulled from the prototype CAD model as shown in

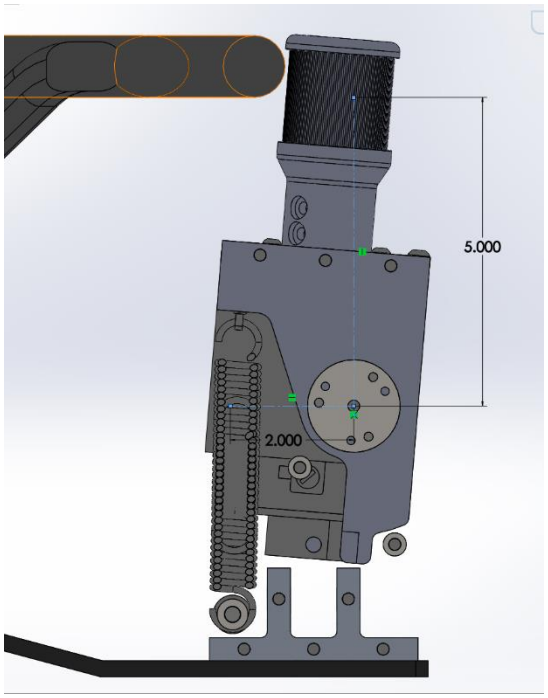

Figure 16.

$$\sum Momentpivot = 0 = -10lb * 5in + Forcespring\_max * 2 * in$$

(Note: CCW +'ve)

$$Forcespring\_max = 10lb * 5in 2in = 25\ lb$$

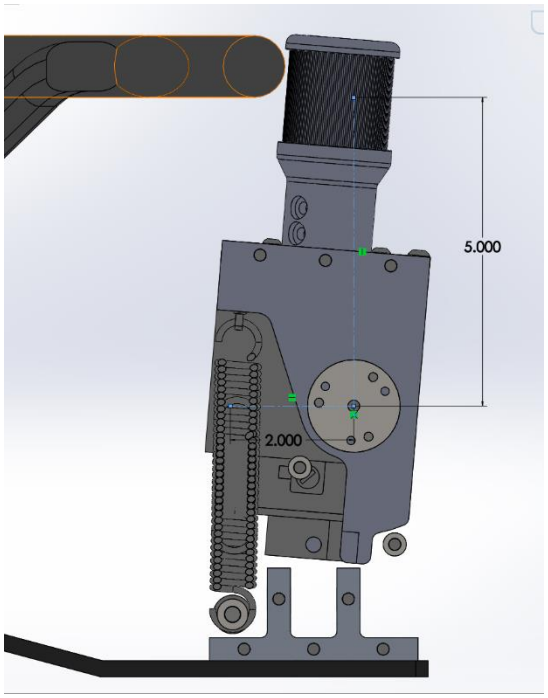

Figure 16 Section view of steering bracketry, showing dimensions for sum of moments equation. The image does not show accurately where the steering wheel contacts the friction wheel. The steering wheel should be shown to contact the center of the friction wheel as dimensioned. All dimensions are in inches.

Spring Selection

Tilting the servo motor throughout its range requires a spring length that varies between 4.1 and 5 inches as determined by the CAD model. The spring should remain under tension when in the minimum position, so the length of the spring at rest should be less than 4.1 inches. No springs with the available range and spring force constant were readily available. A custom ordered spring or redesign of the bracketry may be desired for the final product. For the prototype, a spring with a greater (~1.5x) than required spring force constant was selected from McMaster Carr. This spring will demonstrate all the functionality of the steering system. The prototype spring will not limit the force that could be imposed to a user's hand to 10 lb, instead limiting it to 15 lb. The team feels comfortable that this force is low enough to not cause any injury and would at most cause mild pain. This spring will work to further develop the system.

Calculation - Friction Wheel Material

A contact force of 10 lb limits the amount of torque that can be transmitted from the servo motor to the steering wheel. A suitable friction wheel material needed to be chosen to ensure that enough torque can be transmitted in order to turn the tractor's steering wheel. The torque required to turn the test tractor's steering wheel was measured using a digital torque gauge, a Mastercraft Torque Adapter [7] and found to be 17 lb-in. Note that this is the torque required to turn the steering wheel itself, not the torque required to spin the friction wheel. Although the measurement was test tractor specific, the test tractor's steering wheel is typical to that of most tractors, being made of a smooth plastic and being 16" in its outer diameter. The frictional force required to turn the steering wheel is calculated below.

$$Torque = Force * Distance$$

$$Torque_{steering\_wheel} * Factors_{safety} = Force_{static\_friction} * Radius_{steering\_wheel}$$

$$Torque_{steering\_wheel} * Factors_{safety} = Force_{normal} * coefficient_{friction} * Radius_{steering\_wheel}$$

The torque required to steer the tractor was measured to be 17 lb-in. A safety factor of 2 seems appropriate. The normal force is limited to a max of 10. The radius of the steering wheel is 8 inches.

$$17\ \text{lbin} * 2 = 10lb * coefficient_{friction} * 8in$$

$$17\ \text{lbin} * 2 = 10lb * coefficient_{friction} * 8in$$

$$coefficient_{friction} = \frac{17\ lbin * 2}{10lb * 8in} = 0.425$$

The friction wheel material was chosen to be a polymer with a durometer of 95A. The friction co-efficient between a polymer with a durometer of 95A and smooth plastic (steering wheel) is 0.5 based on Figure 13. An example of an everyday object with a durometer of 95A is a garden hose [8]

which also reduces the chance of operator injury in the case of a hand caught between the friction wheel and steering wheel.

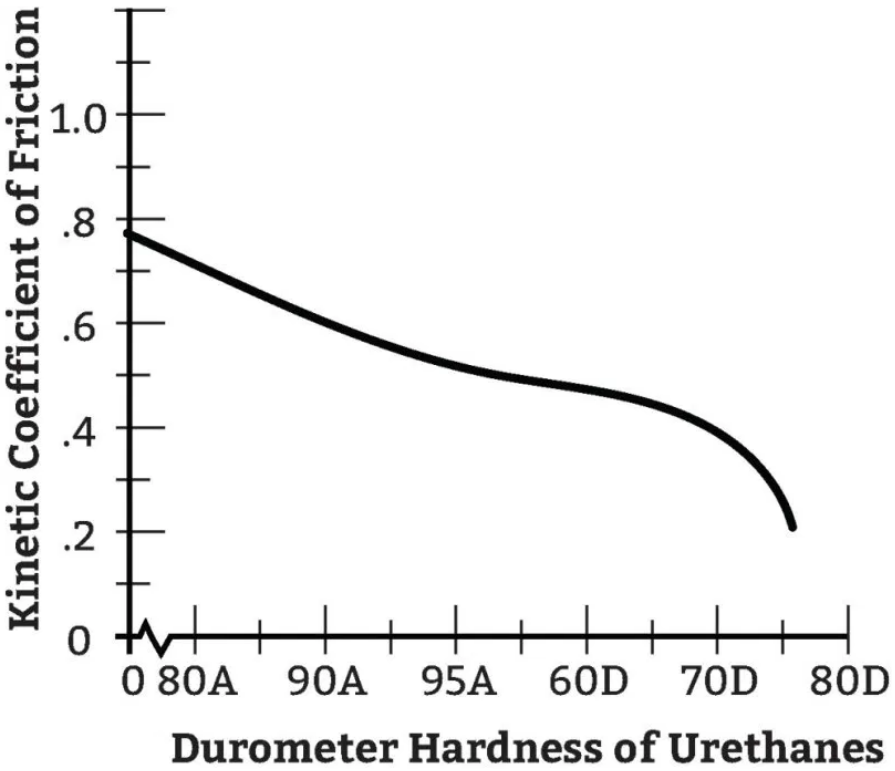

Figure 17: friction coefficients of urethanes against smooth plastic https://gallaghercorp.com/design-guide/polyurethane-coefficient-friction/

Calculation - Required Motor Torque

The minimum torque required at the motor can be calculated from the ratio difference between the friction wheel and steering wheel as well as the torque required to turn the steering wheel.

$$Torque_{motor\ min} = Torque_{steering\_wheel} * \frac{Diameter_{friction\ wheel}}{Diameter_{steering\ wheel}} * Factor_{safety}$$

The diameter of the steering wheel was measured to be 16 inches. The diameter of the friction wheel was designed to be 2 inches. The torque required to turn the steering wheel was measured to be 17 lb-in.

$$Torque_{motor\ min} = 17lbin * \frac{2\ in}{16\ in} * 2 = 4.25\ lbin$$

The torque of the motor used for the prototype could not be found since the motor is obsolete and the manufacturer does not provide specifications for obsolete products. The torque would be difficult to measure using available resources. As a substitution to the actual specification, current production model dc servo motors of the same basic size dimensions are offered by the manufacturer in torques ranging from 2.6 lb-in to 11 lb-in [9]. If we ignore the factor of safety, then the motor used in the prototype almost certainly has enough torque. This remains to be tested. The friction wheel could be made slightly smaller which reduces the required motor torque, this modification to the friction wheel design will be made only if necessary.

The torque of the motor should be low enough so that the motor stall, rather than slips, when the steering wheel is manually blocked by a user. This max limit on torque is calculated below.

$$Torque_{motor\ max} = Force_{contact} * coefficient_{friction} * Radius_{friction\ wheel}$$

$$Torque_{motor\ max} = 10lb * 0.5 * 1in = 5\ lb * in$$

As stated above; the prototype motor's torque specifications are unknown. It is very easy to limit the motor's torque through limiting the electrical power provided to the motor. The extent of required motor limiting will be fine-tuned experimentally once the system's prototype is completed and installed onto the test tractor.

Refinement of Design

The original prototype manufacturing intent was for the major parts to be machined out of aluminum plate. The entire assembly was designed with this manufacturing method in mind. After

generating drawings, but prior to machining, the major parts were 3D printed out of PLA (polylactic acid) to ensure size and fit. After several small iterations of tweaking dimensions to ensure proper fit, the realization was reached that the 3D printed components were in fact quite robust and close to being of high enough strength to be utilized as a substitute to machined aluminum parts. A major round of re-design was begun to allow the parts to be solely 3D printed, stiffening the assembly and utilizing embedded fasteners instead of tapped holes. In addition, plastic materials suitable for 3D printing were researched and a more suitable prototyping plastic was discovered and purchased. PETG (Polyethylene Terephthalate – Glycol) was chosen due to having a larger elastic region (flexing before breaking) and having a higher elastic limit.

Cost Analysis

The following table summarizes the costs incurred by the team in validating the steering component of this system.

Steering Prototyping Costs		
Item	**Quantity**	**Cost (CAD)**
PETG filament (3D printer material)	1 kg	28.95
Spring	1 (pack of 6)	10.61
Spring Loaded Plunger	1	14.14
Spring Hanger	1	4.17
DC Servo Motor	1	300 (estimated value)
TOTAL		357.87

Table 7 Summary of cost for steering components

Human Machine Interface

Concept Introduction

The system requires a human machine interface (HMI) to allow the operator to monitor system status and make operational changes. The GUI of the HMI is discussed elsewhere in this report. The physical characteristics of the HMI will be discussed here. However, the processor selection for the HMI and control system are not the main focus of the design. Supplementary information on how these parts were selected can be found in Appendix B and Appendix C respectively.

Screen Selection

Our HMI solution should be similar to that what is used in existing GPS autosteer systems. Customers are familiar and accepting of these systems. Data was compiled on existing systems and itemized in the chart below

	Screen Size	Colour Screen	Touchscreen	Lightbar
Trimble EX-Guide 250	4.3”	✓	×	LED
Trimble CFX-750	8”	✓	✓	LED
Trimble GFX-750	8”	✓	✓	LED
Trimble TMX-2050	12.1”	✓	✓	LED
Ag Leader InCommand 1200	12.1”	✓	✓	LED
Ag Leader InComand 800	8.4”	✓	✓	LED

AgLeader Compass	7”	✓	✓	LED
John Deere 4640	??	✓	✓	On Screen

Table 8 Data for monitors of various existing systems.

References for the table: (Ag Leader, 2017) (John Deere, 2017) (Trimble, 2017) (Trimble, 2017) (Trimble, 2017)

An average screen is about 8” and is a colour touch screen. Touchscreens technology options consist of capacitive and resistive. Capacitive touch screens sense conductivity changes caused by the skin. Resistive touch screens sense pressure applied to the screen. Since this system is expected to operate in conditions where fingers may be covered in dirt and dust, a resistive touch screen is a better option.

An ~8” resistive touch screen was not readily available for prototyping. A 7” resistive touchscreen was chosen based on being commonly available, especially for use with Raspberry Pi (more mentioned on this later). This size is smaller than desired but still lies within the size range of available GPS autosteer HMI screens.

Motor Driver Selection

The steering actuator, as previously discussed, is a large dc motor of unknown specifications. An H bridge module was chosen to drive the motor. A quick stall current test showed that the motor can draw approximately 5 amps at 12V. Powering the system directly from the tractor’s 12V system seemed logical, any voltage conversion would require extra components. A commonly available hobby level H bridge driver is the IBT-2, featuring a BTS 7960 H bridge IC that is capable of driving up to 43 amps at 12V. This rating is an unenclosed rating. Since our application required the module to be enclosed, we used our engineering judgement to justify some of the over-specification. A group member had

previous experience with this module and had previously purchased one, making it the ideal choice for prototyping.

Voltage Regulator

The Pi, Arduino, H bridge logic, and touch screen all operate on 5 VDC. The input to the monitor should be 12 VDC from the tractor. A voltage regulator was required to step down the voltage. The tractor's 12 VDC is very similar to automotive 12 VDC, meaning that the voltage can be anywhere from 10-14 VDC and see large voltage spikes and high frequency noise. A common voltage regulator IC is not appropriate for this task. An automotive grade voltage regulator module was chosen based on its availability and current specifications, capable of providing the up to 3 amps total required by the Pi and touch screen. The Arduino's and H bridge logic's power requirements are unknown, but known to be negligible in comparison to the Pi and touchscreen.

Mounting Connection for Monitor

The HMI needs to be mounted to the tractor in some manner. Tractor cabins often have provisions for mounting third party HMI's such as those used by corn planters, sprayers, etc. The provisions are generally bolt holes. A company called RAM Mount manufactures adjustable mounting solutions that are widely used in the agricultural industry. To make our system compatible with RAM Mounts, a 1.5" ball was added to the rear of the display.

Enclosure Construction

The enclosure required complex features and was chosen to be 3d printed due to the minimal manufacturing effort required. A backing plate was laser cut from clear acrylic so that the inside of the enclosure was clearly visible for visual appeal during showcasing.

Camera Selection and Mounting

Prototype Structure

To prototype the design of the camera mounting system, the mounting and housing of the design was created in full scale. The focus for the assembly prototype was to demonstrate the user interaction with the sub-system and compatibility with the rest of the system. Due to time limitations and budget, constraints such as water resistance and robustness were not tested with this prototype. These constraints are not necessary to demonstrate the core functionality of the whole system, and therefore were not tested in this prototype design. The full assembly of the camera enclosure was printed using PLA plastic on a 3D printer. This allowed for rapid prototyping at a reduced cost. The assembly used all standard bolts and nuts that would be used in the final design. The full enclosure system prototype seen on Figure 21 demonstrates the feasibility of the fine-tuning and locking the camera angle mechanism as proposed by the final design. By loosening and tightening the bolts and nuts of the system, a user may easily change the camera direction. For prototyping of the actual camera, a cheaper alternative webcam was utilized. After initial testing demonstrated that a small, cheap webcam with built-in lens functioned with the algorithm, it was selected for the prototype.

Minimum Camera Resolution

In many machine vision applications, higher pixel resolution is an objective to maximize. However, in this application, higher resolution creates more pixel data and therefore causes a larger strain on the processor. To maximize the efficiency of the camera system, the minimum resolution that achieves adequate performance must be selected. This resolution can be found by applying the equation:

$$Sensor\ Resolution = Image\ Resolution = 2 \times \frac{Field\ of\ View\ (FOV)}{Smallest\ Feature}$$

The field of view can be generalized to be around 2.75m, obtained from row size, and the smallest feature to be around 15cm, from an estimated plant size. While technically the field of view is calculated as an area accounting for width and height, it is commonplace to generalize the calculation to a single dimension. This application suits the horizontal direction due to the nature of the arrangement of features across the crop rows.

$$Sensor\ Resolution = 2 \times \frac{2.75m}{15cm} = 366.667\ pixels$$

This value indicates that either in the horizontal or vertical direction, there must be at least 367 pixels to properly detect the crop. This value further confirms a 640x480 resolution camera to be adequate mathematically. It also demonstrates that a smaller resolution such as 480×360 would not be adequate for proper imaging, under these assumptions.

Lens Focal Length

A camera with an incorrectly selected lens will result in a greatly decreased image quality and may cause vignetting (causes the image to fade into its background without a definite border). For these reasons, it is very important to select the correct type of lens for its paired camera sensor and application. Certain lens design decisions, such as form factor, are trivial from an engineering perspective, however calculations are required for converging on the correct focal length.

$$Focal\ Length = \frac{Sensor\ Size\ \times Working\ Distance}{Field\ of\ View\ (FOV)}$$

Previous engineering analysis selected a specific CMOS sensor, the e2v EV76C560 1/1.8". It was selected for many quantitative and qualitative reasons, such as very high dynamic range, and global shutter. From the 1/1.8" size, it has sensor dimensions of 7.176mm x 5.319mm, which correlates to an 8.932mm measurement along the diagonal. That value is used in the above calculation. The field of view was stated in the previous section of this report as around 2.5m. The working distance is defined as the distance between the lens and the target object. This value will vary based on the angle of the camera mount, and for this reason, the camera mount angle and lens focal length are linearly dependent. In industry, typical calculations involve using estimated working distances to select a correct lens. Once the lens has been selected, new working distances threshold can be calculated. The lens is selected first, as it is typically the most difficult to change later from a design perspective. To reiterate, only estimated dimensions are required to generate the proper focal length, as differences in working distance from different tractor models will be correct with a properly designed mount.

$$Focal\ Length = \frac{8.932\ \text{m} \times \sqrt{(1.03\ \text{m})^2 + (1.05\ \text{m})^2}}{2.75\ \text{m}} = 4.77\ mm$$

As seen by the above calculations, the previously selected Azure-0420MM lens will be adequate, with a lens focal length of 4mm. Any deviations will be corrected by proper user positioning of the mount.

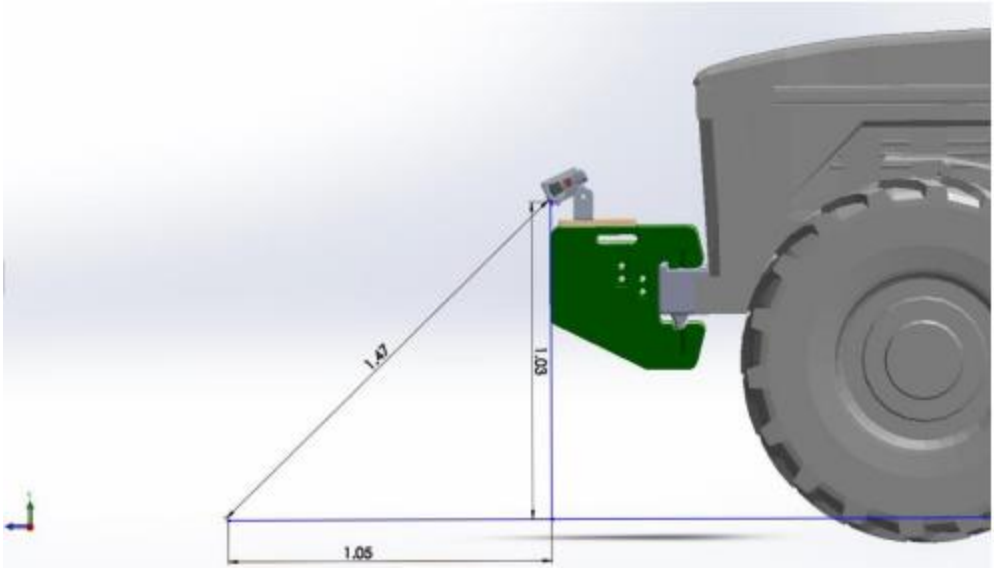

Figure 18 Side view of the camera mount installed on the tractor.

Enclosure and User Adjusting Mount

The camera enclosure was designed to allow for adequate protection of the camera and lens, while allows allowing for proper heat dissipation. It was inspired by high quality enclosures on the market used for machine vision applications. The normal operating temperature of the camera is specified in datasheet to be from 5°C to 45°C with a power consumption of 2.2W at 12V.

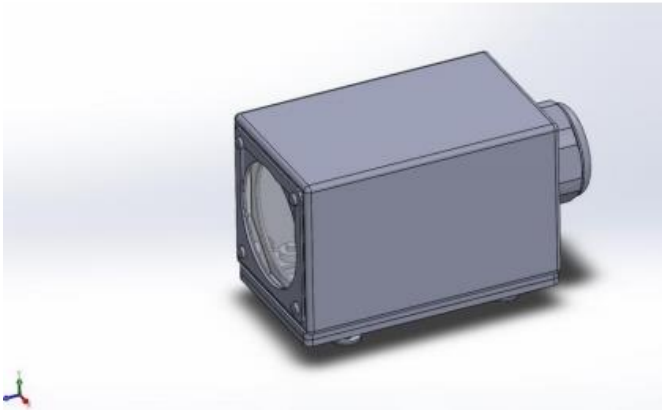

Figure 19 Camera enclosure design

The adjusting mount design involves two main pinned connections. The main load from the camera enclosure is supported with a high frictional tolerance between the supporting bracket attached to the base and the hinged platform for the enclosure. This bolt can be adjusted by the user to ensure the camera enclosure remains rigid, and is prevented from rotating during use.

The threaded rod protruding behind the camera mount is the selected method of fine camera adjustment. This mechanism is not designed to support a load, and has no part in the cameras rigidity to the mount. It is a tool for the end user to implement during the camera's set up. By tightening the nut and loosening the threaded rod, it manipulates the enclosure to produce a different working distance to the object it is targeting. This adjustment mechanism is important in order to support tractors that have different dimensions, in particular the height of the weight bracket may vary on different tractor models.

Refer to Figure 20 for an image of the SolidWorks model of the camera and base design.

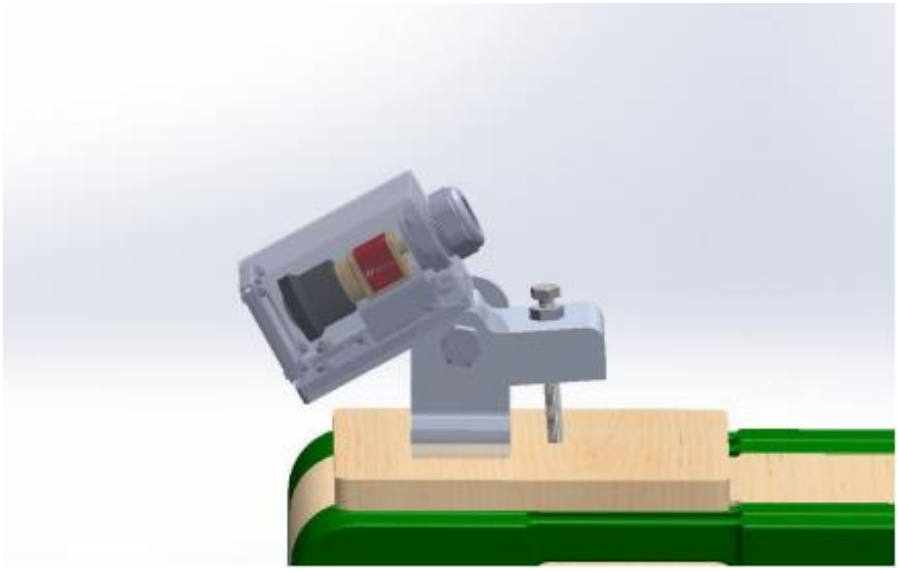

Figure 20 Camera enclosure mount and base design\

Front weight bracket attachment

The design for attaching the base to the front of the tractor has been finalized. As mentioned in previous reports, all John Deere brand tractors in the scope of our project are equipped with a bracket for fitting a balancing weight. Across a range of John Deere tractor models, each uses the same weight bracket. This means that a mounting system compatible with one model of John Deere tractor would also be compatible with another model of John Deere tractor. The John Deere part number of this bracket is RE117599, a CAD model and image of it installed on a John Deere 7410 may be seen in Figure 18.

Prototype Cost

The total material and tooling costs of the camera and mount subsystem's prototype are as follows:

Item	**Quantity**	**Cost**
USB Webcam	1	$20.00
¼-20 Nuts, Bolts	4	$0.14
¼ -20 Threaded rod	1	$3.00
10-32 Screws	8	$0.96
3D printing of all components	1	$20.00
Usb Extension cable	1	$4.00
Total		$48.10

Table 9 Table summarizing the prototyping cost of the camera and camera mount component of our system

Camera and Mounting Final Design

The following is a CAD model of the final design.

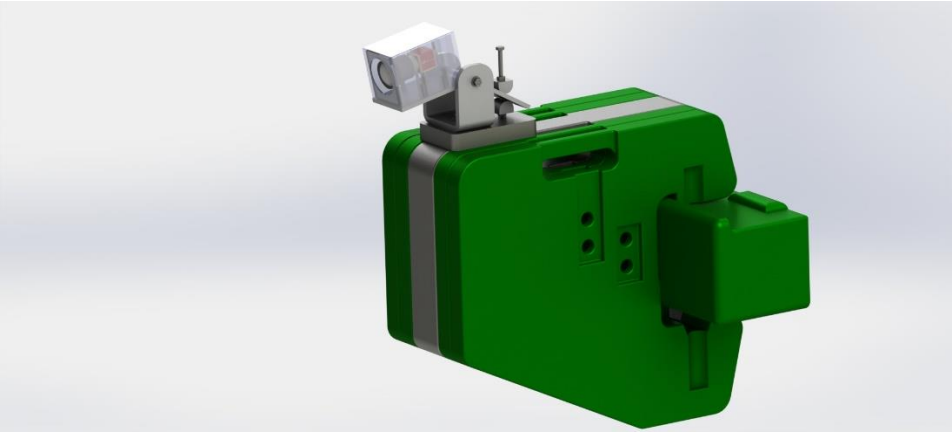

Figure 21 Camera enclosure, mount and base design

Validation

Software Validation

The software of the system was initially validated independently of the whole system. The purpose of this was to measure the accuracy of the algorithm previously described in the 'Algorithm' section of this report. The software validation was conducted using 100 consecutive frames from a video footage of a tractor driving in a corn field. The video was taken using a low-resolution webcam mounted at the front of the tractor using zip-ties. The algorithm and the threshold values determined for day-time condition was applied to all the images. A line was drawn where the algorithm had determined the row of corn to be. A sample of the images from the testing is shown in Figure 22.

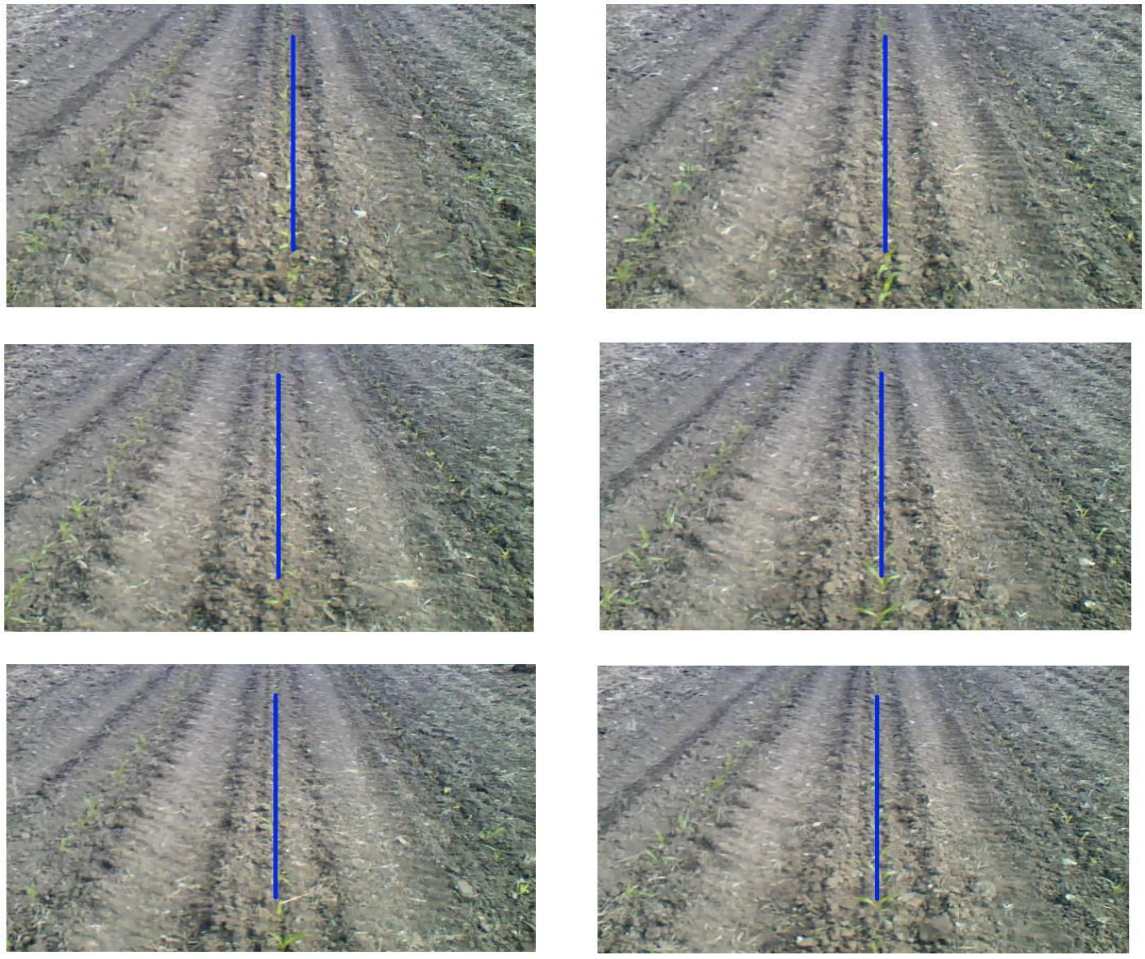

Figure 22 The above images show sample images that were tested to validate that the algorithm is accurate.

The 100 images were then manually tested using a MATLAB script. Each image generated from the algorithm was put through the validation script where a point on the line detected by the algorithm was selected and a point where the user would reasonably determine the row of corn to be was selected. The difference between these two points was divided by the total width of the image and multiplied by 100. The result was the positional error produced by the software. The average error of the 100 images was computed to be 0.518%. This met the design objective of keeping the accuracy of the algorithm high (it is over 99% accurate).

An important constraint to mention is that there was limited data was available to the team. Only the day-time conditions were validated in this process because images for cloudy or evening conditions were not available. However, the team was able to validate that the navigation software works during the day-time which would be the most common time for farmers to perform agricultural operations.

System Validation Protocols

The prototype was tested on a John Dheere 7410 tractor. The steering assembly and monitor with the Raspberry Pi were connected and mounted on the tractor. Due to weather constraints and lack of crop availability, the team initially conducted testing on an orange line spray painted on the ground. Then, after tuning the controller and adjusting the prototype, the team validated the prototype on road lines. The system was able to accurately follow the road lines with minimal oscillations. Hence, the prototype effectively validated that the designed system could accurately steer based on the machine vision algorithm. For more information on the controller and communication protocol for the system, refer to Appendix A.

System Validation Results and Analysis

In order to verify the performance of the control system, a full-scale test was deemed necessary. Do to the nature of the project a test bench prototype would not adequately demonstrate the feasibility of the control system design. For the test, the monitor, motor drive, and camera prototype subsystems were installed on a John Deere 7410 model tractor. Due to the lack of input data from crop rows, alternative tests were conducted to test the stability of the full system. As previously mentioned, the thresholding algorithm was already verified, so trivial image thresholding such as a line was adequate. Early small-scale testing used an orange line drawn on the ground with the algorithm thresholded for the bright colour. Once initial basic tuning of the double-PID loop was completed with confidence, a

full-scale test on the road was completed. The yellow line in the center of the road was thresholded for, and the system's ability to remain centered to the set point was tested. Initial tested resulted in an unstable system, with increasing oscillations around the center point. After tuning the system for the slightly altered environmental conditions, large oscillations were mitigated, until a stable system was achieved. The system was verified experimentally by running many tests. The auto steer system was able to maintain a straight course of action, with little deviation as specified by the scope of the project.

Conclusion

The original design required that the system be robust, portable, safe and cost-effective. Through the design process, the team's focus shifted to making the product marketable. As a result, the design objectives were refined such that it met consumer needs. Specifically, the original design involved having the system detect the end of the field and either automatically stop the tractor or alert the user that the end of the field has been reached. However, after conducting a market survey, it was concluded that users did not want an auto-stop feature included in the design because of the variability between fields making it hard to accurately detect what is at the end of a crop row. Furthermore, in order to make the product marketable to a wider range of users, the multiple steering interface option was introduced. The friction drive system was designed as a universal system that could fit on most popular brands and models of tractors. Alternatively, the software could also interface with an existing CAN bus or hydraulic system on the tractor.

Within the scope of the project, the team designed the friction drive wheel system, a camera mounting and a machine vision algorithm for the navigation system. All of these components were analyzed as detailed throughout this report. Force and moment calculations were done to verify that the steering design was feasible and met the design objectives effectively. Furthermore, the resolution and

lens focal length were analyzed for the camera mount by calculating the required focal length and the field of view. Finally, the image processing algorithm was examined based on the efficiency and accuracy. After this initial analysis, the team was able to build a full-scale prototype to validate the design.

The main limitation of the design was that there was not enough data available to the team. Moreover, due to weather constraints, field testing on actual crop rows could not be conducted. Hence, instability due to uneven terrain and rocks could not be validated by the prototype. However, the image processing algorithm was tested successfully with actual images and video footages of crop row. One of the main advantages of our design is that the algorithm uses colour-thresholding and the user interface allows the threshold values to be adjusted dynamically. This adaptability is imperative to our design because it provides a robust means of allowing the system's implementation to be extended to other agricultural applications.

Future Work and Recommendations

The prototype was functional in displaying the design intent. To further develop the solution as a product much remains to be completed.

The algorithm and control system were not able to be tested simultaneously in a field using crop rows as a reference. The driving conditions in a field are different than that of pavement; the soil is soft and deforms under tire pressure which could change system dynamics and require a different control law. The crop detection algorithm developed was only tested on corn in 30" rows at the 3 leaf stage. It is unknown whether the same algorithm is able to function on soybeans or at different corn growth stages.

The majority of the solution was manufactured using 3d printing technologies. This is not appropriate for a final product for multiple reasons; temperature stability, UV resistance, and general visual appeal. The steering solution could easily be redesigned to be manufactured from machined aluminum.

The HMI prototype was made as an assembly of off the shelf components, packaged in a 3d printed enclosure. A final product may be able to use an off the shelf touch screen tablet intended for vehicle use, such as those used in heavy equipment [10]. These tablets typically run an Android operating system and would require us to wrap the image processing and control code into a custom app. alternatively a completely custom HMI could be made using custom circuits and enclosures.

References

[1] Merrian Webster, "Dictionary," [Online]. Available: https://www.merriam-webster.com/dictionary/postemergence. [Accessed 13 11 2017]

[2] University of Saskatchewan, "Row Crop Cultivation," [Online]. Available: https://words.usask.ca/plsc243/management-techniques/tillage/tillage-in-crop/interrow-cultivation/. [Accessed 13 11 2017]

[3] Trimble, "Steering Solutions," [Online]. Available: https://agriculture.trimble.com/precision-ag/products/steering-systems/. [Accessed 13 11 2017]

[4] Shipman, "Introduction to Color Theory," [Online]. Available: http://infohost.nmt.edu/tcc/help/pubs/colortheory/web/hsv.html. Available: http://infohost.nmt.edu/tcc/help/pubs/colortheory/web/hsv.html. [Accessed 4 4 2018]

[5] M.S Sanders, E.J McCormick, Human Factors In Engineering Design. New York, New York City: McGraw Hill, 1993, pp.286

[6] K. B. Microsoft, M. Satran, and M. Jacobs, "Targeting - UWP app developer," Targeting - UWP app developer | Microsoft Docs, 02-Aug-2017. [Online]. Available: https://docs.microsoft.com/en-us/windows/uwp/design/input/guidelines-for-targeting. [Accessed: 06- Mar-2018].

[7] "Mastercraft Torque Adaptor | Canadian Tire," Shop Canada's Top Department Store Online & at 500 Locations. [Online]. Available: http://www.canadiantire.ca/en/pdp/mastercraft-torqueadaptor-0588900p.html. [Accessed: 13-Mar-2018]

[8] J. G. B. Enterprises, “Hoses, Fittings and Couplings,” JGB Enterprises, Inc - Largest INDUSTRIAL and HYDRAULIC HOSE company in Syracuse NY. [Online]. Available: http://www.jgbhose.com/Data_Returns/detail_company.asp?prod_id=1100016. [Accessed: 13-Mar2018].

[9] “DC Motors,” MTI Torque Systems > DC Motors > Platform 3500. [Online]. Available: https://www.torquesystems.com/dc-motors/dc-brush-servo-motor-platform-3500. [Accessed: 14-Mar2018].

[10] “FLEX Controls™,” *McNeilus*. [Online]. Available: https://www.mcneiluscompanies.com/concrete-mixers/flex-controls/. [Accessed: 9-Apr-2018].

Appendices

A. Communication Protocols

The final design of the system involved many separate control devices, and therefore required a method of communication between them. During prototyping the touch-screen user interface unit housed a Raspberry Pi for image processing, and an Arduino Nano for implementation of control systems and direct control of motor drivers. These lead to the Raspberry Pi acting as the main processing unit with the Arduino being utilized as a companion for driving the system. A one-way serial communication protocol was developed in order to convey information. The serial communication method developed involved prefacing sent data bytes with specific characters. When data is sent to the Arduino via software serial pins, it parses for the characters 's', 'x', 'e'. The Raspberry Pi also terminates all data strings with the '\n' character. When an 's' or 'x' character is read, it engages or disengages the auto steer. This allows for easy control by the user. When the Arduino reads an 'e' character, it parses for an integer value that corresponds to an error signal from the image processing algorithm. This value is then passed into the control system as an input. For the current prototype only a one-way communication was necessary, as the companion Arduino did not return data to the Raspberry Pi.

B. Processor Selection

Image processing and GUI code needs to execute on a processor. This application calls for a processor capable of running the required code in real time. The processor should also be capable of interfacing with the touch screen. Power consumption is not important since the amount of power used will be negligible in comparison to any other tractor system, instead the heat generated is important. The HMI enclosure should be dust proof, which limits cooling options. A passive cooling option is easy to implement in a dust free manner but is limited in the amount of heat it can dissipate. A low power consumption is therefore desirable. A group member had ownership of a Raspberry Pi 3B, a small

(85mm x 49mm) computer that features a 1.2 GHZ quad-core ARM Cortex A53. This computer met all the requirements and had the added benefits of the team being familiar with it and it already having been purchased.

C. Control System Processor

A processor separate from the main Pi was chosen to control the steering actuator. This choice was made to keep the system modular, allowing the image processing and steering to operate independently and to make quick control system changes without changing main Pi code. An Arduino Nano, coupled with a custom Bluetooth circuit, was chosen because it is small, the team was familiar with programming in the Arduino language, and it allowed programming changes to be made remotely from a laptop over a Bluetooth connection. This choice proved well during the prototype testing and tuning phase, often making control system changes wirelessly from a laptop while the tractor was in motion.