DEVELOPMENT OF THE AUTONOMOUS COTTON HARVESTING ROBOT

by

KADEGHE GOODLUCK FUE

(Under the Direction of Glen Rains and Changying Li)

ABSTRACT

Timely harvesting of quality cotton fiber is among the most pressing challenges in the cotton production industry. The current practice of mechanical harvesting after defoliation has increased acreage and production, but reduced efficiency of harvest. Farmers pick the cotton after at least 60% to 75% of the cotton bolls are open, at which point many of the earlier opening bolls have been exposed to weather more than 40 days waiting to be picked. Boll quality is compromised, and some bolls have already fallen to the ground, unharvestable. An additional problem is the availability and expense of a skilled labor force. The average age of a farmer in the U.S. is 60 years old, and that age has been increasing for decades. Thus, it is paramount to utilize the current nascent technologies in automation and robotics to develop revolutionary solutions to address these issues. This dissertation focuses on the development of the cotton harvesting robot to increase efficiency, save labor, and improve farming management.

A center-articulated and hydrostatic rover with an attached cartesian manipulator was designed and implemented. The robot integrated advanced sensing systems using encoders, a low-cost RTK-GNSS, a potentiometer, RGB stereo cameras, and IMUs to control navigation and picking manipulators. The robot also integrated three controllers to do advanced object detection, and control to harvest the bolls. Robot Operating System (ROS) was used to integrate and

control the robotic system for cotton boll tracking, cotton boll location estimation, cotton rows detection, navigation, and harvesting. The sensor fusion algorithm Extended Kalman Filter (EKF) was utilized to perform autonomous localization and navigation of the robot. Cotton harvesting was achieved by using a ROS-independent finite state machine (SMACH), modified pure pursuit algorithm, and proportional–integral–derivative controller. The performance of the robot was evaluated and reported. Experimental results showed that the developed robot could precisely and efficiently navigate over the cotton rows and harvest the cotton bolls. Furthermore, the robotic design has shown that traditional vacuum harvesting can well be adopted in robotic systems to harvest the cotton bolls. The designed robot sets preliminary development success to improve cotton harvesting management.

INDEX WORDS: Cotton, Machine Vision, Cotton Robotics, Stereo Camera, Visual Servoing, Precision Agriculture, Cotton Harvesting, Localization and Navigation, Center-articulated rover, 3D imaging, 3D Location, PID Control, RTK-GNSS, ROS, Cotton Boll Tracking, Cotton Rows Detection, Cotton Boll 3D Location Estimation, Row-crop Navigation

DEVELOPMENT OF THE AUTONOMOUS COTTON HARVESTING ROBOT

by

KADEGHE GOODLUCK FUE

B.S., University of Dar es Salaam, Tanzania, 2011

M.S., University of Florida, 2014

A Dissertation Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

ATHENS, GEORGIA

2020

DEVELOPMENT OF THE AUTONOMOUS COTTON HARVESTING ROBOT

by

KADEGHE GOODLUCK FUE

Major Professors:	Glen Rains
	Changying Li
Committee:	Wesley Porter
	Mark Haidekker
	George Vellidis

Electronic Version Approved:

Ron Walcott
Interim Dean of the Graduate School
The University of Georgia
August 2020

DEDICATION

To all the people who are working in the cotton industry and agriculture around the world.

To my darling wife, for her kindness and devotions, and for her endless support to take care of the kids during all the time that I was in school.

To my precious mama and beloved sister for their kindness and endless support.

To my late dad and brother, who are now in heaven for their prayers. I pray that you continue to rest in peace. "And I heard a voice from heaven saying unto me, Write, Blessed are the dead which die in the Lord from henceforth: Yea, saith the Spirit, that they may rest from their labours; and their works do follow them." (Revelation 14:13)

"I can do all things through Christ who strengthens me. " (Philippians 4:13)

ACKNOWLEDGEMENTS

I have been blessed with talented and experienced mentors to guide me throughout my research. I would like to send my special appreciation and gratitude to my advisor, Dr. Glen Rains, for his unwavering support and guidance on my research. He has done tremendous work to review my research work and provide timely advice whenever I needed it. I look forward to doing more research with you in the future.

I would like to thank my advisors, Dr. Changying Li, and Dr. Wesley Porter. Dr. Li provided tireless support on my engineering courses and guidance. My research work on planter downforce studies with Dr. Porter was tremendous, and he has provided me with excellent support and guidance throughout my studies in Tifton, Georgia. I look forward to doing more research on planters with you in the future.

My special thanks and appreciation go to Dr. George Vellidis and Dr. Mark Haidekker for serving on my committee. I appreciate your feedback and constructive comments on my research. Also, I would like to thank Cotton Incorporated and Dr. Edward Barnes for supporting my research. I would also like to thank my research collaborators, Mr. Ricky Fletcher and Mr. Gary Burnham, for their helpfulness and technical support in building the rover platform and collection of data. Also, I would like to thank Mr. Logan Moran for the field testing of the rover, Mr. William Hill, and Ms. Tearston Adams for labeling the cotton images, and Mr. Jesse Austin Stringfellow for 3D design and printing.

I sincerely appreciate the opportunity that I was given by my employer in Tanzania (Sokoine University of Agriculture) for study leave.

I am very grateful to my dear wife, kids, mom, and sister, who spent so much time and energy for prayers while encouraging me to pursue my dreams. They provided me with all the love and invaluable support that I have never imagined.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

1.1 Background and Significance of This Study

Cotton, as a commercial crop, holds an essential position worldwide. The cotton industry, worth $25 billion, employs more than 200,000 people in the U.S. (USDA/NASS, 2018). The U.S is third in the production of cotton in the world behind India and China. As a large industry, however, it has faced multiple challenges in its operations. Among the biggest challenge is the timely harvest of the quality cotton fiber. Specific cotton harvesting challenges are;

- Open cotton bolls can sit up to 50 days until picked when at least 60% to 75% of the cotton bolls are opened (UGA, 2019). This waiting time exposes the open bolls to harsh conditions that degrade their quality.
- It is not possible now to harvest low or high micronaire cotton separately and hence, increases the cost of separation later on.
- Contamination of the cotton bolls (mulches, sticky cotton, or plastics) during harvesting increases the cost of ginning and may reduce bale prices.
- The mechanical combines are huge and expensive. The new 2019 picker costs around $725,000, and it is stored (under the shed) for more than nine months a year without being used.
- Cotton combines weigh more than 33 tons, causing soil compaction, which reduces land productivity.

- The maintenance of combines is expensive and complicated due to the size and weight of the machine.
- Breakdowns in the field can take days, reducing operating efficiency and exposing bolls to further weather-related quality degradation.
- Most of the machines use proprietary software and hardware that prevents farmers from repairing their machines and hence, deny the right-to-repair tools that they own (Waldman & Mulvany, 2020).
- The labor shortage in agriculture is getting worse while the cost of available labor is skyrocketing (Zahniser et al., 2018). It is due to youth movement to urban areas leaving behind an aging farming society.

The rise of nascent robotics and Artificial Intelligence (AI) in agriculture, especially in specialty crops, creates an opportunity to adopt robotics in row crops such as cotton, which have received little attention until recently (Bergerman et al., 2016; Comba et al., 2010; Fue et al., 2020b; Ramin Shamshiri et al., 2018). Furthermore, robotics systems are small and can be designed to accomplish multiple farming tasks in addition to harvesting. To the best of our knowledge, there are no commercial cotton harvesting robots yet (Fue et al., 2020b).

Hence, it is essential to aggressively develop a robotic harvester that provides an opportunity to have multifunctional equipment that can be used in multiple farm operations. A robot will be able to serve farming society by;

- Harvesting early cotton bolls without waiting. That could serve farmers in case disasters like hurricanes happen and help improve quality of cotton harvested.
- Separating low or high micronaire bolls.
- Removing the use of human drivers in daily operations.

- Harvesting clean cotton bolls which are free from contamination.
- Reducing the cost of maintenance and breakdown costs.
- Reducing the cost of ginning since the robot picks the cotton bolls directly without mixing it with other plant debris.
- Reducing the costs of owning cotton harvesting machines by providing equipment that costs a fraction of the current cost of the combines.
- Removing the use of the chemical defoliants and hence, serve the environment and reduce cost.
- Reducing fatal or severe injuries while working in dangerous environments.
- Reducing soil compactions and improve soil productivity.
- Allowing quicker return of the equipment to farming after a rain event since the robots are light and can navigate muddy conditions.
- Providing a chance for farmers and other technicians to repair the open-source robots.

1.2 Objectives

The overall goal of this dissertation was to develop a cotton harvesting robot and test its performance in real field and direct sunlight conditions.

Specific objectives were to:

1. Conduct an extensive review of current trends in cotton robotic harvesting systems.
2. Develop cotton bolls and rows detection algorithms using RGB stereo cameras
3. Develop cotton boll tracking algorithms using a moving RGB stereo camera

4. Develop robust systems to localize and control navigation of a center-articulated and hydrostatic transmission rover using a low-cost RTK-GNSS, IMUs, and encoders
5. Develop cotton harvesting robot and robust algorithms using task-level architecture techniques

1.3 Research Contribution and Dissemination

The algorithms and system hardware developed in this study are open-source systems and open-robotics. They can be adopted by the industry and other pioneers to accomplish the dream of developing cheap, dynamic, robust, and reliable robotic systems for cotton harvesting. The project codes are shared freely in our GitHub accounts; https://github.com/kadefue and https://github.com/UGA-AgRobotics

The study has contributed to the cotton industry development. It stands as one of the early efforts to make a cotton harvesting robot in the U.S. The outputs (15 articles) have been either submitted or published in journals and presented in conferences and workshops. The leading publications that are presented in this dissertation are as follows:

[1]. **Fue, K.,** Barnes, E., Porter, W., Li, C., and Rains, G., (2020). Center-articulated Hydrostatic Cotton Harvesting Robot using Visual-servoing Control and a Finite State Machine. Sensors (Submitted)

[2]. **Fue, K.,** Barnes, E., Porter, W., Li, C., and Rains, G., (2020). An Autonomous Navigation of a Center-articulated and Hydrostatic Transmission Rover using a Modified Pure Pursuit Algorithm in a Cotton Field. Electronics (Submitted)

[3]. **Fue, K.,** Barnes, E., Porter, W., Li, C., and Rains, G., (2020). Evaluation of a Stereo Vision System Effectiveness in Row Detection and Boll Location Estimation on a Cotton Harvesting Rover in a Direct Sunlight. Agronomy (Submitted)

[4]. **Fue, K**., Barnes, E., Porter, W., and Rains, G., (2020). Ensemble Method of Deep Learning, Color Segmentation, and Image Transformation to Track and Count Bolls using a Moving Camera in Real-time. Transactions of ASABE. St Joseph, MI: ASABE. (Submitted)

[5]. **Fue, K.,** Barnes, E., Porter, W., and Rains, G., (2020). An Extensive Review of Mobile Agricultural Robotics for Field Operations: Focus on Cotton Harvesting. AgriEngineering., 2(1):150-174

1.4 Overview of the Dissertation Chapters

This dissertation consists of seven chapters. Chapter 1 discusses the significance of this study, objectives, research contribution, and overview of the dissertation chapters. Chapter 2 provides an extensive review of cotton harvesting robotics studies and discusses the current status of agricultural robotics around the world.

Chapter 3 studies the use of color stereo cameras to detect cotton bolls and rows. Accurate cotton bolls detection and 3D location estimation are essential for robotic use. Calibration of the camera is particularly essential for the precise detection of the cotton bolls location. Cotton rows detection in a heavily occluded farm is primarily vital if the navigating rover loses GNSS fix while working and hence, could provide assistance to the robotic system to continue working and navigating. The algorithm was able to detect rows and estimate the positions of the cotton bolls.

Chapter 4 reports the development of the tracking algorithms for cotton bolls. The robot needs to remember the positions of the bolls while working on the farm since detection is not reliable. The best technique is once detected, never forget the position until the boll is harvested.

The algorithm explicitly took advantage of the appearance of the cotton bolls to track them. The developed algorithm was compared with the other six open-source industry-standard algorithms.

Chapter 5 introduces the use of the fusion algorithm Extended Kalman Filter (EKF) to help autonomous robots localize and navigate along the cotton rows. The designed algorithm was able to use several sensors such as a low-cost RTK-GNSS, IMUs, encoders, and potentiometer to guide the rover along the cotton rows accurately. The rover was a center-articulated and hydrostatic transmission rover. The developed rover and algorithms can be adopted in any other center-articulated rover.

Chapter 6 reports the center-articulated and hydrostatic robot that uses a ROS-independent finite state machine method to harvest the bolls autonomously. The robot used YOLOv3 to detect the bolls. It used the 2D Cartesian manipulator to pick the bolls. The robot was programmed to be controlled using PID control to autonomously position itself close to cotton bolls for the manipulator to pick.

Chapter 7 provides research conclusions, limitations of this dissertation, and future research studies.

CHAPTER 2

AN EXTENSIVE REVIEW OF MOBILE AGRICULTURAL ROBOTICS FOR FIELD OPERATIONS: FOCUS ON COTTON HARVESTING[1]

[1] Fue, K., Barnes, E., Porter, W., and Rains, G., (2020). *AgriEngineering.*, 2(1):150-174. Reprinted here with permission of publisher.

2.1 Abstract

In this review, we examine opportunities and challenges for 21st-century robotic agricultural cotton harvesting research and commercial development. The paper reviews opportunities present in the agricultural robotics industry, and a detailed analysis is conducted for the cotton harvesting robot industry. The review is divided into four sections: (1) general agricultural robotic operations, where we check the current robotic technologies in agriculture; (2) opportunities and advances in related robotic harvesting fields, which is focused on investigating robotic harvesting technologies; (3) status and progress in cotton harvesting robot research, which concentrates on the current research and technology development in cotton harvesting robots; and (4) challenges in commercial deployment of agricultural robots, where challenges to commercializing and using these robots are reviewed. Conclusions are drawn about cotton harvesting robot research and the potential of multipurpose robotic operations in general. The development of multipurpose robots that can do multiple operations on different crops to increase the value of the robots is discussed. In each of the sections except the conclusion, the analysis is divided into four robotic system categories; mobility and steering, sensing and localization, path planning, and robotic manipulation.

2.2 Introduction

The cotton industry holds an important position as a commercial crop worldwide, especially in the U.S, China, India, and Brazil, who are the leading producers of cotton (USDA/NASS, 2018). However, the cotton industry has several challenges, particularly in cotton harvesting. Timely harvesting of the quality cotton fiber is among the most pressing challenges in the cotton production industry. The current practice of mechanical harvesting after defoliation has led to huge losses in the industry since its inception in the 1950s (Fue et al., 2018b). The

open bolls can sit 40 days waiting to be picked, since it is advised to pick the cotton after at least 60% to 75% of the cotton bolls are opened (UGA, 2019). Also, the cotton picker needs defoliated plants to harvest, which adds expense to the farmer (UGA, 2019). The defoliants are applied to the plants, and the farmers need to wait 10 to 14 days before harvesting the crop. Defoliant activity can also be compromised by rainfall. Also, cotton is harvested at or below 12 percent moisture because wet cotton brings clogging problems in the picker and the ginning process, which can add more waiting time during harvesting (UGA, 2019). This waiting time exposes the open bolls to harsh conditions that degrade their quality. Any solution that would reduce cotton losses and improve quality would be welcomed by the industry. In most cases, the mechanical combine machines are very big and expensive (the current six-row cotton picker costs around $725,000). Unfortunately, expensive cotton pickers are stored under the shed for more than nine months a year, waiting to harvest for only three months. Also, the machines weigh more than 33 tons, causing soil compaction, which reduces land productivity (Antille et al., 2016). The maintenance of such machines is also expensive and complicated. Breakdowns in the field can take days to repair, reducing operating efficiency, and exposing bolls to further weather-related quality degradation (Fue et al., 2018b).

Most cotton harvesting technologies are either “stripper” or “spindle” pickers (UGA, 2019). The “stripper” grabs the lint from the plant and some amount of plant matter (UGA, 2019). Later, the lint is separated from the plant matter by dropping the heavy plant matter while leaving the lint behind, which is directed to the basket at the back of the machine. The “spindle” grabs the seed-cotton from the plant by using barbed spindles that rotate at a high velocity. Then, a counter-rotating doffer is used to strip the seed-cotton from the spindles (UGA, 2019). So, for each row, one “stripper” or “spindle” picking tool is used. It means six-row picking technology

has six picking tools for each row. The current six-row picking technology navigates at 5.5 mph in the field and covers around 8 to 10 acres per hour. Boman (2012) and Prostko et al. (2018) estimated that 40-inch row spacing has 23.2 bolls per row-ft for two bales per acre yield while 30-inch row spacing has 17.4 bolls per row-ft for a two bales per acre yield. A 40-inch row has 13,081 linear feet per acre, while a 30-inch row has 17,424 linear feet per acre. It means that an acre has an estimated 303,479 bolls per acre for a 40-inch row and 303,178 bolls per acre for 30-inch row spacing (Table 2.1). Small robotic rovers that collect at least 12,140 bolls per trip every day 25 times per harvest cycle will cover around 303,500 fresh open bolls that have been exposed to minimum degradation. A small rover moving at 3 mph and picking one boll every 3 seconds and working 10 hours per day would finish harvesting one acre of 40-inch rows within 50 days (Table 1). Hence, the development of a robot that costs around $7000 will equate to 104 robots to compare with one large machine that costs more than $725,000 (Table 2.1).

Table 2.1. Comparison of the conventional machine and robot for cotton harvesting.

Parameters	**Conventional Machine**	**One Manipulator Robot**
Number of bolls per acre	303,178	303,178
Times to harvest per acre (pass)	1	25
Time to harvest an acre(hours)	0.1	250
Unit Cost	$725,000	$7000

One potential advantage of robotics is using a single rover platform for multiple tasks by using interchangeable attachments. By changing attachments and selecting the appropriate software

application, robots can perform tasks like planting, weeding, spraying, and harvesting. Also, these machines can be reprogrammed to cover different tasks on a different crop, which can be a huge cost-saving measure for both small and large farmers.

The deployment of autonomous machines can improve the quality of the fiber since only a boll is picked. Autonomous means no human intervention, which reduces labor costs. Also, the need to serve the environment by stopping the use of undegradable defoliants is very important, as the robots may pick the mature bolls as they appear without defoliation. The use of lightweight machines dramatically reduces soil compaction (Fue et al., 2019b). The cost of operations can also be reduced since autonomous robots may need less supervision and hence low labor costs. Also, electrical energy sources like solar energy can be introduced to reduce fuel costs because robotic machines are light, electric and can survive with limited energy consumption.

The unstructured environment, like the agricultural field, requires more advanced methods of machine learning (C Wouter Bac et al., 2014). The unstructured approach is required for agriculture rather than a structured approach (Roldán et al., 2018). However, advancements of machine vision, machine learning (especially deep learning), sensing, and end effector manipulation have fueled the application of robots in an unstructured agricultural environment. However, most of these machines have been deployed in horticultural crops only (Lowenberg-DeBoer et al., 2019).

The development of a cotton harvesting robot is feasible because there is an opportunity to use the current advancements in machine vision, actuators, motors, and agricultural robotics to develop an autonomous platform to harvest cotton bolls and be adaptable to other cotton operations like planting, spraying, weeding and scouting. To the best of our knowledge, there is

no commercial cotton harvesting robot available yet. Hence, we propose to review current cotton harvesting robot research and opportunities and challenges for future development.

2.3 Methodology

We discuss the most important issues concerning agricultural robots, harvesting robots, and finally, cotton harvesting robots. We start our discussion with general agricultural robots and then harvesting robots because we believe multipurpose agricultural robots have more value compared to single-purpose agricultural machines. So, it is feasible to adopt some of the commercially available machines to develop cotton harvesting robots at a lower cost. To achieve this, the literature was identified using the following keywords; "robot," "agricultural robot," "cotton harvesting robot," "crop imaging," "cotton harvesting," "cotton robot," "picking robot," "robots in agriculture" and "harvesting robots." Several relevant papers with the keywords from the leading databases and indexing, such as Web of Science, Google Scholar, Science direct, UGA libs, ProQuest, IEEE Xplore, and Scopus, were retrieved and identified. Also, Google and Bing search engines were used to retrieve any commercial or non-academic material related to the mentioned keywords, as most of the commercial companies prefer to advertise products instead of writing scientific papers. Since the cotton harvesting robot is a new idea, there were very few materials covering the topic. Other sources of information like YouTube were also investigated to uncover any commercial or related works presented by companies or hobbyists. Approximately 74 peer-reviewed articles, 4 chapters, 3 books, 6 scientific reports, 24 refereed conference proceedings, and other sources from websites of commercial agricultural robotic companies were selected and included in this review.

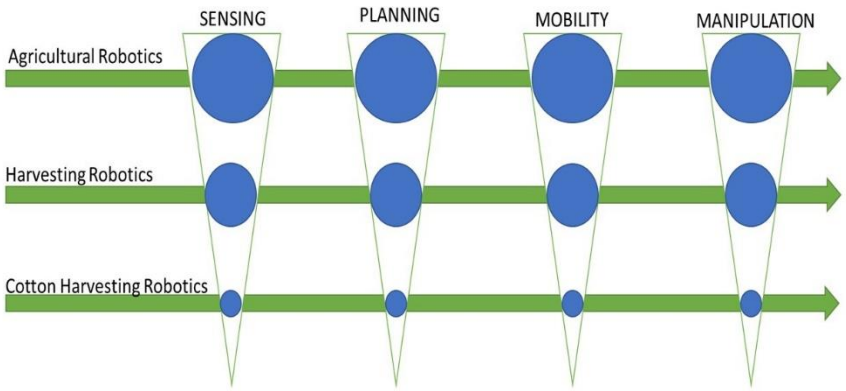


Figure 2.1. Framework organization of this paper.

Each of the articles retrieved was analyzed according to the robot components in Figure 2.1. Robots usually consist of 4 components (Figure 2.1): sensing and localization (Sensing), path planning (Planning), mobility and steering (Mobility), and end effector manipulation (Manipulation) (Bechar & Vigneault, 2016). The adoption and performance of agricultural robots rely solely on those four components (Bechar & Vigneault, 2017). It is difficult for a robot to succeed when some of the components do not meet expectations. Mobility and steering mainly focuses on providing the movement ability of the robot to reach the target and accomplish the mission. This can consist of legs, wheels, tires, plane wings, undulation abilities, propellers, etc. Sensing and localization is the perception of the environment and its occupying objects that may allow or hinder the operation of the robot. The robot needs to reason which environmental characteristics are conducive or not for it to operate. This is done by detecting a clear path, obstacles, detecting targets, and remembering current and past positions. Path planning is the optimized decision made by the robot to reach the targets. Robots need to be designed to identify the target position and then plan their movement according to their capability and the sensed information. This allows the robot to decide the most optimized path that can lead to a quick

achievement of the mission. When the robot reaches the target, then end effector and manipulators (manipulation) are executed to accomplish the mission. In the case of agriculture, this can be picking fruit, spraying chemicals, measuring the size of the target, killing weeds, picking soil or leaf samples, planting seeds, plowing land, or plant irrigation.

Therefore, the robot's operation is summarized as sense, reason, plan, and act. In each of these procedures, the robot may succeed or fail and take some time to succeed or fail. Therefore, it is possible to measure success rates and execution times of each operation. Bechar and Vigneault (2017) proposed a very good method and parameters to measure robot performance in the field (Table 2.2).

Table 2.2. Methods and parameters to measure the performance of the robot in the field (adapted from Bechar and Vigneault (2017)).

Measure	Description
CT: Cycle Time (s)	The average time required to finish a specific action in a task. (e.g., harvesting a cotton boll, spraying herbicides, scouting with camera)
OT: Operation Time under real-time conditions (s)	The average time required to finish an intended task under real-time in an agricultural field. This can be time taken from the start of robot planning, navigation, sensing, and manipulation.
OV: Operation Velocity under real-time conditions (inch s-1)	Average velocity taken by the robot to finish a mission (navigation can be very complex or simple depending on-farm management task)
PR: Production Rate (lbs h-1, ac h-1, number of actions h-1 , etc.)	Amount of successful actions or task (e.g., number of cotton bolls picked) treated per time unit
CORT: Capability to Operate under Real-Time	The ability of a robot to accomplish tasks under real-time conditions presented in binary form: either can operate under real-time conditions, CORT+, or cannot operate under real-time conditions, CORT-. This can be achieved if navigation, sensing, and manipulation are well designed.

Measure	Description
conditions (CORT+ or CORT-)	
DC: Detection Capability (DC+ or DC-)	The ability of robot sensors to detect objects to accomplish a specific mission and it is presented in binary form; either a robot can detect an object, DC+; or cannot detect an object, DC-
DP: Detection Performance (%)	Performance of the robot in detecting objects for its mission. Detection results can be True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). DP is the sum of the True positives and True Negatives over all the elements that were presented for detection. Other parameters like accuracy, recall, precision, and F1 score can be calculated (Powers, 2011).
ASR: Action Success Ratio (%)	The ratio of successful actions performed by the robot without destroying the plant over the total number of actions
ADM: Appropriate Decision-Making (%)	The ratio of the number of correct decisions made over all the decisions done by the robot while accomplishing an agricultural task
PEa: Position Error Average and PEsd:	The standard deviation and average of positioning error made by a robot from true locations where it is located to reported location sensed by the robot's sensors.

Measure	Description
Position Error Standard Deviation (inch, etc.)	
Safety	It the parameter that describes robot behavior on the farm that cannot threaten other objects around the farm. It is the safe actions of the robot while operating in an agricultural field.
Wholeness	The ability of the robot to execute tasks as required or as designed to full completion using its autonomous coordination of actions to accomplish all the tasks.

Finally, we discuss four main topics regarding cotton harvesting robots consisting of the four main operations in Figure 2.1. In the first topic, we discuss general agricultural robotic advances and opportunities that can be inherited in specified field operations. In the second topic, we discuss the main harvesting robots available and similarity to other farm operations. In the third topic, we discuss cotton harvesting robotics status and compare it with other harvesting robots. Lastly, we conclude and frame the future work required for cotton harvesting robots.

2.4 Agricultural Robotics

General operations in agricultural robots are equivalent to each other for similar crops and differ slightly in other types of crops. It is important to discuss the agricultural robot framework that can be adapted to other crops. In agriculture, various jobs such as plant phenotyping, sorting and packing, scouting, mowing, pruning, thinning, planting, spraying, weeding, harvesting and picking could be automated using robots. This can be achieved by the same robot with changes in attachments and selecting a different computer program for robotic perception and a different end-effector for the new task. This kind of robot is called a multi-functional intelligent agricultural robot (MIAR).

2.4.1 Agricultural Robot Mobility and Steering

Most agricultural robots reported use wheels or legs (Table 2.3). Legs are advantageous for flexible movement in the agricultural field with high occlusion of stems and branches, but wheels provide faster and more convenient navigation in the field. Some emerging technologies involve the use of drones for agricultural operations, such as spraying

and scouting, but are excluded from other operations such as crop harvesting and pruning. A more leveraged approach for operations like scouting is to combine the drone large area sensing with a ground robotic system that is partially directed by analysis of drone data (Burud et al., 2017). The combined system achieves timely and efficient operations in the agricultural field (Burud et al., 2017). Legged robots may be limited in speed, but are advantageous for multiple obstacle avoidance, irregular terrains, and crevices (Iida et al., 2008). Over time, the deployment of wheeled robots has become more prevalent (Iida et al., 2008). Comparing the two, the execution time is good with the wheeled robot, but legged robots achieve a good success rate, and have the flexibility to maneuver over diverse terrains (Iida et al., 2008).

There are several types of mobility in agricultural robots according to the condition of the agricultural field or robotic operation that would be cost-effective and fast. For high-speed navigation, robots over rails are useful, especially in phenotyping studies (Figure 2.2b). Legged robots like AgAnt (Figure 2.2a) and the Tarzan robot, which swings over the crops on a wire (Figure 2.2c), are both preferred in wetlands and close inspection of crops and animals. The rack-like (Reiser et al., 2019) weeding robot (Figure 2.2d) and Fuji Agricultural robot (Figure 2.2f) are preferred in slippery grounds to reduce skidding. Dogtooth (www.dogtooth.tech), which is a strawberry harvesting robot, uses a track in a nominal strawberry growing system because it is a convenient method of navigation in greenhouses. Swinging robots like the Tarzan robot discussed above can be very good for high throughput phenotyping tasks as they can maneuver close to the plants or animals compared to drones.

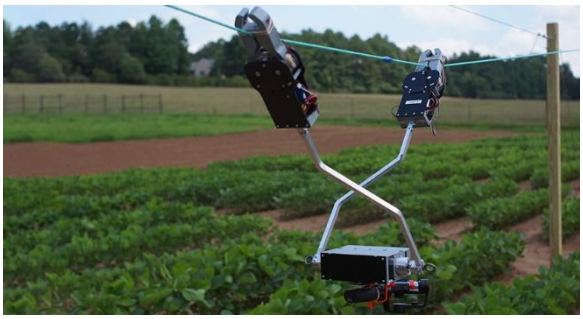
However, the mobility needed also depends on the flexibility required on the farm. Four-wheel steered robots like SwagBot (Figure 2.2i), Thorvald II (Figure 2.2e), or Agribot (Figure 2g) are required for conditions where the wheel traction is difficult, such as a feedlot, or any muddy environment. However, in most cases, for normal operation, a two-wheel turning robot is enough for farming operations.

Auat Cheein et al. (2011), Ouadah et al. (2008), and Cheein et al. (2010) presented a simple model for a mobile robot that can explicitly demonstrate how mobility is modeled with a car-like unmanned mobile robot.

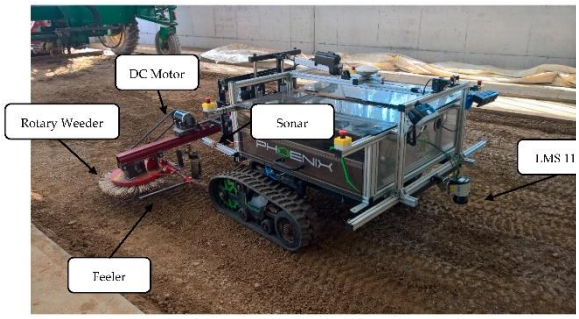
Xue et al. (2012) reported a skid-steer robot that controlled the wheels on either side of the mobile robot by linking them.

(**a**)

(**b**)



(**c**)



(**d**)

(**e**)

(**f**)

(**g**)

(h)

Figure 2.2 Some of the agricultural robots with a different arrangement of components; (a) AgAnt (source: cleantechnica.com), (b) Fraunhofer Institute for Production Systems and Design Technology IPK dual-arm robot (source: agromarketing.mx), (c) Tarzan swing robot (Davies et al., 2018; Farzan et al., 2018) (**d**) Weeding Robot (Reiser et al., 2019) (**e**) Thorvald II Agricultural Robotic System Modules (Grimstad & From, 2017) (**f**) Fuji industry Robot (source: fuji.co.uk) (**g**) RAL Space Agribot with robot arm weeding raspberries (source: autonomous.systems.stfc.ac.uk) (**h**) SwagBot, omnidirectional electric ground vehicle (source: confluence.acfr.usyd.edu.au).

The navigation of a robot (Figure 2.3) in row-crop production should be easy to track and retrieve while working to allow self-navigation when some of the sensors (GPS or IMUs or cameras) fail.

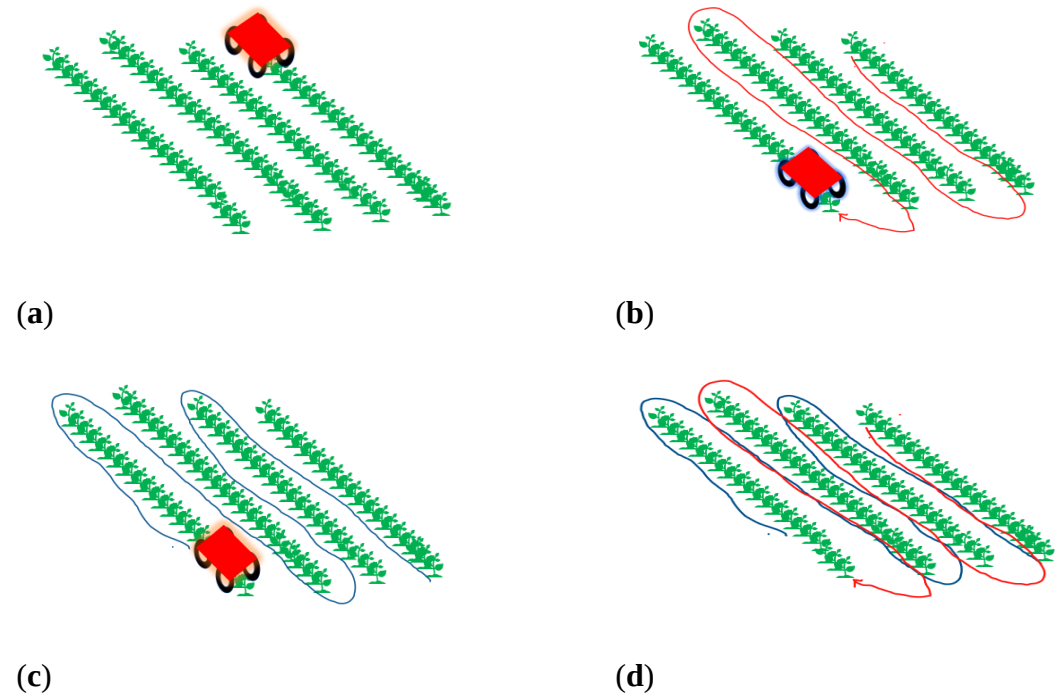


Figure 2.3. Tracking the robot using wheel odometry of the camera Inertial Measurement Units (IMU) (the autonomous rover can be tracked while working on the farm by using visual SLAM and GPS); (a) Rover is starting to navigate, (b) Rover is about to finish the farm (**c**) Rover can generate the returning path by using history navigation (**d**) Blue is the predicted path going back while red is the path taken by the rover.

2.4.2 Agricultural Robot Sensing

Sensing is done to update the system on the environment so that it can navigate or pick fruits (Bechar & Vigneault, 2016; Fue et al., 2018b), discover disease, insects, or weeds, control spraying height above the canopy, and other tasks. Robust sensing systems are

required for the robot to work well in dynamic environments with changing weather conditions, vegetation variation, topographical changes, and unexpected obstacles. Most agricultural robots, so far, use image sensing systems and Global Navigation Satellite Systems (GNSS) to achieve the localization of the robot (Figure 2.3). The advancement of imaging technologies has provided a great opportunity to sense and create 2D, 3D, and 4D (spatial + temporal) images of plants (Rahaman et al., 2015). Technologies to obtain 2D, 3D and 4D perception of the environment has been achieved using the following sensors in agricultural fields; visible light, near-infrared, thermal, fluorescence, spectroscopy, structural topography imaging, fluorescence, digital imaging (RGB), multispectral,color infrared, hyperspectral, thermal, spectroradiometer, spectrometer, 3D cameras, moisture, pH, light-reflective, light detection and ranging (LIDAR), sound navigation and ranging (SONAR), ground-penetrating radar and electrical resistance tomography (Cubero et al., 2011; Deery et al., 2014; Dong et al., 2017; Rahaman et al., 2015; Safren et al., 2007; Sankaran et al., 2015; Sun et al., 2017).

Other sensors, such as potentiometers, inertial, mechanical, ultrasonic, optical encoder, RF receiver, piezoelectric rate, Near Infrared (NIR), laser range finder (LRF), Geomagnetic Direction Sensor (GDS), Fiber Optic Gyroscope (FOG), piezoelectric yaw, pitch and roll rate, acoustic and Inertial Measurement Units (IMUs) have been used to provide direction of the robot and navigation feedback (Bak & Jakobsen, 2004; Mousazadeh, 2013).

The choice of imaging sensor somewhat depends on the distinct characteristics of the target from the rest of the obstacle-dense environment. The normal digital camera may be used if the target on the field can be visually identified. For example, to identify green citrus or green bell pepper in a population of green plants may require using an alternative sensor, or the method of detection may be complicated by involving advanced methods of machine learning (Choi et al., 2017; Moghimi et al., 2015; Qureshi et al., 2017; Sengupta & Lee, 2014; C. Wang et al., 2018). Images may suffer from illumination changes, motion change, cluttering, temperature swings, camera motion, wind-induced movements, deformation, and scene complexity. Hence, some image refinement algorithms may be required to enhance the images (Choi et al., 2017; C. Wang et al., 2018). Then, object recognition or feature extraction using pattern recognition and other machine vision algorithms can be performed. There are several methods of image rectification and enhancement that have been reported; image smoothing and segmentation (Hannan et al., 2007; Moghimi et al., 2015), morphological operations and filters (Choi et al., 2017; Sengupta & Lee, 2014), a fast bilateral filtering based Retinex (C. Wang et al., 2018), illumination normalization (C. Wang et al., 2018), image color space-changing (Tao et al., 1995), and normalized co-occurrence matrix and gray level co-occurrence matrix (Chang et al., 2012). Feature extraction can be achieved using classical image processing techniques or advanced techniques in machine learning, such as color filtering and masking (Fue et al., 2018b).

After sensing the surrounding environment, the robot sensors need to recognize and establish a position within the environment so that the robot can make navigation decisions to

reach its target. The use of machine vision and GPS has been used in agriculture to recognize the position and even help the robot to move in-between or over the rows of crops and turn at the end of the row (Bergerman et al., 2016). The robot needs a quick decision for localization so it can decide to move. In so doing, the simultaneous localization and mapping (SLAM) algorithms are required to achieve the mission (Bergerman et al., 2016).

In a compact robot, it could be useful to use wireless sensors and utilize the Robotic Operating System (ROS) to transmit data between controllers and sensors. However, the wireless transmission may be affected by several features like the radio transmission standard used, data rate, nodes allowed per master controller, slave enumeration latency, the data type to be transmitted, the range of transmission, extendibility, sensor battery life, costs and complexity (N. Wang et al., 2006).

2.4.3 Agricultural Robot Path Planning

Path planning in agricultural fields means the decisions made by the robot to navigate in an agricultural field safely without destroying the plants (Figure 2.3). Path planning also involves a technique to plan for the movement of the manipulators to the target. In other words, path planning is the technique used to utilize the information provided by the sensing unit of the robot to decide on steering and manipulation to accomplish the mission.

There are several path planning algorithms developed for robotics systems, such as grid-based search algorithms (assumes every point/object is covered in a grid configuration (Jaulin & Godon, 1999; Jensen et al., 2012)), interval-based search algorithms (generates paving to cover an entire configuration space instead of grid (Jaulin & Godon, 1999)),

geometric algorithms (find safe path from the start to goal initially (Grötschel et al., 2012)), reward-based algorithms (a robot tries to take a path, and it is rewarded positively if successful and negatively if otherwise (Zeng et al., 2019)), artificial potential fields algorithms (robot is modeled to be attracted to positive path and repelled by obstacles (Qixin et al., 2006)), and sampling-based algorithms (path is found from the roadmap spaces of the configuration space). Each of the algorithms has potential use, and some are just classic methods like grid-based algorithms (Shvalb et al., 2013). However, the most advanced methods are sampling-based algorithms, as they attain considerably better performance in high-dimensional spaces using a large degree of freedom. Since many robots in agriculture will work in swarms to accomplish tasks comparable to the big machines currently used, real-time path and motion planning are required to control and restrict swarm agents' motion (Shvalb et al., 2013).

For plants like cotton, overlapping leaves prevent the robot from seeing clear rows to navigate and move the manipulator (Fue et al., 2019b). This was not the case for large plants like citrus, in which the rows were clear for the robot to move in between and pick the fruit on both or one side of the row (Subramanian et al., 2006). The robot also needs to plan how it is going to move between the row without repeating the same rows using simultaneous localization and mapping (SLAM) and how the arm is going to move without destroying branches (ASABE, 2019).

2.4.4 Agricultural Robot Manipulation

Manipulators and end effectors are tools designed for the smooth operation of the robot on objects and the environment. End effectors consist of the gripper or a certain tool to be impactive (physically grasping objects, like the citrus robot reported by Hannan et al. (2007)), ingressive (physically penetrate the surface of the object, like the soil sampling robot reported by Cao et al. (2003)), astrictive or attractive (suction objects by using external forces, like the tomato gripper reported by Monta et al. (1998)) or contigutive (direct adhesion to the object) (Cho et al., 2002; Naoshi Kondo & Ting*, 1998; Monkman, 1995; Paul, 1981; Rodríguez et al., 2013; Tai et al., 2016). Some robots may use a combination of two or more end effector techniques; for example, Monta et al. (1998) used both astrictive and impactive grippers to improve success rates in tomato picking.

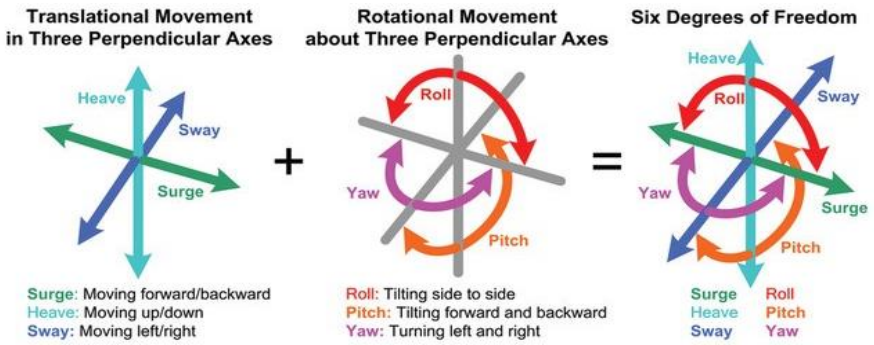


Figure 2.4. Possible movements for robot manipulators (sensing.honeywell.com).

Table 2.3 discusses other agricultural robots designed to work on non-harvesting tasks. N/A means the authors did not report any information related to that category. Most of the robots use GPS, camera, and four-wheel platforms.

Table 2.3. Other agricultural robots for weeding, soil sampling, scouting/phenotyping, pruning, spraying, and sowing.

Activity	Reference	Mobility	Sensing	Path Planning	Manipulation
Weeding	(Bakker et al., 2010; Bakker et al., 2006)	Four-wheel vehicle	Camera, GPS, and angle sensors	Hough transform method for detection of rows	N/A
	(Bak & Jakobsen, 2004)	Four-wheel vehicle	Camera, GPS, gyroscope, magnetometer	Strategic planning (based on previous knowledge of weed population), adaptive planning (for the unexpected occurrence of weeds) and path tracking control	N/A
	(Kim et al., 2012)	Continuous track vehicle	IMU and LRF	Path Tracking methods	The inter-row spacing weeder was made of three spiral-type cutters (three arms and three

Activity	Reference	Mobility	Sensing	Path Planning	Manipulation
					weeder plows) [2DOF]
Pruning	(Haruhisa et al., 2008)	Four active wheels are set at regular intervals around the tree	N/A	Climbing method (implementing rotation of wheels along the vertical direction and diameter of the trunk).	2DOF (with cutting blade)
	(Devang et al., 2010)	Two active wheels	N/A	Climbing method (implementing rotation of wheels along the vertical direction and diameter of the trunk). Arm trajectory motion planning with a search mechanism	9DOF (with cutting blade)

Activity	Reference	Mobility	Sensing	Path Planning	Manipulation
	(Botterill et al., 2017)	Four-wheel vehicle	3D cameras	The randomized path planner [random tree (RRT)-based planner, RRT-Connect]	6DOF (cutting tool consists of a router mill-end attached to a high-speed motor)
	(Ueki et al., 2011)	Four active wheels	3D position measurement device and 3D orientation sensor	Innovative climbing strategy [grid based]	2DOF
Soil Sampling	(Cao et al., 2003)	Two-wheel robot	GPS, encoder	GPS path tracking [Adaptive grid-based Navigation]	2DOF (Linear actuator and Cone penetrometer)
	(Fentanes et al., 2018)	Four-wheel vehicle (Thorvald)	RTK-GPS, force sensor, measurement device, soil moisture sensor	GPS tracking method [grid-based]	2 DOF (penetrometer)

Activity	Reference	Mobility	Sensing	Path Planning	Manipulation
	(Scholz et al., 2014)	Four-wheel vehicle (BoniRob)	RTK-GPS, and soil moisture sensor	GPS tracking method [grid-based]	2 DOF (penetrometer)
Scouting or phenotyping	(Kicherer et al., 2015)	Four-wheel vehicle	RTK-GPS, NIR camera, and RGB Multicamera system	GPS Auto steering methods	N/A
	(Salas Fernandez et al., 2017)	Four-wheel tractor	RGB Stereo camera, RTK-GPS	GPS Auto steering method	N/A
	(Obregón et al., 2019)	Four-wheel tractor	GPS, RGB camera, inertial sensors, 3D LIDAR, 2D security lasers,IMU	Simultaneous Localization And Mapping	N/A

Activity	Reference	Mobility	Sensing	Path Planning	Manipulation
	(Young et al., 2018)	Continuous track	RGB Stereo cameras, single-chip ToF sensor, IR sensor, RTK-GPS gyroscope, and optical encoders	Extended Kalman filter (EKF) and nonlinear model predictive control	N/A
Spraying	(Sammons et al., 2005)	Sliding on rails vehicles	Induction sensors, IR sensors, bump sensors	N/A since it was following the rails	N/A
	(Sharma & Borse, 2016)	Four-wheel vehicle	Camera, temperature, humidity, soil moisture sensors, GSM modem	N/A	N/A
	(Nakao et al., 2017)	Four-wheel vehicle	LRF sensor, GPS and magnetic sensor	Path tracking method and self-positioning method	N/A
	(Cantelli et al., 2019)	Four-wheel vehicle	LRF sensor, ultrasonic, laser scanner, stereo	Path tracking using planned trajectory	N/A

Activity	Reference	Mobility	Sensing	Path Planning	Manipulation
			camera, encoders and GPS		
Sowing	(Haibo et al., 2015)	Four-wheel vehicle	Encoder, angle sensor, pressure sensor, IR sensor	Path tracking methods	2DOF (sowing device)
	(Srinivasan et al., 2016)	Continuous track [caterpillar treads]	Magnetometer, the ultrasonic sensor	Navigation by using sensor data to follow rows	2DOF (sowing device)

Manipulators can be identified by their freedom of movement in space. This is known as degree of freedom (DOF) (Figure 2.4) which means the body can freely change the position as up/down (known as heave), left/right (known as sway), forward/backward (known as surge) and it can do orientation through rotation by yawing (around normal axis), pitching (around lateral axis), or rolling (around longitudinal axis) (Paul, 1981). In agriculture, robots have been designed to accommodate various levels of DOF from three DOF (strawberry robot designed by Cho et al. (2002)) to seven DOF (tomato robot designed by Monta et al. (1998)). As DOF increases, flexibility increases, but it may become heavier and slow in response (C Wouter Bac et al., 2014; Naoshi Kondo & Ting*, 1998). In agriculture, high power-weight ratio actuators are more suitable and effectively used (Bergerman et al., 2016).

2.5 Agricultural Harvesting Robotics

We identified several harvesting robots that have been developed and reported that could potentially be used as a template for a robotic system in cotton.

2.5.1 Agricultural Harvesting Robot Mobility and Steering

Most of the reported robots in agriculture above for harvesting were wheeled robots (Table 2.4). Also, these robots have an arm mounted on top of the vehicle moving in-between or over the rows (Table 2.4). Most of the four-wheeled robots reported turn using front tires (Ackerman steering model) (Table 2.4). Some that are deployed in greenhouses use rails, since greenhouses are semi-structured farms (C. Wouter Bac et al., 2017; Qingchun Feng et al., 2018).

Fraunhofer Institute for Production Systems and Design Technology IPK (www.ipk.fraunhofer.de) developed a prototype of a dual-arm robot that navigated by using rails for cucumber harvesting (Figure 2.2b) that was semi-autonomous.

2.5.2 Agricultural Harvesting Robot Sensing

Cotton bolls appear like flowers; hence, any potential flower harvesting robot could be adaptable. The 3D positions of flowers can be obtained using stereotypic cameras (Kohan et al., 2011). Also, Kohan et al. (2011) reported that in stereotypic cameras, increasing the distance between lenses reduces errors, while increasing the distance between the lens and the object (flower) increases error. In harvesting, it becomes more complicated due to the occlusion of the bolls. Ripe fruit may be located inside the canopy, where access can be limited.

2.5.3 Agricultural Harvesting Robot Path Planning

In harvesting, path planning is dependent on the manipulators, end effectors, and the agricultural produce to be harvested. In any case, if the fruit to be gripped is very delicate, then path planning becomes more complicated for impactive end effectors compared to sucking end effectors to avoid collisions that may damage the fruit (Hohimer et al., 2019). Also, the fruit to be sold to consumers is expensively harvested as the robot needs to match human picking action compared to fruits harvested for juice or industrial processing. Most of the heavy mechanical robotic machines may be used to harvest fruits for industrial use since the machines may be fast enough compared to a robot.

If the plant branches are weak or the fruit is very delicate, path planning becomes expensive to preserve the plant that needs to be left undestroyed. Path planning is also expensive when many degrees of freedom (DOF) arm is used. However, most methods for path planning are more effective and successful when the number of DOF is optimized to be small enough to achieve the purpose (Faverjon & Tournassoud, 1987). Also, in multiple arm robots, some machines use a prescription map to harvest many fruits at high speeds (Zion et al., 2014). Hohimer et al. (2019) concluded that by increasing the degrees of freedom, the apple fruit picking robot was performing well but at the slowest speed. This was caused by the path prediction algorithms, and the time the actuators took to reach the target. They advised attempting to use a lower degree of freedom to achieve the speed required to attend large fields like cotton farms.

Most of the robots reported path tracking algorithms to navigate on the farm using GPS and cameras (Table 2.4). Most of the robots used for greenhouse harvesting use rails; hence they do not need navigation algorithms but rather position control algorithms (Table 2.4). Also, most of the studies except Lili et al. (2017) reported motion planning, which is done using arm trajectory motion without including search mechanism algorithms for path planning or obstacle avoidance. However, Noguchi and Terao (1997) introduced path planning in agriculture using advanced methods in neural networks (NN) and a genetic algorithm (GA) in 1997. Also, Zuo et al. (2010) developed a robot path planning system with limited end-of-row space using a Depth-First Search (DFS) algorithm.

2.5.4 Agricultural Harvesting Robot Manipulation

For manipulators, various degrees of freedom (DOF) have been studied, including a three-DOF rectangular coordinate manipulator to the nine-DOF manipulator (Table 2.4). For end-effectors, it mainly depends on the type of farm product to grip and the degree of abrasion that can be tolerated. Impactive, attractive, and contigutive end effectors are the most common, with most of the end-effectors being attractive, impactive, or both. This was because fruits that require robotic harvesting need expensive handling to avoid abrasions (Hayashi et al., 2014). Hence, ingressive end effectors are not common in agricultural harvesting of fruits as most must be pristine for the fresh market.

Manipulators are evaluated using success rates (Bechar & Vigneault, 2017). Cotton boll harvesting needs less than 3 seconds for each boll to be effective (Fue et al., 2019a). Xiong et al. (2019) reported a strawberry robot success rate of 53.9% while Yaguchi et al. (2016) got a success rate of 62.2% on picking tomatoes. Silwal et al. (2017) got a success rate of 84% for apple picking. All the researchers (Table 2.4) that reported the execution time have achieved an execution time of more than 20 secs per fruit. Hence, cotton harvesting cannot adopt the manipulation methods reported, at least without some modification to increase success.

Table 2.4. Recent robotic systems developed for harvesting agricultural produce.

Reference/crop	Mobility	Sensing	Path Planning	Manipulation
(C. Wouter Bac et al., 2017) for Sweet pepper	The railed vehicle robot platform	A ToF camera, RGB cameras	Robot over the rails. Manipulator used Arm trajectory motion planning with a search mechanism	9DOF, Fin Ray end effector (scissors and fingers) and Lip-type end effector (knife and vacuum sensor).
(Lili et al., 2017) for Tomato	Four-wheel vehicle	binocular stereo vision	PID control for Ackerman steering geometry. The manipulator used C-space and the A* search algorithm	5DOF harvesting manipulator
(Xiong et al., 2019) for strawberry	Four-wheel vehicle [Thorvald II]	RGB-D camera, IR sensor	Vehicle controlled manually by a joystick, but manipulator used motion sequence planning algorithm	5DOF arm with a cable-driven gripper

Reference/crop	Mobility	Sensing	Path Planning	Manipulation
(Qingchun Feng et al., 2018) for cherry-tomato	The railed vehicle robot platform	RGB Stereo camera, Laser sensor	Arm trajectory motion planning for the manipulator	6DOF with double cutter end-effector
(Mu et al., 2017) for Kiwi-fruit	Four-wheel vehicle robot	Laser sensors, Hall position sensor, Pressure sensor, Optical fiber sensor	Arm trajectory motion planning without search mechanism	2DOF with 3D printed bionic fingers end-effector
(Zion et al., 2014) for Mellon	The 2-m wide rectangular frame which spans the melon bed robot with four wheels	RTK-GPS, encoders, RBB stereo cameras	Arm trajectory motion planning without search mechanism	3DOF Multiple Cartesian manipulators
(Q. Feng et al., 2015) for Tomatoes	The railed vehicle robot platform	RGB Cameras, wheel encoders, a gyroscope and an ultra-wideband (UWB) indoor positioning system	Arm trajectory motion planning without search mechanism	6DOF manipulator with a 3D printed gripper

Reference/crop	Mobility	Sensing	Path Planning	Manipulation
(Chen et al., 2019) for Apples	Four-wheel vehicle	RGB cameras, wheel Encoders	A visual servo algorithm based on fuzzy neural network adaptive sliding mode control for vehicle and manipulator	5DOF manipulator
(Yuanshen et al., 2016) for Tomatoes	The railed vehicle robot platform	RGB stereo camera	Inverse kinematics for manipulator and no navigation algorithm for vehicle and Arm trajectory motion planning	Two 3-DOF Cartesian type robot manipulators with saw cutting type end-effector

2.6 Cotton Harvesting Robot

Cotton bolls, as seen in Figure 2.5, do not require soft robotics like other fruit crops, which may require very careful design of the end effector and manipulation to avoid fruit damage. The plants are close to each other because the cotton plant tends to fill out spaces as it grows (Ritchie et al., 2007). Most of the bolls begin opening from the bottom of the canopy (Ritchie et al., 2007).

Figure 2.5. The undefoliated cotton field at UGA farms.

2.6.1 Cotton Harvesting Robot Mobility and Steering

Most of the cotton harvesting robots reported use four-wheel vehicles (Figure 2.6, Figure 2.7 and Figure 2.8). Figure 2.7 is a prototype developed in India by a startup owned by Sambandam company. The prototype involves a four-wheel vehicle that is used in small farms in India. However, this prototype was designed to be controlled by human operators for navigation. The same approach (Figures Figure 2.6 and Figure 2.8) of using a four-wheel rover but with center-articulation was proposed in our group as well (Fue et al., 2019a, 2019b; Fue et al.,

2018b; Rains et al., 2014). Currently, no other type of mobility or steering and navigation algorithm or method for cotton harvesting has been reported.

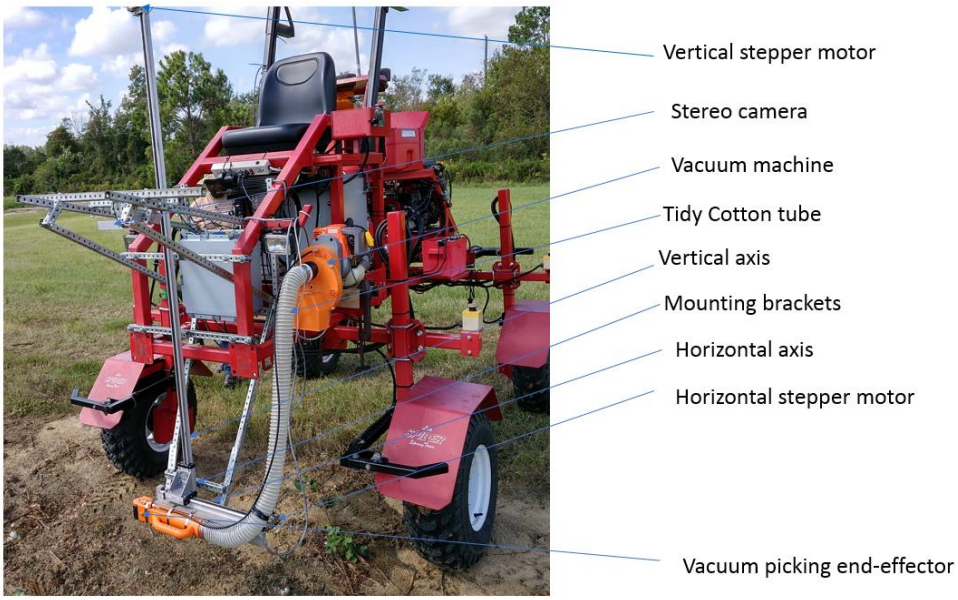


Figure 2.6. The cotton robotic system proposed by our team (Fue et al., 2019a; Rains et al., 2014).

Figure 2.7. Cotton picker robot prototype (source: www.kas32.com).

The use of robots in harvesting cotton faces a complex environment for robot mobility (Figure 2.5). Since plants are very close to each other and leave a thin path, the adoption of accurate autonomous navigation that uses the fusion of sensors like IMUs, GPS, and machine vision becomes a vital requirement. Accurate path following without breaking branches will increase the precision and other metrics of the robotic system. Due to this complexity, Bechar and Vigneault (2016) proposed the use of humans in operating semi-autonomous robots just to increase the productivity and quality of the operation rather than leaving the machines alone. The technology for semi-autonomous or autosteering navigation is also currently available; hence, it can be more easily accepted and adopted. However, autonomous commercial tractors are highly desirable in precision farming because they are cost-effective, can reduce labor requirements, and are safe to humans if designed well. (Fue et al., 2019b) proposed a navigation algorithm for navigating the cotton field by detecting the rows from above. Depth maps are acquired, transformed into binary depth maps, and then rows detected using a sliding window algorithm, which compares the depth of the pixel to differentiate between canopy and land.

Figure 2.8. Cotton picking robot prototype proposed by Clemson University (source: www.agweb.com).

2.6.2 Cotton Harvest Robot Sensing

Cotton is an indeterminate crop and continues to open bolls for a period of approximately 50 days (Ritchie et al., 2007). Hence, there is a need to harvest bolls as they open. Sensing capability should be able to distinguish fully opened bolls from others, and it should be able to detect open bolls located at the bottom of the plant canopy. However, lowering the camera into the canopy could readily destroy the lenses due to plant branches' impact.

Fortunately, the cotton's whitish color gives it a distinguishing feature to be easily detected by a color camera. Also, the cotton recognition algorithm should be able to work well under direct sunlight. There are several cotton recognition algorithms reported using machine vision techniques like color segmentation (Fue et al., 2018b), optimized segmentation algorithm based on chromatic aberration , color subtraction and dynamic Freeman chain coding , region-based segmentation , deep learning methods and ensemble methods (Fue et al., 2018a; Y. Li et al., 2016; Li et al., 2017; Mulan et al., 2008; Y. Wang et al., 2008). All the methods described in these studies can be adopted to improve the current cotton harvesting prototypes. However, it was a challenge to detect separately occluded bolls using color segmentation (Fue et al., 2018b; Y. Wang et al., 2008).

Our group designed a cotton detection algorithm using a stereo camera that was able to precisely locate and track cotton bolls using deep learning (Fue et al., 2018a). A stereo camera (ZED) was used to estimate boll positions and found the mean error standard deviation increased as the speed of the rover and installed camera increased to 0.64 km/h (Fue et al., 2018b). The robot was performing well with 9 mm RMSE and an average R2 value of 99% when stationary,

but when the vehicle started moving to approximately 0.64 km/h, the R2 dropped to 95%, and RMSE increased to 34 mm (Fue et al., 2018b). It was the only study that has demonstrated the detection and estimation of the location of cotton bolls in field conditions in real-time using an embedded system.

2.6.3 Cotton Harvest Robot Path Planning

With high cotton boll occlusion, path planning for navigation and manipulators becomes a very crucial requirement for the successful deployment of a commercial agricultural robot (Ramin Shamshiri et al., 2018). There was no research seen that describes path planning for a cotton harvesting robot rather than navigation planning of the robot along the rows (Fue et al., 2019b). However, it seems the current researchers do not see the necessity to develop a commercial product in a non-specialty crop that demands good path planning. Most of the designed path planning algorithms in agricultural robots use IMU, camera, and RTK-GPS (Table 2.3 and Table 2.4). Hence, cotton harvesting systems may adopt this approach too. If small robots are adopted in cotton harvesting, navigation between the rows using Lidar has been shown to be successful (Higuti et al., 2019).

The cotton field environment is highly unpredictable due to varying plant canopy growth patterns. Cotton crop canopy grows to fully cover the space between the rows, and it can grow very tall (Ritchie et al., 2007). Planting practices, especially plant spacing, requires special recommendations for robotic harvesting. This could be done in cotton by modifying farm management practices or by manipulating the genes of the crops to allow easy access to the bolls for robotic manipulators. This is common for specialty crops like apples, strawberries, and grapes, which were bred to provide effective access to fruits.

2.6.4 Cotton Harvest Robot Manipulation

Current approaches to grippers are not effective for cotton plants because the cotton boll fibers stick on the end effector. So, grippers need to be strong enough to grab the boll effectively without destroying the plant. With harvesting as bolls open comes a challenge to design manipulators which start harvesting bolls at the bottom of the plants, and that are highly occluded by the canopy. The reported cotton harvesting manipulators and end effectors picked the cotton boll, but they also broke the plant branches, removed leaves, or knocked down unharvested bolls to the ground (Fue et al., 2019a). Therefore, a well-designed astrictive or attractive method is desirable for cotton harvesting. Figure 2.6 shows a prototype of a cotton harvesting robot with the two-DOF cartesian manipulator that holds a vacuum suction end effector (Fue et al., 2019a). Figure 2.11 shows a Clemson-developed cotton harvesting prototype robot that uses a two-DOF cartesian manipulator. Figure 2.9 and Figure 2.10 present a gRoboMac prototype robot that uses a three-DOF manipulator and four-DOF manipulator, respectively. All the reported manipulators in cotton harvesting use astrictive or attractive grippers since cotton lint does not require careful handling like other fruits (Fue et al., 2019a).

In 2019, a team in India designed a rigid vacuum cleaner machine as the best alternative for a cotton harvesting end effector (Figures 2.9 and 2.10). The gRoboMac team did not report execution time, which was a very important parameter for effective cotton harvesting. Fue et al. (2019a) obtained a preliminary execution time of 17 seconds per boll. Both groups (Fue et al. (2019a) and gRoboMac) reported manipulators that used two-DOF and four-DOF manipulators, respectively. Simple manipulators have a high execution time (Hohimer et al., 2019). For example, Hohimer et al. (2019) reported that the eight-DOF apple harvesting robot was 10,000 times slower compared to the five-DOF manipulator robot. However, the eight-DOF robot was

flexible to reach most of the fruits hence provided high success rates. A Clemson University team (Figure 2.11) also proposed a similar approach but using a small rover riding in between the rows. Fue et al. (2019a) reported the use of a vacuum end effector with rotating tines to remove bolls (Figure 2.6), which has been widely used by humans to pick cotton in China and other developing countries. Fue et al. (2019a) modified the system to be used in robotic systems.

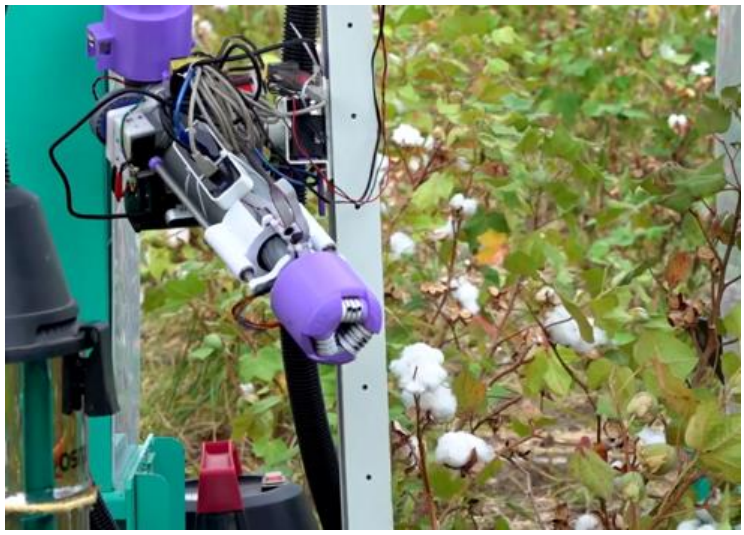


Figure 2.9. Green Robot Machinery (gRoboMac) manipulator trying to get the cotton boll (source: www.grobomac.com).

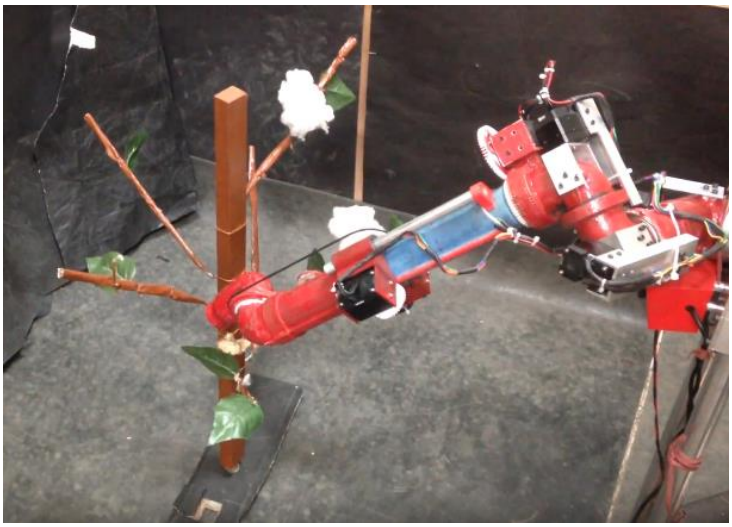


Figure 2.10. Old design of Green Robot Machinery (gRoboMac) manipulator trying to get the cotton boll (source: thetechpanda.com).

Figure 2.11. Cotton robot testing at Clemson University (source: agweb.com).

2.7 Challenges in Commercial Deployment of Agricultural Robots

The initial investment in row crop robotics systems may become very big for an average farmer (Lowenberg-DeBoer et al., 2019). As much as USD 319,864 for an 850 ha farm is required for investment in intelligent machines to achieve maximum break-even point (Shockley & Dillon, 2018). Fortunately, Shockley and Dillon (2018) and Pedersen et al. (2008) concluded

that farming robots would bring profitable business to farmers because robots can reduce 20% of the scouting costs for cereals, 12% for sugar beet weeding, and 24% for inter-row weeding. Robots can work like a swarm of small robots to accomplish farm operation at a very competitive cost compared to current machines (Gaus et al., 2017). Non-horticultural crops like maize, soybean, barley, potato, wheat, and cotton have not been given priority in economic studies on robotic systems after evaluating several studies in databases such as GreenFILE, Business Source Complete, AgEcon Search, Food Science Source, Emerald, CAB Abstract, and ScienceDirect (Shockley & Dillon, 2018). Fortunately, the same challenges in agricultural robotics cut across different farming operations and crops.

There are five commercial parameters, and at least one of them should be unlocked for agricultural robotics to succeed (Bechar & Vigneault, 2016). Firstly, the cost of the new robot should be lower than the current methods used. Secondly, the introduction of robots should increase the capability, productivity, quality, and profitability of production. Thirdly, the introduction of robots should increase uniformity and quality in farm production and decrease variability and uncertainty. Fourthly, the use of robots may increase and fasten farm management decisions that are not able to be achieved by the current methods. Lastly, the use of robotics should remove human beings from operating on environmental risky tasks, particularly the use of heavy machines and chemicals, hence reducing labor and insurance payment for labor. Also, there are other factors that can be indirectly important for farmers, such as the ease of use and maintenance of the robot compared to current methods and reduction in soil compaction (Bechar & Vigneault, 2016).

The design of the manipulators may also be a great challenge in the agricultural field. Single-arm robot design also may not be effective for large farms. However, the challenge of

agricultural robotics with more than three DOF has been “sensing and moving” at rapid harvesting rates (Ramin Shamshiri et al., 2018). It has been a challenge for on-the-go field harvesting due to the robotic arm moving the branches of the target; hence, camera feedback was necessary to determine the latest position of the target before harvesting by the manipulator (Ramin Shamshiri et al., 2018). So, it was concluded that research and development of commercial harvesting systems should concentrate on increasing the speed and accuracy of robots in a harsh and varying environment.

The current research in the cotton harvesting robot our team is developing provides a MIAR prototype for cotton production. To our knowledge, no research has been conducted to develop robotic systems for other cotton operations, as seen in Table 2.3. An MIAR that would work on multiple farming tasks like sowing, spraying, weeding, scouting, and soil sampling would be useful. Cotton Inc has committed itself to funding robotic systems research in cotton and emphasizes the adoption of open-source robotics. Open-source systems have the advantage of open collaboration, multiple partners, and continuous updating. The Robotic Operating System (ROS) is a good example of open-source adoption and continuous improvements and additions through the community of open-source users (Koubâa, 2017). Thus, open-source creates a harmonized environment for researchers that is cost-effective and can speed up development efforts. It also encourages reuse of the core libraries in the development of robots, hence reducing costs and enabling more cost-effective commercialization of robotic platforms (Koubâa, 2017). The robotics industry is a profitable industry to engage in now. In 2019, the IDTechEx research company analyzed the robotic market and technology development growth and predicted the agricultural robotics industry would be worth $12 billion worldwide by 2027 (Ghaffarzadeh, 2019). There is an advantage of using robots as the economics models show that

the net returns can increase by up to 22% compared to the current practice of using conventional machines in row crop production (Shockley & Dillon, 2018).

2.8 Conclusion and Future Work

In this paper, we performed a literature review on robotics in agriculture. We have looked at the relationship and similarities of the robotics systems in agriculture that can accelerate the development of cotton harvesting robots. We also examined aspects of mobility, sensing, path planning, and manipulator design. Our aim in this study was to highlight the recent opportunities and challenges of agricultural systems and the promising future of cotton harvesting robotic systems.

Sensor development for machine vision is advancing quickly, and commercial products that support sensing have also been realized. Despite modern technological advancement, the algorithm to allow a smooth interpretation of visual sensing is still a challenge in agricultural fields (Kamilaris & Prenafeta-Boldú, 2018). The sensitivity, aperture, and resolution are improving, and the present technologies in deep learning have surpassed human eye accuracy in object classification and identification (Szegedy et al., 2015). Machine learning, especially deep learning algorithms, has brought high accuracy in the identification of weeds, plant cultivars, fruit counting and estimation, land cover classification, and crop and fruit type classification (Kamilaris & Prenafeta-Boldú, 2018; Liakos et al., 2018; Szegedy et al., 2015). Most of the navigation and motion planning algorithms to navigate in row crops do not provide fully autonomous capability compared to tree crops (Kamilaris & Prenafeta-Boldú, 2018; Liakos et al., 2018). Cotton needs color sensors to differentiate open bolls from semi-open bolls and flowers during harvesting.

Mobility in a cotton field may use four-wheel-drive systems to increase the speed of harvesting as reported by some researchers because cotton fields are big and require speedy and long navigation. Trained robots cannot be used since cotton is produced on outdoor farms. However, it has shown good adoption in greenhouses. Path planning is needed in four-wheel-drive systems because the robot needs to pass over the rows carefully so as to not break branches or knock cotton bolls onto the ground.

Manipulators have shown good performance when fewer degrees of freedom are used. However, for the careful handling of fruits, more degrees of freedom are required. This is not the case for cotton plants, for which the fruit is the lint. The grippers may just use astrictive or attractive grippers without destroying the lint. This is the main reason most of the research in cotton harvesting has focussed on two-DOF, three-DOF, and four-DOF manipulators.
Future designs of cotton harvesting robots need effective manipulators and sensing that can locate and pick cotton bolls located at the bottom of the canopy. Designs that involve multiple manipulators will provide fast harvesting that can match current harvesting machines. Manipulators that use fewer degrees of freedom will provide fast picking of cotton, which is critical to get to one boll every 3 seconds. Future design and development research should also include alternative energy sources to decrease energy costs. Studies to determine power requirements, footprint, and cost are necessary for robots to be developed for multipurpose functions and work in collaborative “swarms.”

CHAPTER 3

ENSEMBLE METHOD OF DEEP LEARNING, COLOR SEGMENTATION, AND IMAGE TRANSFORMATION TO TRACK, LOCALIZE, AND COUNT COTTON BOLLS USING A MOVING CAMERA IN REAL-TIME[2]

[2] Fue, K., Barnes, E., Porter, W., and Rains, G., Submitted to *Transactions of ASABE*, September 13, 2018.

3.1 Abstract

In robotic applications, good perception may be computationally costly and create undesirable latency before a control decision is initiated. Most of the object detection deep learning methods available are either fast with low accuracy or slow with high accuracy. Fast and accurate methods are both necessary to track and localize objects like cotton bolls that may be visible or occluded by each other or not well illuminated to be detected. In this study, an ensemble of a deep learning method and other image processing techniques were used to detect cotton bolls infield on defoliated plants. In each image, a trained deep learning method, the YOLOv2 model was used to detect open cotton bolls, and color segmentation was applied to confirm if the bolls detected by the YOLOv2 model were actually white to avoid false positives. Boll tracking was performed by following the spatial movement of the good features on the edges of the bolls using the Lucas-Kanade algorithm. An image transformation algorithm was applied to the next image in case the boll previously detected was lost to retrieve the information of the missing boll. Each boll tracked and localized was stored and counted to give the total number of bolls detected. In this study, detection accuracy was sacrificed for image processing speed by using the YOLOv2 deep learning model. Detection accuracy was improved by using an ensemble method that combined image color segmentation, optical flow, and an image transformation technique. This method was compared to eight other open-source methods implemented in OpenCV. The ensemble method detected and counted the bolls at a speed of 7.6 fps with an accuracy of 94.4% using the Jetson TX2 embedded system to process 1K resolution images, outperforming the other OpenCV methods in various measurements.

3.2 Introduction

Cotton is a significant crop in the United States that utilizes large and expensive machines to harvest in a once-through system at a time when many of the bolls have been harvestable for several weeks. Consequently, farmers suffer losses in quantity and quality because of the indeterminate ripening of the cotton fruit (boll) (UGA, 2018). Current technologies that are heavy, expensive, and difficult to maintain can be unprofitable as the current cotton picker alone can cost over $750k (Fue et al., 2018b). For cotton production to remain competitive, it is imperative to adopt new modern technologies that are more cost-efficient for farmers. Machines for harvest are required to be relatively inexpensive, scalable to the size of the farm operation and developed to harvest as the bolls open to reduce losses and preserve cotton quality (Fue et al., 2018b; UGA, 2018). Smaller machines do not increase compaction, destroy plant branches carrying bolls that open later, and should discriminate bolls ready to pick from the cotton canopy early in the season. A possible approach is to develop small, but useful, robots that can be deployed as an "army of bots" to harvest open bolls continuously. These robots could be developed such that they are also able to harvest a diverse number of crops and remain active for longer periods of the year. The need to selectively collect cotton bolls in space with a robotic system requires very effective research on machine vision algorithms that will guide the end effector of the robotic arm (Bloch et al., 2017).

The current 6-row picking technology navigates at a speed of 8.9 km/h in the field and covers around 3.2 to 4.0- Ha (8 to 10 acres) per hour. 101.6-cm (40-inch) and 76-cm (30-inch) row spacing yield approximately 75 bolls per meter square of cotton (Boman, 2012; Prostko et al., 2018). Assuming with small machines that collect at least 12,140 bolls per trip for 25 times per harvest cycle will cover 303,500 fresh open bolls. For a small machine moving while picking

one boll every 3 seconds at 100% field capacity, it could work for 10 hours in 101.6-cm rows to finish harvesting 4046-square meter in 50 days. Estimating a harvesting field efficiency of 60% (refueling, cotton unloading, turning at row ends, maintenance, actual picking, and other downtimes), it is estimated that a robotic harvesting machine with one robotic arm would be capable of harvesting 7284 bolls per day for 10 hours. This achievement depends on real-time detection and tracking of the bolls, optimized power solution, development of cotton unloading travel speed of the robotic harvesting machine, and the design of the manipulator/end-effector. In order to increase the speed of robotic harvesting to match the 25 harvest dates, it is currently assumed that two robotic arms per harvester would be necessary.

This study addresses the real-time detection, localization, and tracking of cotton bolls from a moving camera. The robotic cotton harvester must perceive the horizontal and vertical distances from the camera and manipulator to the cotton bolls in real-time. This characteristic is fundamentally vital for machine vision systems to locate the bolls and store locations while directing a robotic manipulator for picking. Previously, color segmentation and other image processing approaches were used to determine the boll locations in real-time (Fue et al., 2018b). Localization of the cotton bolls by tracking their position provides the robotic system with prior information and improves detection and tracking speed.

Color segmentation is challenging in field conditions with profoundly changing illumination and dense occlusion while using a moving camera. Segmentation makes the color selection an arduous task (Cheng et al., 2001; Gauch & Hsia, 1992; Li et al., 2017). In agriculture, color segmentation techniques can improve image detection and classification. Still, they are limited in discriminative power and cannot differentiate bolls whenever the color appears similar or occluded by each other, a common field occurrence (Choi et al., 2015; Choi et

al., 2016; Fue et al., 2018b). So, it is imperative to investigate alternative methods for detection, such as deep neural networks, which have proved to be very useful, even in challenging lighting conditions (Kamilaris & Prenafeta-Boldú, 2018; Redmon et al., 2015; Redmon & Farhadi, 2016).

In this research, supervised convolutional deep learning neural network architecture was employed to detect and recognize the cotton bolls using the state of the art detector, YOLOv2 model (Redmon, 2016; Redmon et al., 2015; Redmon & Farhadi, 2016). A similar investigation using deep learning technologies to detect cotton bolls was done before and proved to outperform most of the existing methods of cotton boll classification (Li et al., 2017). However, the approach used a slower technique called the deep, fully convolutional neural network to do semantic segmentation. For robotic harvesting, robust real-time detection of bolls is required to reduce latency between detection and sending a control signal to the cotton-picking manipulator. Therefore, the primary objective of this study was to detect, track and count cotton bolls using an ensemble of deep learning and image processing methods to achieve acceptable accuracy and speed of processing in an embedded system.

This study primarily contributes to cotton boll tracking by introducing an alternative tracking method using the Lucas-Kanade algorithm and image transformation. Also, a technique to select good features to track that is near the image transformation prediction is implemented to improve the accuracy of tracking when the boll is not detected in time of occlusion or false negatives. This method is dubbed as CottoTrack. CottoTrack was compared with other tracking methods present in OpenCV to validate its performance. This study does not address leaf detection, but only bolls in defoliated cotton. Future studies will address additional field conditions such as leaf cover and lighting. Hence, the specific objectives were:

1. Develop a model to detect cotton bolls in real-time using an embedded computing system

2. Track, localize and count the cotton bolls from a mobile camera using the model developed in objective 1.
3. Compare the method with the tracking methods implemented in OpenCV.

3.3 Materials and Methods

3.3.1 Experimental Set-up (Training Dataset)

An experiment was conducted at the University of Georgia (UGA) Tifton campus grounds (N Entomology Dr, Tifton, GA, 31793) at (31° 28'N 83° 31'W). The chosen location was open to direct sunlight to simulate field conditions (Figure 3.1). Twenty-four defoliated cotton

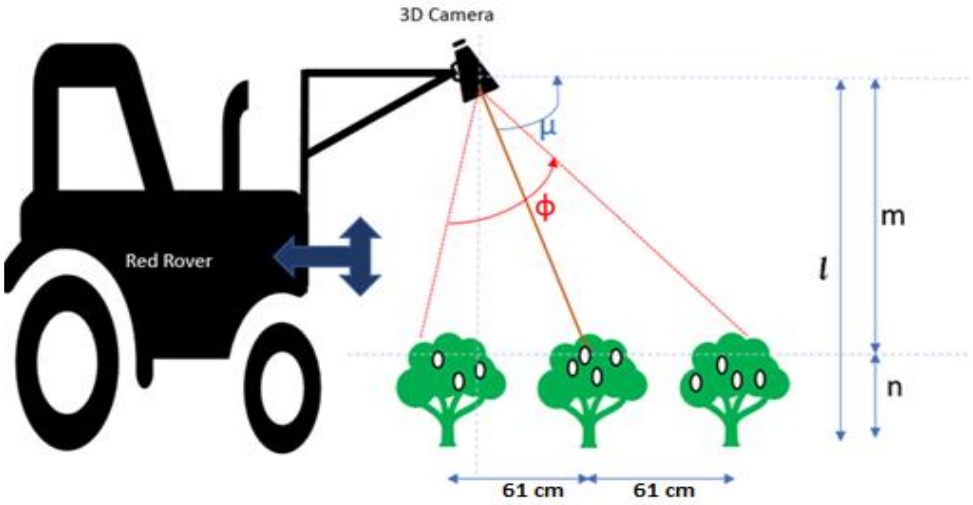


Figure 3.1. The context and view of the experimental setup at the UGA Tifton grounds. The camera platform was mounted on the rover, and the camera was pointing downward.

plants were cut from a nearby farm and put in 25 cm dia. pots. Twelve plants were placed in 2 columns of 6 rows, 91.4 cm between the center of the stalk, and each stalk 61 cm from the next (Figure 3.1).

This experiment simulates the field's condition of the farm with a row spacing of 91.4 cm (36-inch), which is a common practice in Georgia cotton production. Plants were placed 61 cm

apart in the row. This setting is to validate the algorithm before deploying the robotic system to harvest.

A moving camera (<4kph) Samsung Galaxy S6 Edge Plus (Samsung Electronics, Seoul, South Korea) took 720p images of the plants while facing downward at the rate of 30 fps (Figure 3.2) at least 1.5m above the ground. Samsung camera has an f/1.9 aperture and a 16 Megapixel lens. Other training images were taken using a ZED camera (Stereo labs Inc, San Francisco, CA, USA) that was attached to an embedded system (NVIDIA Jetson TX2 development kit, Nvidia Corp., Santa Clara, CA, USA) and mounted on a research rover moving 1-3 kph (Figures Figure 3.1 and Figure 3.2) (Rains et al., 2015). The ZED camera will be used by the rover to detect the bolls and pick cotton with a robotic arm. The ZED camera provides stereo services to locate the position of the bolls and robotic arm in 3-D space. It was used so that the deep learning algorithms would be trained using images from the camera that will be deployed to the robotic system in the future study. The ZED camera was at least 1.5m above the ground taking 720p images as the rate of 15 fps. The camera was tilted (facing downward) at an angle of 90^0 and then, 81^0 (Figures Figure 3.1 and Figure 3.3) to collect training images respectively in two passes. Obtaining variable angles with the Samsung camera and a mixture of images from different cameras was used to enhance the robustness and detection capability of the developed algorithm. Also, the image resolutions (360p, 720p, and 1080p) were set by changing camera settings. All the images were used to train the deep learning algorithm.

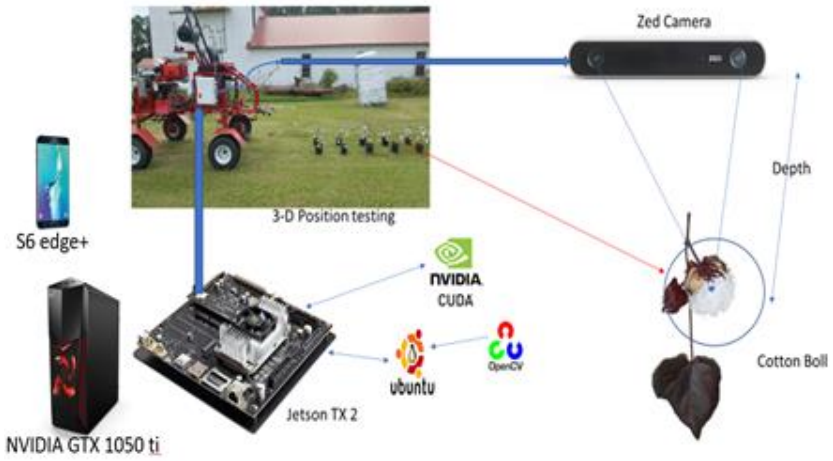


Figure 3.2. The context diagram is demonstrating all the imaging platforms for training and testing the data set

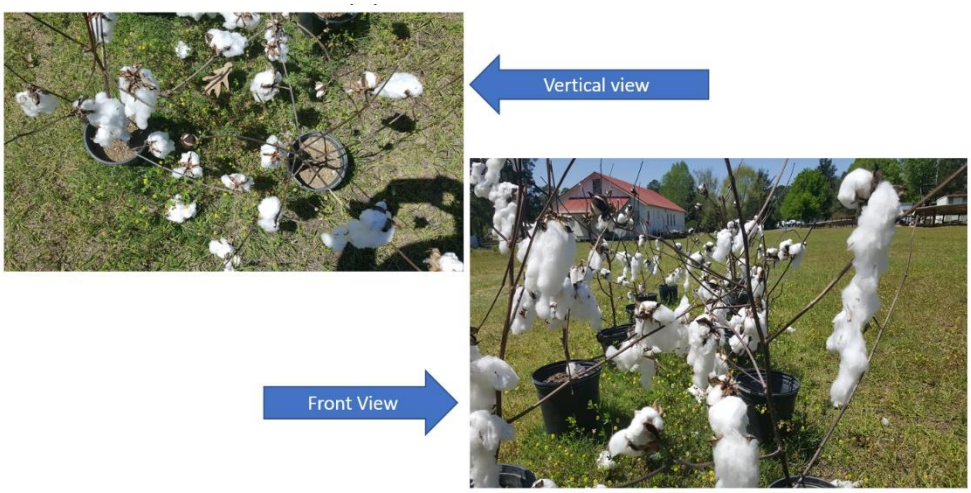


Figure 3.3. Cotton view using the vertical image and a horizontal image. The vertical image was used to view the cotton bolls for tracking and estimation of the boll position while front view or horizontal view shows the cotton bolls vertical dimensions

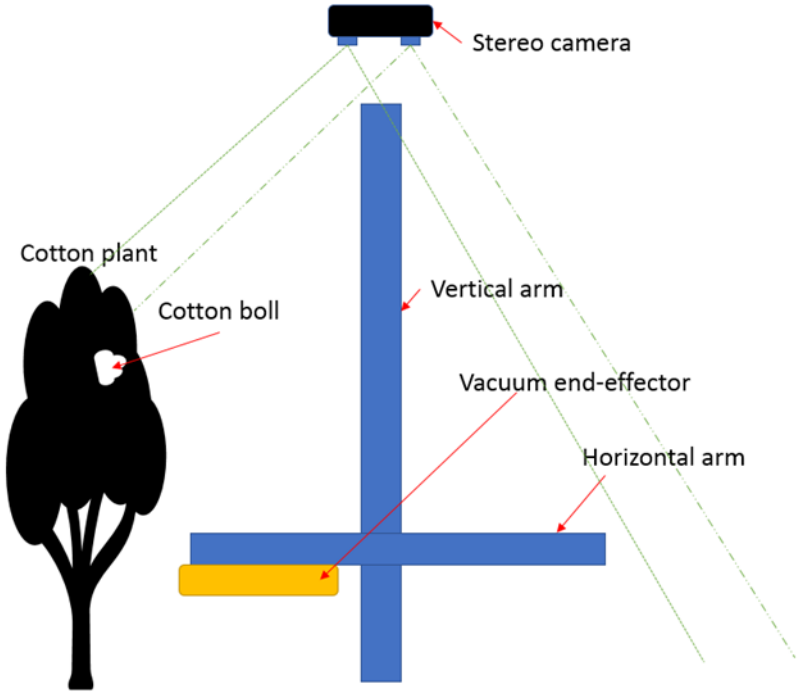


Figure 3.4. Robotic Cartesian Manipulator Context Diagram showing the rover arm, end effector and cotton boll position for the envisioned robotic system design

3.3.2 Image Processing System and Extraction

The Samsung Galaxy cell phone videos were transferred to a desktop Lenovo Legion Y520 (NVIDIA GeForce GTX-1050 Ti graphics running an Intel i5 7th generation CPU) with Ubuntu 16.04 installed (Figure 3.2). The images obtained using the ZED camera were also transferred to the Lenovo desktop that had TensorFlow 1.9 (tensorflow.org) framework installed. Training data and testing data were both loaded to the Lenovo. Later, videos were loaded onto the Ubuntu operated Jetson TX2 embedded system.

3.3.3 The procedure of Data Training and Testing

Twelve plants were used for training, and another group of twelve plants was used for testing. Four hundred eighty-six images with a total of 7498 bolls were annotated for training. The training images were of different resolutions from 360p (31 images), 720p (404 images), to

1080p (51 images). Images of varying quality provide the convolutional neural network (CNN) model with data to learn different image size challenges. Also, to improve the model detector, the images were taken at different times of the day. Images were taken during the morning (212 images), afternoon (121 images), and evening (97 images) at 9:00 AM, 3:00 PM, and 6:00 PM, respectively. Noon was avoided to remove images that had strong glare due to light reflection on white cotton bolls. Some images taken during morning and afternoon were occluded to create shadow and varying illumination over the bolls. Most of the images were taken from defoliated plants. A few images from varied Internet sources (56 images) were also included in the training dataset. The YOLOv2 detection algorithm used the following configurations; The batch size (number of samples processed before the model is updated) was 32, subdivisions (mini-batches) 4, 30 filters (the learned weights of the convolutions), and 2000 epochs (cycles through the full training dataset) were enough for training (Redmon & Farhadi, 2016). The training procedure was clearly described online in the darkflow page(Trieu, 2018). The model reached an accuracy of 75.3% after 2000 epochs (Table 3.7).

Later, the images were transferred to a desktop computer with TensorFlow installed, which was used to train the network deployed in this study. The computer was installed with Darknet, which is a neural network framework implemented in C and CUDA, developed explicitly for YOLOv2 (Redmon, 2016). DarkFlow framework, which translates the pretrained weights from the Darknet to TensorFlow, was also installed. DarkFlow enables Darknet to work with YOLOv2 in TensorFlow. It is not resource-effective to install another Deep Learning framework in an embedded system, so, Tensorflow was used instead of the Darknet framework. YOLOv2 classification is based on modified Darknet-19, which has 19 convolutional layers and five max-pooling layers while a smaller version, tiny YOLOv2, has 9 convolutional layers and

six max-pooling layers. (Redmon & Farhadi, 2016). The images were manually annotated using the Labeling tool called LabelImg and trained using tiny YOLOv2 weights, as explained online in the darkflow webpage (Trieu, 2018). The model was trained to detect only the cotton bolls. The model was then frozen and transferred to the Jetson TX2 for detection and tracking experiments. Freezing a model is a process to identify and save all of the required components of a model like a network graph and weights into a single file that can be easily exported to other systems for inference (Redmon, 2016; Trieu, 2018). The model used tiny YOLOv2, which is too shallow and thin to apply quantization without heavily affecting its accuracy. So, the frozen model was shipped without quantization.

3.3.4 Detection of The Cotton Bolls

A program that utilized the frozen model to detect and predict the location of the cotton bolls was developed. The YOLOv2 predicted the probability of 0 to 100% for every detection of a cotton boll. So, the algorithm was set to consider any detection with more than 50% to be cotton boll. It was also set to use only 40% of the memory. The 40% slot memory ensures the system cannot crash due to full memory problems, but also has enough memory to speed up its predictions while it does other operations.

After model prediction, the software confirmed if the detected bolls were white by differentiating the bolls color from the background using the color segmentation algorithm. The color segmentation task involved four steps (Fue et al., 2018b; Gong & Sakauchi, 1995):

- Collect an image frame,
- Using the RGB color threshold, separate each RGB channel of the image. For cotton bolls, the white components of the image were masked (All color threshold red, green, and blue channels were set above 170 except red, which is 150). The chosen threshold

was sufficient because it was rigorously chosen after testing multiple images using different thresholds.

- Mask or remove the image background from an original image.
- Get all the regions where the contours cover the white parts of the objects detected.

Table 3.1. Algorithm describing the detection of false positives and removal

Algorithm 1: Remove the false positive detections

Input: current video frame, prediction results of the YOLOv2 model [Oj]

Output: Current, correct matches of the bounding boxes [Ci]

```
lower <- [170, 170, 150]
upper <- [255, 255, 255]
mask <- compute the range between upper and lower bounds from the current frame
FOR EACH Oj in [Oj]
    boll <- get the bounding box mask[Oj]
    nzCount <- get non zero points that correspond to boll
    w <- calculate width from Oj
    h <- calculate height from Oj
    area <- calculate area from Oj
    per_nz <- calculate the percentage of non-zero nZcount over the area
    IF per_nz is more than 25%
        Assign a new match Ci
    END IF
END FOR
Return all the matches [Ci]
```

In Table 3.1, the pseudocode describes the detection of the white parts of the frame and compares them with the YOLOv2 predictions to make sure that the boll detected fills at least 25% of the bounding box. This algorithm is run every five frames to detect new bolls that entered the camera scene. It can be run with lower or higher frequencies, but the performance and speed of the algorithm can be affected. After detection of the bolls, the system gets good corner features of the boll to track it.

3.3.5 Detection of Good Features to Track

The pseudocode in Table 3.2 shows the steps to detect the "good features to track" using the OpenCV "goodFeaturesToTrack" method. The method tries to get the corner points (Tables Table 3.5 and Table 3.6) of the boll so that it can track the boll accurately (Figure 3.5). It gets the boll and its mask to make sure the system can only see the boll and its adjacent edges to obtain good features to track (Shi & Tomasi, 1994). These features were used by the tracker to obtain the next position of the boll in the image.

Table 3.2. Good Features to track that are within the boll

Algorithm 2: Good Features to track within the boll
1: **Input**: matches {Ci}, masked frame with bolls (mask), and areas close to it.
2: **Output:** Objects identities of [B_k]
3: p <- Compute the good features to track from 500 maximum corners with the quality of at least 30% and a minimum of minimum Distance 7pixels and block size 7 pixels from the masked frame with points on the boll
4: IF p is not NULL:

```
5:      FOR EACH Ci in [Ci]
6:          for x, y in p
7:              xi,yi,xj,yj <- Ci
8:              area <- compute area by (xj-xi)*(yj-yi)
9:              IF ( xi<= x AND yi <= y AND xj >=x AND yj >= y):
10:                 Register objects to track Bk <- ((x,y,xi,yi,xj,yj,area))
11:                 break
12:             ENDIF
13:         ENDFOR
14:     ENDFOR
15: ENDIF
16: Return all the object identities [ Bk ]
```

Figure 3.5. The boll that has been detected by YOLOv2 will be color segmented to get the edges of the boll (red) and then use the Shi-Tomasi method to get corner features that are located within the boll. At least one feature is specifically targeted and tracked by the model developed

3.3.6 Tracking of the Boll and its Features

The tracking points (features) were calculated using Lukas-Kanade (LK) Optical flow algorithm (Table 3.3) that was implemented in OpenCV. Only the points that were successful in being tracked were used to make the next trajectories (tracklets) of the object's path. Any of the points that were not calculated were passed to be calculated using homography transformation.

Table 3.3. Tracking of the cotton bolls using the optical flow algorithm

Algorithm 3: Tracking of the cotton boll using optical flow algorithm

1: **Input:** Previous frame, current frame, objects to track [$\overline{B}_k$]

2: **Output:** Objects with added track [B_k]

3: [$\overline{p}_k$] <- get only first two values from [B_k]

4: [$\overline{p}_k$], [status], [errors] <- compute the Lukas Kanade Optical flow of points [$\overline{p}_k$] from current image and previous image.

5: FOR EACH B_k, $\overline{p}_k$, status in [B_k], [$\overline{p}_k$], [status]

6: cx,cy,xi,yi,xj,yj,area <- get the contents of 1st element of B_k

7: x, y <- get the contents of $\overline{p}_k$

8: xi <- xi + (x - cx)

9: yi <- yi + (y - cy)

10: xj <- xj + (x - cx)

11: yj <- yj + (y - cy)

12: area <- compute area of the bounding box (xj-xi) * (yj-yi)

13: b <- assign the object with values of the new bounding boxes (x,y,xi,yi,xj,yj,area)

14: IF status is true that is the point was well tracked by LK method

15: Assign the new value of the B_k with the b

16: IF the length of the object that tracks the boll is very long (more than 50)

17: Delete the tail of the tracking object

18: ENDIF

19: ENDIF

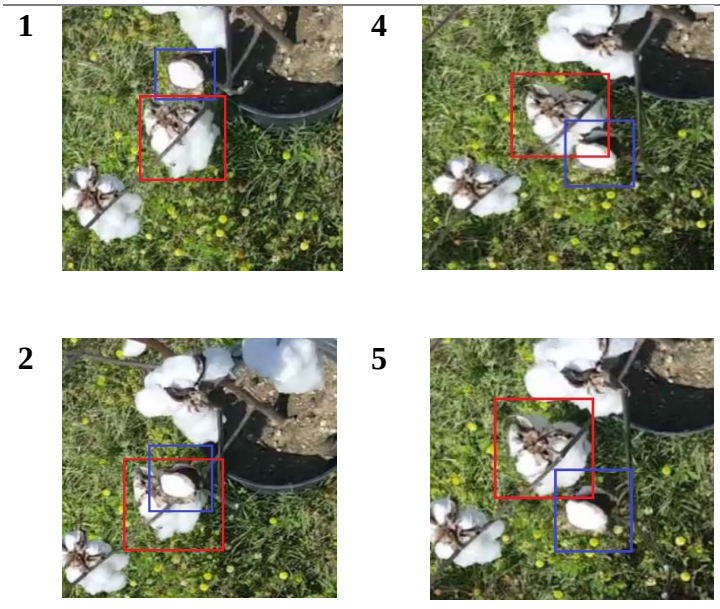
20: ENDFOR

21: Return all the identities of the objects [$\overline{B}_k$]

The transformation was achieved by finding a 3 by 3 matrix of the image perspective transformation using the "findHomography" OpenCV method. The points that had good status in the LK method were used to calculate the homography matrix. Using the previous frame and current frame and RANSAC method, the transformation matrix (H) was obtained (Table 3.4). The matrix was used to transform the bounding boxes and features to track. The bolls that occlude each other were tracked by the transformation matrix as the good features to track may have disappeared for occluded bolls. Hence, the system will force the bounding box to appear while it was not sure if the boll was present. This box was the temporary tracking of the boll. It should be discarded if the boll was not found for five consecutive frames (Algorithm 4). The consecutive frames may be increased but can make the model slow to process as the occlusion was very common in this camera view from above. Figure 3.6 shows the bolls tracked together with the tracklets, which demonstrate the first detection of the boll and consecutive appearances in the coming frames. Figure 3.8 shows two bolls in which the bottom boll was occluded by the top boll. The red bounding box tracked the bottom boll while the blue was the upper one. The

upper bounding box appeared to cross over the lower boll but later, the bounding box tracked and localized it. Actually, the bolls were directly over each other.

Figure 3.6. Tracking of the cotton bolls. The lines indicate tracklets, which is the path the bolls were detected across different frames. The green line is the line which the boll which was tracked from at least previous 4 frames will be counted



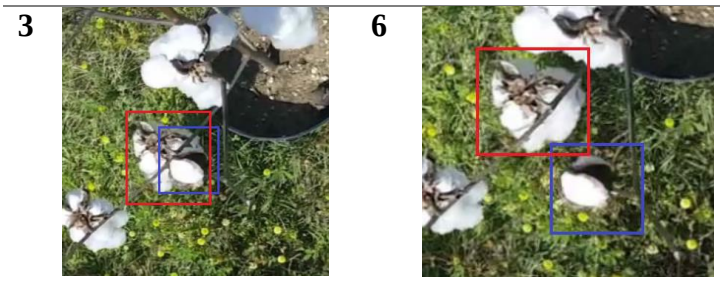


Figure 3.7. Demonstration of occlusion between two bolls that appear to be over each other across consecutive frames from frame 1 to 6. The tracking algorithm tries to maintain tracking of the occluded boll. The algorithm needs to have detected the occluded bolls before tracking. The red boll is at the bottom while blue is at the top.

Table 3.4. Homography transformation of the consecutive images to restore bolls that were lost due to missing good features

Algorithm 4: Homography transformation of the frames to track the lost good features
Input: Previous points (Sokolova et al.),, current points [$\overline{p}_k$], Objects to be tracked [$\overline{B}_k$]
Output: Objects [$\overline{B}_k$] with new values added
1: FOR EACH status in [status]
2: IF status is a good tracked feature
3: P_k <- Assign corresponding element from (Sokolova et al.)
4: $\overline{P}_k$ <- Assign the corresponding element from [$\overline{p}_k$]
5: ENDIF
6: ENDFOR
7: H <- compute homoghraphy (P_k, $\overline{P}_k$, RANSAC)
8: FOR EACH $\overline{B}_k$, status in [$\overline{B}_k$], [status]

9: IF $\overline{B}_k$ has more than one element, and status is not good (zero)

10: x, y,xi,yi,xj,yj,area <- get the head element contents $\overline{B}_k$

11: IF the boll is position is more than 300 pixels away from above

12: D1 <- assign the vector (x0,y0,1)

13: D2 <- assign the vector (xi,yi,1)

14: D3 <- assign the vector (xj,yj,1)

15: S1 <- Multiply the homography matrix H with vector D1

16: S2 <- Multiply the homography matrix H with vector D2

17: S3 <- Multiply the homography matrix H with vector D3

18: x, y,xi,yi,xj,yj,area <- compute the vectors by dividing each of the first and second element with the third element in each of the S1,S2,S3

19: Get the shortest distance from the closest white pixels so that the point doesn't lie on empty bounding box without a boll in it

20: b <- assign the element with the values (x, y,xi,yi,xj,yj,area)

21: IF the length of the object $\overline{B}_k$ is greater than 80

22: Delete the tail of the tracking object $\overline{B}_k$

23: ENDIF

24: IF no boll found in the bounding box

25: Mark the boll as temporarily disappeared boll

26: ENDIF

27: IF temporary disappeared for more than 5 consecutive frames

28: Mark the temporary for deletion

29: ENDIF

30: Assign the new value of the object $\overline{B}_k$ with the b

31: ENDIF

32: ENDIF

33: ENDFOR

34: Return all the identities of the objects [$\overline{B}_k$]

3.3.7 Counting the Bolls and Cleaning Loosely Tracked Bounding Boxes

It was assumed that the end-effector on the robotic arm will be near the bottom of the image, harvesting the bolls. For a boll to be counted, it needs to have tracked at least for the past 4 frames. Otherwise, the tracking object of the boll will be deleted and not counted.

Table 3.5. Counting of the bolls and data cleaning before going back to grab the next frame

Algorithm 5: Counting of the bolls and data cleaning

Input: Objects with elements [$\overline{B}_k$], Current frame, counter

Output: Updated identities of the objects [$\overline{B}_k$]

1: FOR EACH $\overline{B}_k$ in [$\overline{B}_k$]

2: cx,cy,xi,yi,xj,yj,area <- get the head element of the $\overline{B}_k$

3: lower bound <- [170, 170, 150]

4: upper bound <- [255, 255, 255]

5: mask <- compute the range between upper and lower bounds from the current frame

6: boll <- mask out the bounding box { xi,yi,xj,yj }

7: nzCount <- get non zero points that corresponds to boll

8: w <- calculate width from boll

9: h <- calculate height from boll

10: area <- calculate area from boll

11: per_nz <- calculate the percentage of non-zero over the area

12: IF per_nz is less than 1% do

13: Mark the element $\overline{B}_k$ for deletion

14: ELSE

15: xc1, yc1,xa1,ya1,xb1,yb1,area1 <- get the head element of the $\overline{B}_k$

16: xc2, yc2,xa2,ya2,xb2,yb2,area2 <- get the runner's head element of the $\overline{B}_k$

17: IF (yc2 <= divider AND yc1 > divider AND length of the $\overline{B}_k$ is greater than 4)

18: counter <- Count the boll

19: ENDIF

20: ENDIF

21: IF (ya1 > divider):

22: Mark the element $\overline{B}_k$ for deletion

23: ENDIF

24: ENDFOR

25: [$\overline{B}$k] <- Delete the marked $\overline{B}_k$ elements from [$\overline{B}_k$]

26: Return the remaining identities of the objects [$\overline{B}_k$] and number of bolls (counter)

The boll was only counted after its centroid has passed a divider, which was a line across the 650th row-pixel of the image height close to the bottom of the image (Figure 3.6, green line).

650th row-pixel of the image was the position that we assumed the Cartesian robotic arm could be placed to pick the cotton boll. However, it could be put anywhere. The system checks if the cotton boll passed the line and counts. Then the system will check if the tracked boll has finished passing the line and delete it.

3.3.8 Benchmark Experiment Design and Evaluation

The state-of-the-art tracking algorithms implemented in the OpenCV 3.4 were used and evaluated against CottoTrack. The algorithms evaluated were discriminative correlation filter tracker with channel and spatial reliability (CSRT), Kernelized Correlation Filters (KCF), GOTURN, TLD (Tracking, Learning, and Detection), MedianFLow, Multiple Instance Learning (MIL), Minimum Output Sum of Squared Error (MOSSE) and Boosting (Babenko et al., 2009; Bolme et al., 2010; Grabner et al., 2006; Held et al., 2016; Henriques et al., 2012; Kalal et al., 2010, 2011; Lukezic et al., 2017). Each of the algorithms can track one boll. OpenCV provides a Multitracker algorithm that accepts multiple object tracking by providing a tracker for each boll (Table 3.6).

Table 3.6. Model evaluation of the tracking algorithms using Multitracker method implemented in OpenCV

Algorithm 6: Evaluation of models

Input: current video frame, prediction results of the YOLOv2 model [Oj], Multiple trackers object [Ci], the current value of the counter, counter

Output: Current, correct trackers of the multiple trackers [Ci], next value of the counter

1: lower <- [170, 170, 150]

2: upper <- [255, 255, 255]

3: mask <- compute the range between upper and lower bounds from the current frame

```
4:  IF frame is multiple of five
5:      for each Oj in [Oj]:
6:          boll [ xi,yi,xj,yj] <- get the bounding box mask[Oj]
7:          nzCount <- get non zero points that corresponds to (Fischler & Bolles)
8:          w <- calculate width from Oj
9:          h <- calculate height from Oj
10:         area <- calculate area from Oj
11:         per_nz <- calculate the percentage of non-zero nZcount over the area
12:         IF per_nz is more than 25% do
13:             Assign a new tracker to the {Ci}
14:         END IF
15:     END FOR
16: ELSE
17:     [Ti] <- assign the trackers [Ci] to the corresponding tracklet
18:     FOR Ti in [Ti]
19:         IF current value Ti position is greater than the divider
20:             counter <- Count the boll
21:         END IF
22:     END FOR
23: END IF
24: Return all the identities of the Multiple tracker's object [Ci] and number of bolls (counter)
```

3.3.9 Experimental Setup

After training, the 12 defoliated cotton plants (as seen in Figure 3.1) were replaced with new plants that had a total of 131 bolls. For 91-cm (36-inch) row spacing, the cotton field would have 5.2, 10.4, and 15.6 bolls per linear feet for 0.5, 1.0, and 1.5 cotton bales per acre, respectively (Prostko et al., 2018). This would be more than the number of bolls harvested in a continuous harvest system (every 2-days) and was deemed representative in number.

Three trial videos were taken using a moving Samsung camera. In each of the test videos taken, the 12 plants were randomized and rearranged so that we could evaluate the counting algorithms under different orientations. Each trial video was extracted to get individual images. Hence for each video, more than 600 images were extracted. For the six algorithms and ground truth, more than 12,600 images were required to be used in this study. Since we were interested in investigating the tracking algorithms, it was decided to sample 24 images out of each trial video in the interval for detailed analysis from the 104^{th} to the 584^{th} frame at 20-frame intervals. So, a total of 504 out of 12600 images were selected for the investigation to find the tracklets. Set intervals were used instead of random sampling to remove the potential for biased data. Intervals provided data from the start to the end of the row to be equally represented in a dataset. Tracklets were visually checked to determine if they were accurately tracking the boll; otherwise, it was labeled as false positive or if missed, then false negative. The images obtained from the (1) video had 1280 by 720-pixel resolution. Eight parameters from each method were calculated to test model performance. The speed of processing frames (frame per second (fps)) with the embedded system was measured. The reported number of bolls counted was also collected. Also, the total number of frames processed by each model was evaluated. Using manual methods, data to measure true positives (TP), false positives (FP), and false negatives (FN) were collected

(Sokolova et al., 2006) . For each video frame in the sample, the total number of TP, FP, and FN were added and evaluated. After that, the sensitivity or recall, accuracy, F1 score, and precision of the algorithms (Equation. 1) were determined as follows (Sokolova et al., 2006) :

$$Sensitivity \text{ or } recall = \frac{TP}{TP + FN}$$

$$Accuracy = \frac{TP}{TP + FP + FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP}$$

$$F1\ score = \frac{2 * recall * precision}{recall + precision}$$

where

TP = True Positives,

FP = False Positives, and

FN = False Negatives.

3.4 Results and Discussion

3.4.1 Detection of the Cotton Bolls using YOLOv2

First, we investigated the YOLOv2 accuracy by running the trained model to find the prediction for each trial. Table 3.7 shows the performance of the YOLOv2 for each trial and the average frame per second. The average accuracy of YOLOv2 was 75.3±0.5%, and the processing speed was 5.7±0.1 fps. YOLOv2 produced an average precision of 99.6±0.3%, which indicated the model was well trained to identify bolls; however, it was missing many cotton bolls with sensitivity at 75.6±0.7%.

Table 3.7. YOLOv2 predictions for each trial

Trial	TP	FP	FN	Sensitivity (%)	Accuracy (%)	Precision (%)	F1 Score	Speed(fps)
First	286	0	96	74.9	74.9	100.0	0.86	5.6
Second	410	3	126	76.5	76.1	99.3	0.86	5.8
Third	385	2	126	75.3	75.0	99.5	0.86	5.8
Mean				75.6	75.3	99.6	0.9	5.7
Standard Deviation				0.7	0.5	0.3	0.0	0.1

3.4.2 Tracking, Localization, and Counting of the Cotton Bolls

Tracking and localization of each of the detected bolls were expected to improve the detection only model. The target was to increase the accuracy from 75.6% and processing speed from 5.7 fps. This was achieved by only detecting at specified intervals and track for subsequent frames. In Tables Table 3.8, Table 3.9, and Table 3.10, several parameters (sensitivity, accuracy, precision, and F1 score) were determined from all the tracking methods. The tables represent the results of each trial. The first trial had 605 frames, the second trial 727 frames, and the third trial 672. Boosting, Goturn, and TLD were not able to process the images due to a memory full issue. All of them (Boosting, Goturn, and TLD) were very slow and did not pass the 200^{th} frame before failing. The tables report the total number of TP, FP, and FN of the 24 images analyzed for each algorithm. The parameters (sensitivity, accuracy, precision, and F1 score) were calculated corresponding to the value of TP, FP, and FN.

Table 3.8. First trial experiment results

Algorithm	TP	FP	FN	Sensitivity (%)	Accuracy (%)	Precision (%)	F1 Score	Speed(fps)	Count (%)
CSRT	352	0	30	92.1	92.1	**100.0**	0.96	0.7	**100.0**
KCF	333	16	49	87.2	83.7	95.4	0.91	2.2	53.4
MedianFlow	346	10	36	90.6	88.3	97.2	0.94	3.6	98.5
MIL	347	8	35	90.8	89.0	97.7	0.94	0.2	84.7
MOSSE	334	16	48	87.4	83.9	95.4	0.91	**8.1**	57.3
CottoTrack	359	0	23	**94.0**	**94.0**	**100.0**	**0.97**	7.5	95.4

Table 3.8 shows the first trial video performance that was taken while moving the camera faster than the second and third trials. The first, second, and third trials took 21, 25, and 23 seconds respectively to cover 304.8 cm. The slowest camera movement was the second trial. This made samples in the second trial to have more bolls detected between the 104^{th} to 584^{th} frames compared to the first and third trials. In the first trial (Table 3.8), with a relatively higher speed, CottoTrack performed very well compared to the other methods with the second highest speed. In Table 3.8, CottoTrack was slightly outperformed by CSRT when comparing cotton boll counts (100 to 95.4 bolls). However, the CottoTrack's boll counts in the second trial (Table 3.9) with relatively high speed (7.8 fps compared to 0.6 fps) outperformed CSRT in counts but not in sensitivity and accuracy. It was because the algorithm uses a pure Lukas-Kanade algorithm, which requires slow-moving objects to estimate accurately. The sensitivity and accuracy are still very close to CSRT (Table 3.9). MOSSE was very fast (8.1 fps) in the first video because CottoTrack used more transformation to predict position since the change in position tended to be significant in the faster-moving camera.

Table 3.9. Second trial experiment results

Algorithm	TP	FP	FN	Sensitivity (%)	Accuracy (%)	Precision (%)	F1 Score	Speed(fps)	Count (%)
CSRT	518	2	18	**96.6**	**96.3**	99.6	**0.98**	0.6	99.2
KCF	495	24	41	92.4	88.4	95.4	0.94	1.9	74.8
MedianFlow	490	3	46	91.4	90.9	99.4	0.95	3.1	96.2
MIL	510	2	26	95.1	94.8	99.6	0.97	0.2	94.7
MOSSE	485	24	51	90.5	86.6	95.3	0.93	7.7	73.3
CottoTrack	505	0	31	94.2	94.2	**100.0**	0.97	**7.8**	**100.8**

Table 3.10. Third trial experiment results

Algorithm	TP	FP	FN	Sensitivity (%)	Accuracy (%)	Precision (%)	F1 Score	Speed(fps)	Count (%)
CSRT	491	3	20	**96.1**	**95.5**	99.4	0.98	0.6	**104.6**
KCF	465	14	46	91.0	88.6	97.1	0.94	1.8	65.6
MedianFlow	487	14	24	95.3	92.8	97.2	0.96	3.1	**104.6**
MIL	484	11	27	94.7	92.7	97.8	0.96	0.2	87.0
MOSSE	426	12	85	83.4	81.5	97.3	0.90	7.5	71.0
CottoTrack	486	0	25	95.1	95.1	**100.0**	0.97	**7.6**	96.2

While MOSSE had the overall highest average speed(7.8±0.3 fps), CottoTrack had the second-highest frame rate (7.6±0.1 fps), and it was not affected by the number of frames to process compared to MOSSE, which was slower when the number of frames increased (Table 3.11). The low precision in KCF, MIL, and MOSSE (Table 3.11) in tracking algorithms was generally introduced by wrong tracking estimates since YOLOv2 precision was 99.6%. A good

tracker should increase precision like how CottoTrack (100.0±0.0%) and CSRT (99.7±0.3%) achieved. Due to the morphological nature of the cotton bolls, the detection of the bolls may appear as two or more bolls. Hence, counting number that exceeds more than 100% like in CSRT (101.3±2.4%) is desirable (Table 3.11).

Table 3.11. Average and standard deviations (in parentheses) of the three trial experiments

Algorithm	Sensitivity (%)	Accuracy (%)	Precision (%)	F1 Score	Speed(fps)	Count (%)
CSRT	**95.0(2.0)**	**94.7(1.8)**	99.7(0.3)	1.0(0.0)	0.6(0.0)	**101.3(2.4)**
KCF	90.2(2.2)	86.9(2.3)	96.0(0.8)	0.9(0.0)	2.0(0.2)	64.6(8.8)
MedianFlow	92.4(2.1)	90.6(1.8)	97.9(1.0)	1.0(0.0)	3.2(0.2)	99.7(3.5)
MIL	88.2(1.9)	85.7(2.4)	96.8(0.9)	0.9(0.0)	5.1(0.0)	76.3(4.2)
MOSSE	87.1(2.9)	84.0(2.1)	96.0(0.9)	0.9(0.0)	**7.8(0.3)**	67.2(7.1)
CottoTrack	94.4(0.5)	94.4(0.5)	**100.0(0.0)**	1.0(0.0)	7.6(0.1)	97.5(2.4)

3.5 Summary and Conclusion

In this study, an ensemble vision system method that detected, tracked, localized, and counted cotton bolls in real-time was developed. The CottoTrack combined deep learning, color segmentation, and image transformation to locate and track the cotton bolls. The CottoTrack method monitored and counted cotton bolls fast and accurate at 94.4±0.5% accuracy and processing speed of 7.6±0.1 fps. It is an improved performance from YOLOv2 detections, which was 75.3±0.5% accuracy, and the processing speed was 5.7±0.1 fps. The CSRT and MedianFlow performed comparatively higher in boll counting with 101.3±2.4% and 99.7±3.5% accuracy respectively, with much slower speed (0.6±0.0 fps and 3.2±0.2 fps respectively) compared to the

CottoTrack method. MOSSE was as fast as the CottoTrack method but with an average counting of only 67.2±7.1% accuracy. In summary, while MOSSE and CSRT each excelled at separate measures and were poor in others, CottoTrack was consistently the best or within 3% of the best method for all the evaluation metrics and had or shared the lowest standard deviation for five of the six metrics.

The model used two unified techniques (deep learning methods and color segmentation) to detect cotton bolls and used two other techniques (Lucas-Kanade algorithms and homography transformations) to track the bolls. Generally, the speed of image processing in the NVIDIA Jetson TX2 embedded system was slow due to its processor and RAM issue (i.e., if more than 40% of RAM was used, the system would crash). Currently, an improved NVIDIA embedded system (Jetson Xavier) is available, and we are going to test the model using it in future studies. Future work will also include improving the algorithm to track in undefoliated cotton plants with open cotton bolls, cracked bolls, and immature bolls. Also, we expect to add another camera closer to the bottom of the canopy to increase boll tracking with cotton leaves on the plants.

CHAPTER 4

EVALUATION OF A STEREO VISION SYSTEM EFFECTIVENESS IN ROW DETECTION AND BOLL LOCATION ESTIMATION ON A COTTON HARVESTING ROVER IN DIRECT SUNLIGHT[3]

[3] Fue, K., Barnes, E., Porter, W., Li, C., and Rains, G., Submitted to Agronomy, July 1, 2020.

4.1 Abstract

Cotton harvesting is performed by expensive combine harvesters that make it difficult for small to medium-size cotton farmers to grow cotton economically. Advances in robotics provide an opportunity to harvest cotton using small and robust autonomous rovers that can be deployed in the field as a "swarm" of harvesters, with each harvester responsible for a small hectarage. However, rovers need high-performance navigation to obtain the necessary precision for harvesting. Current precision harvesting systems depend heavily on RTK-GNSS to navigate rows of crops. However, GNSS cannot be the only method used to navigate the farm for robots to work as a coordinated multi-agent unit on the same farm because the robots will also require visual systems to navigate, avoid collisions, and accommodate plant growth and canopy changes. Hence, the optical system remains a complementary method for increasing the efficiency of the GNSS. In this study, visual detection of cotton rows and bolls was developed, demonstrated, and evaluated. A pixel-based algorithm was used to calculate and determine the upper and lower part of the canopy of the cotton rows by assuming the normal distribution of the high and low depth pixels. The left and right rows were detected by using perspective transformation and pixel-based sliding window algorithms. Then, the system determined the Bayesian score of the detection and calculated the center of the rows for the smooth navigation of the rover. This visual system achieved an accuracy of 92.3% and an F1 score of 0.951 in the detection of cotton rows.

Furthermore, the same stereo vision system was used to detect the location of the cotton bolls. By comparing the cotton boll distance above the ground with manual measurements, the system achieved an average R2 value of 99% with RMSE of 9 mm when stationary and 95% with RMSE of 34 mm when moving at approximately 0.64 km/h. The rover might have needed to stop several times to improve its detection accuracy or move more slowly. Therefore, the

accuracy obtained in row detection and boll location estimation is favorable for use in a cotton harvesting robotic system. Future research will involve testing of the models in a large farm with undefoliated plants.

4.2 Introduction

Cotton harvesting is heavily dependent on large machinery with human operators. These massive machines are costly to maintain and expensive to own. The emergence of modern technologies in robotics provides an opportunity to explore alternative harvesting methods (Fue et al., 2020a; Hayes, 2017; Kise et al., 2005; Romeo et al., 2012; Winterhalter et al., 2018). The introduction of small, intelligent, multi-agent machines in farming will be an asset to farmers. Swarms can be scalable to the size of the farm, and each machine can be low-cost, multi-purpose, and re-programmable for the task at hand. Smaller machines can also reduce the risk of severe injuries and fatalities sometimes experienced with large field equipment (Fue et al., 2020a; Rains et al., 2015). With modular attachments and selectable programming, these small intelligent machines can be used for multiple tasks, such as precision weeding, chemical application, planting, harvesting, and scouting (Rains et al., 2014).

Recently, several harvesting robots have been developed and reported as research tools or for production agriculture, such as harvesting robots for cucumbers (Van Henten et al., 2003), grapes (N Kondo, 1991; Luo et al., 2016), apples (J. Li et al., 2016), tomatoes (Zhao et al., 2016), strawberries (Hayashi et al., 2014), and sweet peppers (C. Wouter Bac et al., 2017). Most of these robots are slow to pick fruit because of the technology and technique used to identify, locate, and pick the product. The faster robotic machines use a prescription map and multiple robotic arms to harvest many fruits at once (Zion et al., 2014). It is in part due to the difficulty that arises from trying to control the complex farming environment that has variable lighting,

dusty conditions, and machine vibrations, all of which produce "noise" to the imaging system (C. Wouter Bac et al., 2017). These impending conditions pose challenges even to current machine vision technologies.

Plant rows are discernable when cotton plants are seedlings, and the canopy overlaps around 8-10 weeks after planting. Kise et al. (2005) conducted a study to detect rows in young crops using a stereo system and provided an excellent baseline for visual detection of the canopy, an advancement in machine vision research for row crop detection compared to color-based detection algorithms proposed in other studies (García-Santillán et al., 2018; Rovira-Más et al., 2005; Zhai et al., 2016). Winterhalter et al. (2018) proposed the use of LiDAR sensors and RGB cameras to detect small row crops.

The cotton harvesting rover is expected to be deployed in non-defoliated plants as soon as the cotton bolls begin to open. Careful navigation in a fully-grown canopy is required so that the bolls are not knocked to the ground and also are easily located and tracked for picking by the robotic harvester arm. Most cotton in the U.S are planted with row spacing ranges from 30- to 40-inch (76- to 101.6-cm) (UGA, 2019). Therefore, the harvesting machine must make sure that the tires are close to the center of the row spacing. For human-driven tractors, RTK-GNSS is very accurate and can be used to continue following the same course with centimeter accuracy (Higuti et al., 2019; Kise et al., 2005). However, for self-navigation, the visual perception will also be needed to avoid field obstructions. As such, it is important to use visual perception to give the rover an alternative to RTK-GNSS in case it fails and to provide a complementary view of the environment.

RTK-GNSS navigation is challenged by the signal loss during operation resulting from attenuation around buildings, tree cover, and other obstructions (Higuti et al., 2019). This

requires farmers to use an expensive RTK-GNSS system. Additionally, deployment of a swarm of robots that coordinate their work may require preprogramming to ensure that obstacles and machines avoid collisions. Therefore, the use of a camera and simple RTK-GNSS must be sufficient to achieve safe, real-time navigation for farm vehicles traveling over the plants by detecting the canopy and a clear path. There are other sensors like LiDAR that can be used in this operation, but as we expect the machine to work in daylight and over plants, RGB cameras may be sufficient. However, LiDAR, which is an expensive tool compared to RGB cameras, has shown success when small robots navigate at night between large plants (Higuti et al., 2019). However, Bulanon et al. (2004) evaluated the performance of the algorithm using an RGB camera with artificial lighting conditions and found that their algorithm performed very well in the detection of apples when artificial lighting conditions are used.

Some machine vision techniques using LiDAR have been developed to determine the height of cotton plants, but not "on the go," which is a real-time harvesting requirement (Jiang et al., 2016). Machine vision systems for cotton harvesting are not yet available, but some preliminary research has been conducted by Y. Wang et al. (2008) and Mulan et al. (2008). Y. Wang et al. (2008) were able to develop an imaging system for cotton recognition using color segmentation methods and detected cotton bolls at an accuracy of 85%. Furthermore, some work for the visual navigation of a cotton harvesting rover has been done using the Otsu method and noise filtering vision techniques (Xu et al., 2015). Research in India attempted to design an automatic cotton harvesting rover that used image processing techniques to acquire features and perform modeling and matching, but a commercial product was not developed (Rao, 2013). To date, no research has been reported to determine the absolute location of the cotton bolls or cotton rows for robotic purposes.

The location of a cotton boll is the vertical and horizontal distance of the boll from the center of the camera carrying platform. This location is vital for the control of the position of a harvesting manipulator designed to pick individual cotton bolls. However, to achieve this harvesting action, the machines require high performing computing and imaging resources. Present computing technologies can provide a quick solution for cotton boll localization and mapping used to position the robot's end-effector for harvesting. An x-y Cartesian robotic arm can move in two axes: up/down and left/right (Lumelsky, 1986; Zefran, 1996). However, an imaging system is required to predetermine the cotton boll position and send that information to the machine; then, the robot manipulator can plan and move to pick the cotton boll (Lumelsky, 1986; Zefran, 1996).

The main objective of this study was to develop and evaluate a model to detect the rows and cotton bolls in a cotton field and test the performance of the model. A model using a stereo camera to guide a cotton harvesting robot in rows and detect cotton bolls is proposed. The same camera was used to locate bolls and detect cotton rows. The specific objectives of this study were to:

- Develop and evaluate a model to measure the location of the cotton bolls using the stereo camera in direct sunlight
- Develop and evaluate a model to detect cotton rows using a stereo camera in direct sunlight

4.3 Materials and Methods

4.3.1 Materials

The red custom-built articulated rover (West Texas Lee Corp.,) with modifications to meet the field conditions, navigation, and obstacle avoidance requirements of an unstructured

(such as open field, end of the row) and a structured field was used to detect cotton bolls and rows (Fue et al., 2020a; Rains et al., 2015). The rover was 340 cm long and with front and back parts being 145 cm and 195 cm long, respectively. The rover's height and width could be adjusted to a maximum of 122cm and 234 cm, respectively. The rover tires were 91 cm from the center of the vehicle. The rover was 212 cm wide, with a tire width of 30 cm. The four tires had a radius of 30.48cm and a circumference of 191.51cm. The rover had a ground clearance of 91 cm. The rover was mounted with the stereo camera (ZED, Stereo labs Inc, San Francisco, CA, USA) and the rugged development kit, NVIDIA Jetson TX2 (NVIDIA Jetson TX2 development kit, Nvidia Corp., Santa Clara, CA, USA). The NVIDIA Jetson had the following features; NVIDIA Pascal 256 CUDA cores, Quad ARM and HMP Dual Denver CPU, 8GB 128-bit LPDDR4 RAM, and 32GB eMMC SATA drive. A ZED camera was mounted, pointing downward at 81.9° below the horizontal and took images and depth maps at the rate of 5 frames per second at the resolution of 1080p (Figure 4.1). The SDK (standard development kit) was installed with the Ubuntu operating system, NVIDIA CUDA for GPU acceleration, OpenCV (open-source computer vision software), and ROS (Robot Operating System) software. Camera drivers were connected using a ZED camera wrapper that was connected to ROS and collected images from the ZED camera SDK. ZED camera was initiated from the start using ZED SDK, which calculated and rectified the images and disparity maps using stereo camera techniques.

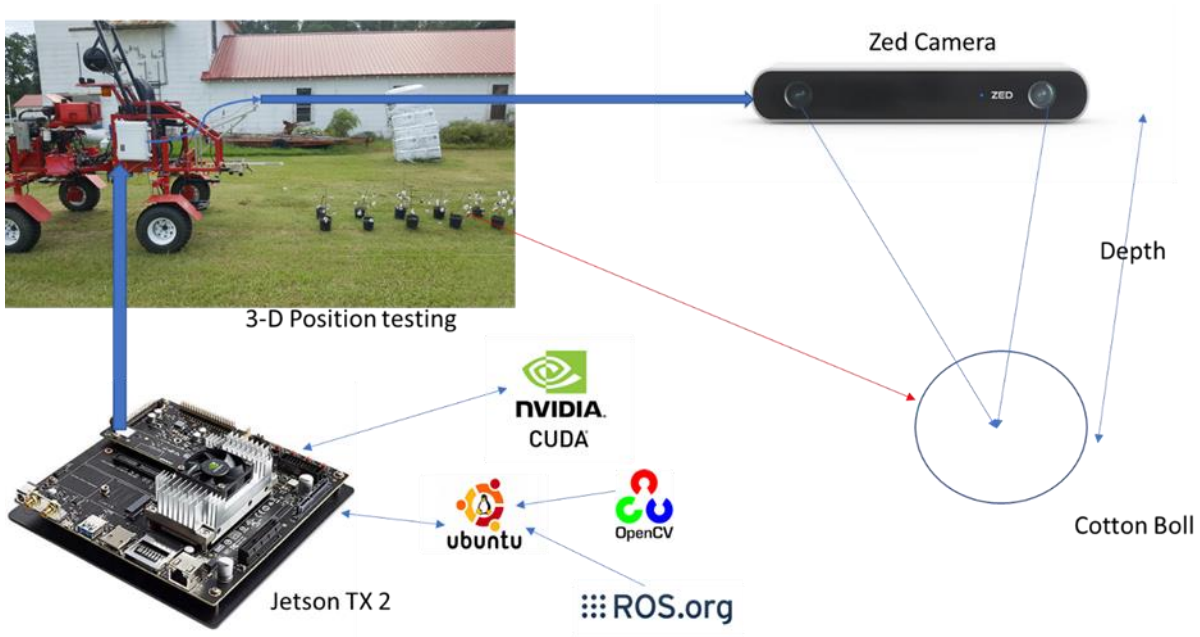


Figure 4.1. Machine vision components of the research

4.3.2 Cotton Row Detection

ZED camera software, the camera was calibrated to achieve the best estimation of the image- and real-world coordinates. Calibration of the camera was important since the model would accurately estimate the center of the rover and real-world position of the wheels along the crop rows. The camera parameters cx, cy, fx, fy, k1, and k2 were found. The symbols fx and fy were the focal lengths, and cx and cy were the optical center coordinates, both in pixels. k1 and k2 were distortion parameters used to rectify the images. The ZED SDK performed rectification in the background, and the rectified images were supplied when requested using the ZED application programming interface (API).

c_x = 674.221

c_y = 374.301

f_x = 697.929

f_y = 697.929

k_1 = -0.173398

k_2 = 0.0287331

$$\begin{bmatrix} I_x \\ I_y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} W_x \\ W_y \\ W_z \\ 1 \end{bmatrix} \quad (4.1)$$

I_x and I_y are image coordinates while W_x, W_y, W_z are real world coordinates

The image coordinates can be transformed accurately into real-world coordinates by using the calibrated parameters and equation (4.1) above. The images obtained from the left lens of the camera were rectified by balancing and removing distortion. Then using the right and left lens image as Figure 4.2, the disparity was calculated by the law of registration of the distance between the two lenses and the location of the point targeted (Lucas & Kanade, 1981).

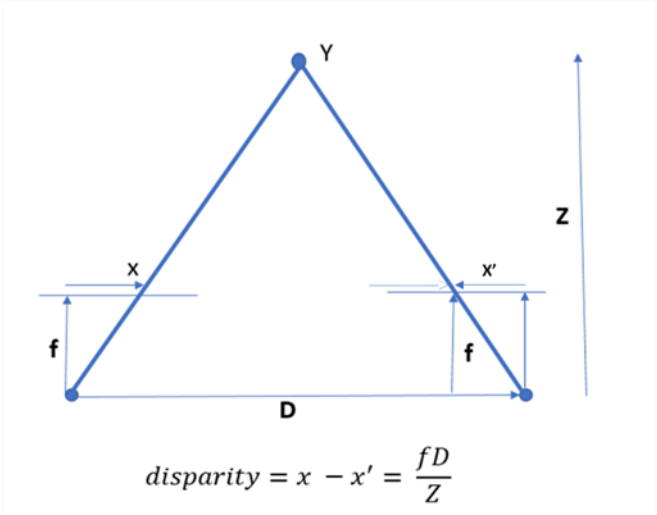


Figure 4.2. Disparity calculation. Where x and x' are the distance between points in the image plane corresponding to the 3D scene point and their camera center. D is the distance between two lenses of the stereo camera, while f is the focal length of both lenses. Y is the location of the object, while Z is the distance of the object to the camera.

In order to balance the view and determine the row width, it was best to transform the depth map. The transformation was performed by the birds' eye view model in which it was assumed that the neighboring pixels presented the rows as large while the back ones were small. Hence, the algorithm used the perspective transform to choose a region of interest and transformed it. The transformation was successfully performed on undistorted images. The source image points and destination points were determined as in Table 4.1. The transformation vertices were determined experimentally by testing several images and determining camera coverage of the rows.

Table 4.1. The perspective transformation vertices for depth maps. Part of the source image vertices are chosen and then transformed into another image (destination) that can easily show the rows in straight patterns, which can easily let the model detect the shape of the rows.

Source Image points (Vertices)	**Destination Image Points (Vertices)**
0.65*960, 0.65*540	960*0.75,0
960,540	960*0.75, 540
0,540	960*0.25, 540
0.40*960, 0.40*540	960*0.25, 0

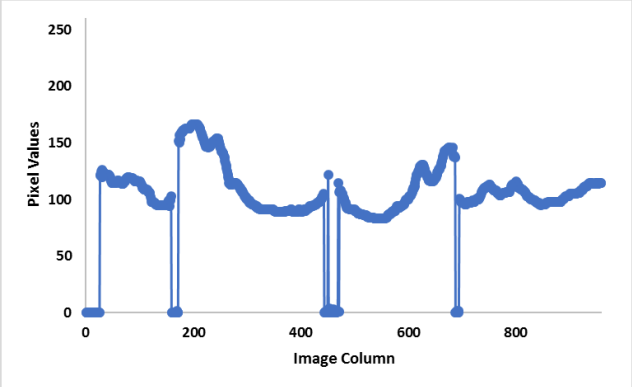
Because the camera is looking downward, the higher the canopy, the lower the value of the depth. In Figure 4.3(1) It means the white pixels (which represent the upper part of the canopy) were pixels that are high 8-bit values compared to gray pixels, which represented the lower part of the canopy and while the soil is represented by black pixels which have very low 8-bit values. So, in each row of the depth map, these values were determined. The sliding window method determined the depth for every 10x10 pixels by finding the average 8-bit pixel gray value. It meant the lower-values pixels represented objects further from the camera while white

pixels closer to the camera. In this essence, the sliding window grouped the highest pixels and predicted the path. The path was achieved by connecting all the pixels that were the highest pixel values, which were determined by choosing the 70th percentile of the pixel values. The 70th percentile of the pixels provided most of the highest pixels covered.

Figure 4.3 shows the depth map that was manipulated at the center (particularly at 300th row out of 540 pixels available). The disparity map was 960 pixels wide. Each of the pixels taken was statistically manipulated to get the 70th percentile, which in this case (Figure 4.3), was 116. Then, all the pixels with the value above 116 were set to white (canopy) and given a value of 255, while all others are reassigned a value of black (value 0). The binary image obtained was further manipulated to delineate crop and row spacing more easily (Figure 4.3(3)).

1

2



3

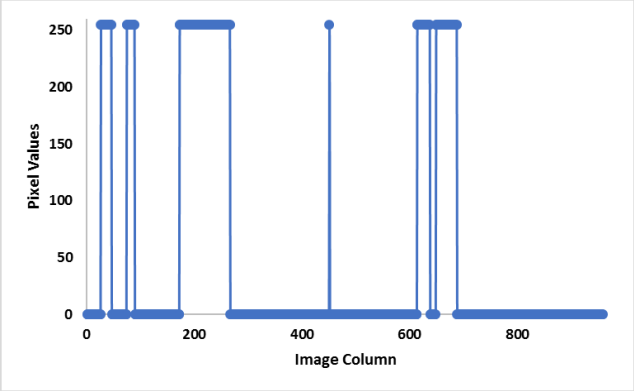


Figure 4.3. Depth map at the row pixel number 300 (each image has 540-row pixels) (1) the pixels are captured in the histogram (2) and then using the 70th percentile only pixels with a value above 116 are used to form a binary image histogram (3).

Error! Reference source not found. presents a depth map that was changed to a binary image and then transformed to locate the rows; then, the sliding window was used to detect the rows. **Error! Reference source not found.**(2) presents a raw depth map. The binary map at the center was obtained by applying the 70^{th} percentile of the row pixels (**Error! Reference source not found.**). Then, the sliding window was used to group the pixels and detected the left and right rows in blue segments. The bottom image in **Error! Reference source not found.** shows the sliding window in green and matching the red line for the left and right row detection. The smooth red line indicates the detection was successful. The algorithm was set such that if the difference between 90^{th} percentile and 10^{th} percentile was less than 60, then the whole pixels of the row were converted to black (or zero) because it meant the difference was not significant to differentiate the top and lower part of the canopy.

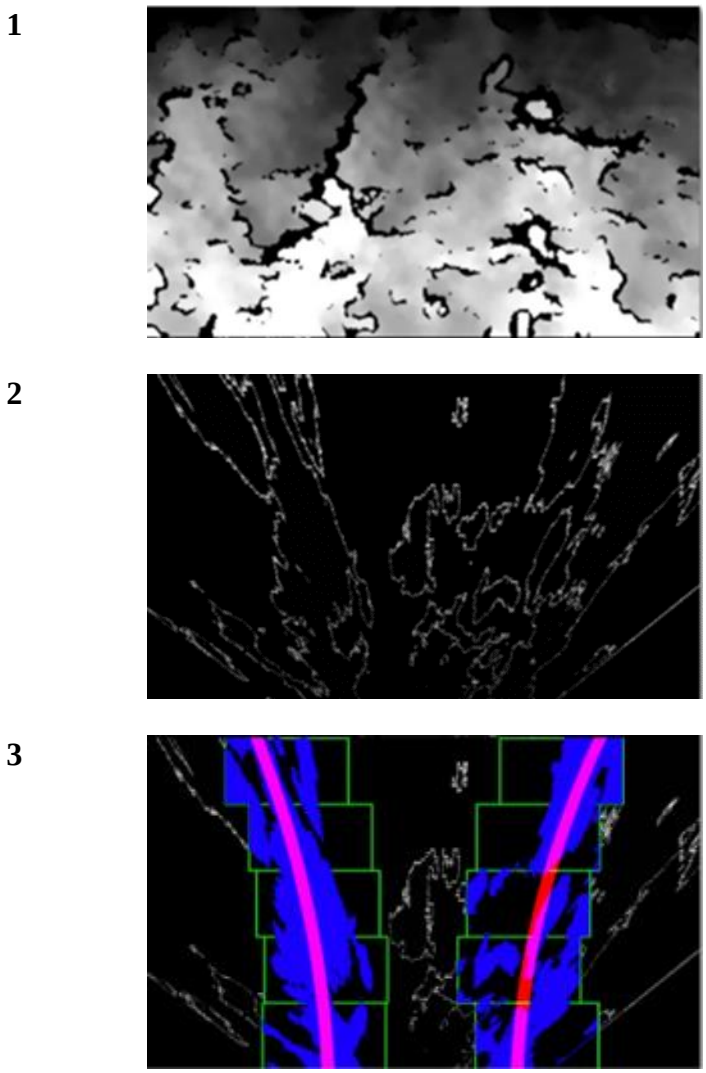


Figure 4.4. Rows are detected by using depth map (1), transformed binary depth map at the center (2), while lower image (3) represents the sliding window detection of the rows.

The detection was determined as the probability as it heavily depended on the appearance of the depth map. If the depth map is not uniform with many variations, it was difficult to get a 70^{th} percentile of the pixels that show uniform changes in the plant canopy. So, a ranking was done (Figure 4.5) to categorize the detection as good (green), moderate (grey), or no detection (red).

1

2

3

Figure 4.5. Ranking the detection of the rows (Upper image means detection was successful as it shows green and yellow stripes, center image detection with gray stripe was moderately successful, while the bottom with red stripe meant no rows were detected)

The binary pixels were detected by placing a 100x50 pixel window along the left and right row. The points were then fitted using a polynomial function to detect the rows. The points were then interpolated to find the polynomial function of the second degree. The assumption is that the rows obey the second-degree polynomial function. Assume, (x_i,y_i) are the distinct points found after matching the pixel sliding window. For distinct points n+1, $x_0,x_1,x_2\ldots x_{n-1}$, x_n and

corresponding points $y_0, y_1, y_2, y_3 \ldots y_{n-1}$, and y_n; there exists a quadratic equation to fit points $[(x_0,y_0), (x_1,y_1), (x_2,y_2) \ldots\ldots (x_{n-1}, y_{n-1}), (x_n, y_n)]$. Assume, function p interpolates pixel values that;

$p(x_i) = y_i$ for all the values of i from 0,1,2…. to n. (4.2)

For second degree polynomial; then

$$p(x_i) = a_2 {x_i}^2 + a_1 x_i + a_0 \quad (4.3)$$

Equating equation (4.2) and (4.3) and arrange them in matrix form leads to;

$$\begin{bmatrix} {x_0}^2 & x_0 & 1 \\ {x_0}^2 & x_0 & 1 \\ \vdots & \vdots & \vdots \\ {x_0}^2 & x_0 & 1 \end{bmatrix} \begin{bmatrix} a_2 \\ a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} \quad (4.4)$$

The left matrix is called to vandermonde matrix (V), while a_2, a_1, and a_0 are the coefficients ($\overline{a}$) of the predicted polynomial equation that we were required to solve. The vandermonde matrix is nonsingular.

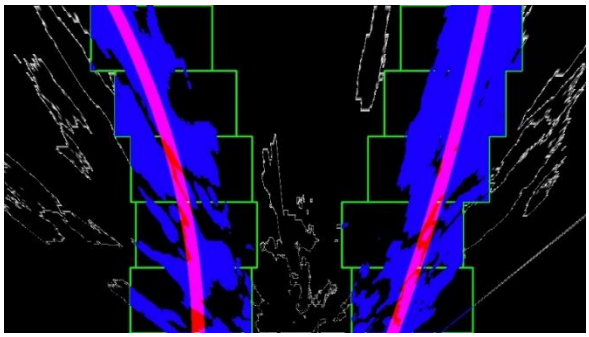
$V\overline{a} = \overline{y}$

$\overline{a} = V / \overline{y}$ (4.5)

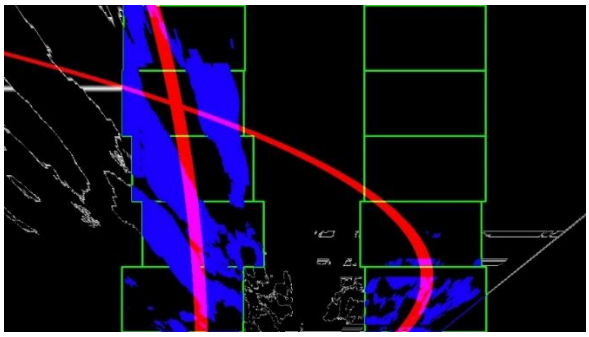
The Manhattan distance between the center of the sliding window and the predicted polynomial fit was calculated to find out how close the sliding window is to the predicted polynomial fit. The points were generated using values of x for each polynomial fit, and then, manhattan distance calculated. The distance was expressed as the percentage from the middle of the sliding window to the polynomial fit. So, the polynomial fit should be inside 100% in the sliding window to be determined as the partial detection. If one of the polynomials was greater than 100% offset, it was concluded the row was not detected at all and marked as a red stripe (Figures Figure 4.5(3) and Figure 4.6(3)). When the polynomial fits were 60% or more away from the sliding window, it was concluded as moderate if there was a row and marked as a gray

stripe (Figures Figure 4.5(2) and Figure 4.6(2)). When the polynomial fits were 60% or less for both left and right rows from the sliding window, it was concluded as the rows were detected successfully and marked with stripes of yellow and green(Figures Figure 4.5(1) and Figure 4.6(1)).

1



2



3

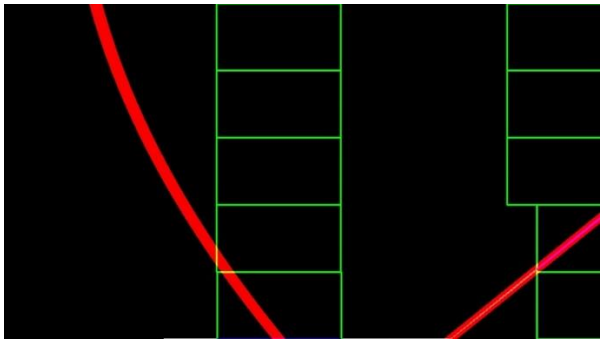


Figure 4.6. Sliding window comparison with polynomial fit using Manhattan distance. The upper image (1) is the successful detection; center image detection (2) was moderate detection confidence while the lower image (3) indicated no row detection

4.3.3 Boll Location Detection

Each image frame was acquired using a ZED camera and analyzed using a 4-step machine vision algorithm (1. depth processing, 2. color segmentation, 3. feature extraction, and 4. frame matching for position determination). These steps were handled by the graphic card optimized rugged development kit (NVIDIA Jetson TX2) to achieve improved matrices calculations using the NVIDIA CUDA cores.

Depth processing was achieved by using the ZED stereo camera, which had two lenses with a separate 1/3" 4MP CMOS image sensor for each lens. This arrangement allowed the camera to have an ability to process 3D images that provided the depth measurement of a cotton boll to the camera. The proximity of the cotton boll was used to determine its distance from the ground as well as the horizontal distance and vertical distance from the center of the camera carrying platform.

Varying light illumination altered image clarity and boll classification frame-to-frame. Also, bolls visible to the sensor in one frame became occluded in a subsequent frame of the boll from a different viewpoint of the camera. The boll detection algorithm must have built-in intelligence to remember the last position of the boll even when it appeared undetected in future image frames. Color segmentation was implemented by using machine vision algorithms deployed in the OpenCV library (Gong & Sakauchi, 1995). A machine vision algorithm was required to mask/subtract all background environment and leave cotton bolls in the frame. Since cotton bolls were white, the algorithms then needed to mask white objects from the environment. The cotton boll detection task involved four steps (Gong & Sakauchi, 1995):

1. Grab an image

2. Using the RGB color threshold, separate each RGB component of the image. For cotton bolls, the white components of the image were masked.
3. Subtract the image background from the original image.
4. Remove all the regions where the contours are less than value M. Value M was determined by estimating the number of pixels defining the smallest boll.

The first step was achieved by applying a threshold to separate the white bolls from the background. For white cotton bolls, the color range/threshold was set to 240-255 in the red, green, and blue channel (8-bit color depth map). It made every boll detectable that gets proper illumination in at least one image frame. It should be noted that this study is more interested in the depth measurement of the stereo vision system. The second step used feature matching and application of a Boolean "AND" operation between the mask image and the original image. The output image was then converted to greyscale.

The last step is feature extraction that is performed by finding contours of consecutive points that have the same intensity and are clustered. Color masking of the grey image was performed, then boundary curves were applied to detect and distinguish all white pixels of the image. For each contour, the center (centroid) was calculated, and the number of pixels that were together was determined. The threshold for the number of pixels together that defined a boll was called M. In this study, two M values, 5 and 15 pixels were chosen and compared.

4.3.4 Frame Feature Extraction, Matching, and Tracking

Frame matching was required to track the position of bolls in respective image frames. In some instances, the algorithm missed the bolls due to illumination problems that impacted brightness, contrast, and sharpness of the image. Hence, the system was developed such that it detected and remembered the boll locations in respective image frames. Since the rover was

moving while the bolls were stationary, multiple frames detected bolls with varying depth measurements. Boll tracking was achieved by calculating the projective transformation (homograph) matrix (3x3) that matched the point corresponding to two consecutive image frames.

Two consecutive image frames were loaded to the CPU, and the ORB feature extraction algorithm applied to get unique features for both frames to calculate the homograph matrix. ORB was a combination of the oriented FAST (Features from Accelerated Segment Test) and rotated BRIEF (Binary Robust Independent Elementary Features) libraries in OpenCV 3.3 (Calonder et al., 2010; Rosten & Drummond, 2006; Rublee et al., 2011). ORB (an open-source machine vision algorithm) was chosen because it was light and the fastest of all the feature extraction algorithms (Rublee et al., 2011). The OpenCV Brute force matcher, FLANN matcher, and findHomography modules were used to get the homograph transformation matrix (Muja & Lowe, 2014). These algorithms were too slow to achieve the required speed as they were taking more than 4 seconds to process two images. C++ bytecode that used 8 CUDA threads to utilize the NVIDIA GPU cores was written. The GPU had CUDA cores that deployed fast graphics computing by implementing parallel processing. The C++ bytecode program utilized only 8 of the 256 GPU cores because only two frames were loaded compared to other applications that deploy a large number of images and hence required more cores. A brute force matcher algorithm that used a random sample consensus (RANSAC) algorithm was then written and applied to the images to get the matching features between the images. The RANSAC algorithm interpreted data containing a lot of gross errors and hence was very useful for where several outliers are prevalent (Fischler & Bolles, 1981). The algorithm used match scores to determine the best matches and left out the outliers (Fischler & Bolles, 1981). The inliers threshold was

determined if at least 5 pixels of the frames matched. Otherwise, it was discarded. By assuming only 20% of the features to match, the system used RANSAC to estimate the data set that contained outliers iteratively (Figure 4.7). RANSAC algorithm followed the following sequence of instruction:

1. A random subset of data was selected, in this case, 20%, and then fit the model.
2. The number of outliers was determined. The data was tested against the fitted model, and the points that fitted the model were considered inliers of the consensus set.
3. The program iterated eight times to achieve the best homograph. The number of iterations was determined by the number of CUDA core blocks and threads. The program established eight threads per block of the CUDA cores.
4. The homograph was then parsed to the main program for tracking and logging boll positions.

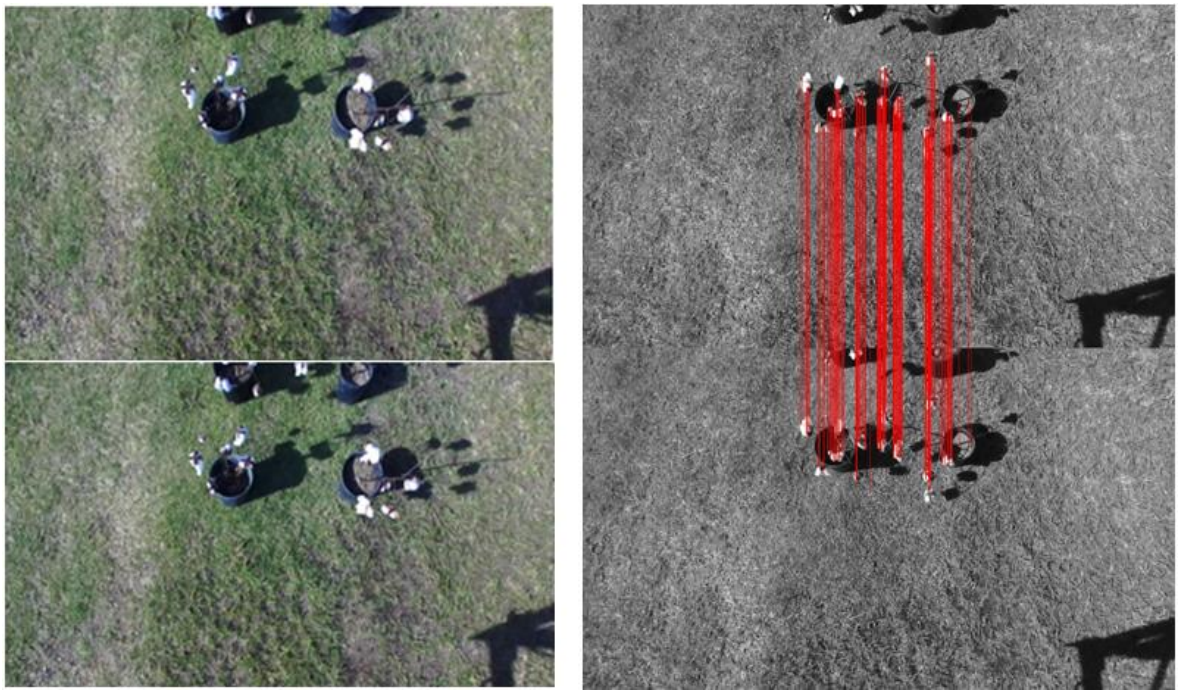


Figure 4.7. Two consecutive images collected from ZED and moving rover (left images) and matching features obtained using ORB and Homograph RANSAC (right image).

The overall imaging software was written in Python, but the RANSAC CUDA code was optimized using C++. Then, the Python subprocess code was used to access the CUDA bytecode. With this CUDA optimized implementation, the system was able to process at least two images per second.

Features between two consecutive frames can be projected and located using the homography transformation matrix, as shown in Figure 4.7. Hence, the new boll position in the next image frame was obtained by multiplying the homography transformation matrix of the initial boll position. For missing bolls in a new image frame, the system used past stored centroids multiplied by the homograph to get the new position of the bolls. After the homograph matrix was obtained, matching boll centroids were determined by using the inverse of the

homograph of the current frame and compared to the previous image frame. The boll position for each boll was logged and stored as an array.

The rectified image was used to get the z-coordinate (height) of the boll after identifying the bolls. The z-coordinate is the vertical distance of the boll from the ground (height from the ground). It is easy to verify this distance manually to evaluate the sensitivity of the camera. However, a hydraulic on/off directional control valve (DCV) was used to turn the rover, and this sudden change in hydraulic pressure caused a "jerk" in the rover when a turn was initiated, and a subsequent side vibration of the camera resulted, introducing errors and bad rectified image frames. The vision system obtained the rectified left camera images and the corresponding depth maps (disparity image) using an interactive API provided by the camera SDK. The depth map corresponded to a perpendicular distance from the left camera lens to the cotton bolls. Hence, a model was developed to get the vertical distance of the boll from the ground. This measurement was the only coordinate that was determined as it is permanent, while other readings were relative measurements and changed as the rover moved over the plants. The camera mounted on the rover was inclined at 81.9° from horizontal and obtained 1280 x 720-pixel frames. The field of view was covered at an angle of 54° (ϕ) vertically and 96° horizontally.

The system calculated the moving average of cotton boll locations as it grabbed images. The average was used to determine the position of the boll relative to the future cotton-picking end effector. Considering Figure 4.8, , the configuration setup of the rover, camera, and cotton plants is illustrated. Corresponding measurements are:

m is the vertical distance from the camera to the cotton bolls,

n is the height distance of the boll from the ground,

θ is the vertical angle of the object (cotton boll) from the bottom of the image to the boll,

ϕ is the vertical field view of the image, and

μ is the vertical angle on the image from the bottom of the image to the boll.

The system was developed to measure the distance of the boll from the ground (n). By considering the middle boll (brown arrow), the depth of the boll and distance from the ground was determined. Depth reported by the camera is equal to the one given by the formula m*atan(μ). By using ZED SDK API, the depth of each pixel with a 16-bit resolution was obtained.

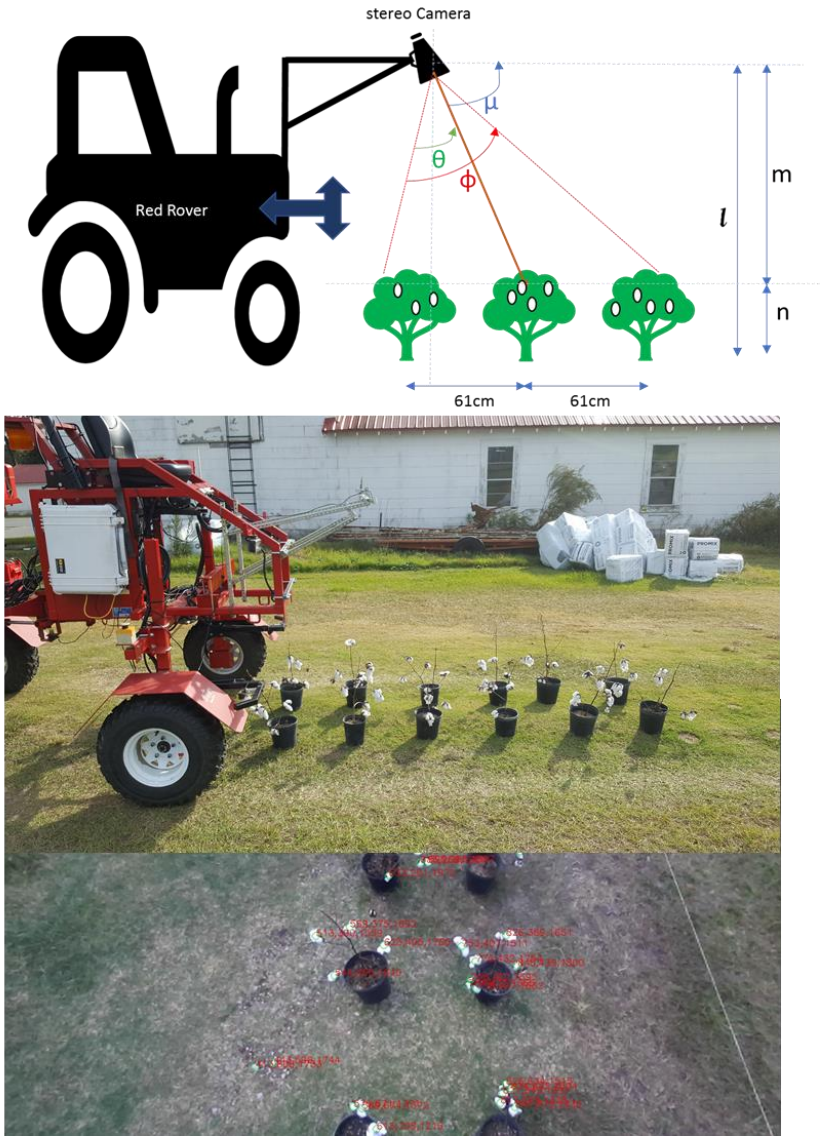


Figure 4.8. Context diagram that shows cotton boll position measurements

$$depth\ of\ an\ object = m * \text{atan}\ (\mu)$$

Since the camera was inclined at angle μ° from horizontal, the equivalent angle (θ) in radians of the object is given by

$$angle(\theta) = \frac{\pi}{180} * abs(\left(\left(\frac{720-y}{720}\right) * \phi\right) - (90 - \mu - \phi/2)) \quad (4.6)$$

Now, distance from the ground is given by

$$n = l - m * \text{atan}\,(\mu) * \cos(\theta) \quad (4.7)$$

Since, depth of an object is provided by ZED SDK then,

$$n = l - depth * \cos(\theta) \quad (4.8)$$

The system generated images that show how the bolls were tracked, and z-coordinate was determined (Equation 4.7). The system published images using ROS, and hence clients could get live video of the frames. This video was slower as the system only calculated an average of two frames per second. Graphs were produced to show results.

Figure 4.9. Left image show processed boll position and tracking of the boll while the right images show a series of 3 image frames acquired from the moving camera.

4.3.5 Data Collection

Field data for row detection were collected at the UGA Lang farm (31.521501, -83.545712) along Carpenter Road, Tifton, GA USA. The images were collected using the ZED camera at an average frame rate of 5 and a rover speed of 4.8 kph near midday on 28th August 2017. The images and depth maps were stored in the internal memory of the development kit. 381 of the collected images (images like Figure 4.10)) that cover four rows passing the first two, and then the next two rows when coming back were used for analysis and validation of the row detection model.

Left lens image Corresponding Depth Map

Figure 4.10. Collected RGB image and corresponding depth map for cotton row detection (0-255, 8-bit greyscale image).

In December 2017, an experiment to evaluate the cotton boll tracking model was conducted at UGA campus grounds (N Entomology Dr, Tifton, GA, 31793) at (31˚ 28'N 83˚ 31'W). The location was open to direct sunlight. Twelve defoliated cotton plants were taken from a nearby farm and put in soil-filled pots. The plants were placed in 2 rows of 6 plants. The plants were 91.4 cm between the center of the stalk (row spacing) and each stalk 61 cm from the next (plant spacing). The distances of all bolls were measured manually, and the rover driven over the bolls collecting RGB and depth information from the ZED camera. A static test was first conducted on 1st December 2017 with the rover set over the bolls, and two consecutive frames were taken.

On 4th December 2017, the second test was done. The plants were randomized, and data collected in three rover speed treatments, 1.04 km/h, 0.80 km/h, and 0.64 km/h. The relative positions of the bolls were determined by measuring boll distance from the ground. The data were collected by changing the M parameter (from 15 to 5) to detect white contours (cotton bolls). A comparison of the camera and manual measurement of boll locations was conducted.

Comparative statistics were used to measure standard error, root mean square error, and mean error.

4.4 Results and Discussion

4.4.1 Row Detection

The 381 images collected were each evaluated to assess the detection of the rows (Table 4.2). Results were based on the ranking of the software, as previously described, to determine if row detection was successful. Images were manually categorized as difficult or easy. The easy detection categorization meant the software detection was correct since the canopies are explicitly separated, and rows can easily be seen, and hence, all the rows aligned with a sliding window. Difficult categorization meant the image had plant canopies heavily overlapping to each other, which makes it difficult to differentiate the two rows, and hence, some of the row pixels were out of the sliding window, but the detection was still successful. The true positive categorization meant the row detection was successful. False Positive categorization meant row detection was found in a place where there were no visible rows. False positives occurred when plants other than cotton appeared between rows, or the depth image obtained was blurred. True negative meant rows were not detected, and the software successfully assigned no rows to that situation. The easy category meant the system was 100% sure there were no rows because there were no differences between upper pixels and lowers pixels confirming the absence of a plant canopy. False Negatives were situations where the software did not detect a row when a row was there. It was most commonly caused by skips in the rows where cotton was not growing.

Table 4.2. Results of the manual inspection of the images.

	Easy	Difficult	Total
True Positive	207	76	283
False Positive	00	01	01
True Negative	54	15	69
False Negative	05	23	28
Total	**266**	**115**	**381**

$$\text{precision} = \text{TP}/(\text{TP} + \text{FP}) = 283/284 = 0.996$$

$$\text{recall} = \text{TP}/(\text{TP} + \text{FN}) = 283/(283 + 28) = 0.909$$

$$\text{F1 Score} = 2 * \frac{\text{Precision} * \text{recall}}{\text{precision} + \text{recall}} = 0.951$$

$$\text{Accuracy} = (\text{TP} + \text{TN})/(\text{TP} + \text{FP} + \text{FN} + \text{TN}) = (283 + 69)/381 = 0.923$$

The model was evaluated from the data collected (Table 4.2). The vision system was found to perform at 92.3% accuracy with an F1-score of 0.951. The algorithm was accurate, but it had many difficulties in predicting the rows that had plants that were shorter or taller than the average cotton plant. Some rows were occluded by plants from both sides of the row that had a big canopy that fully occupied the row. These situations led to the row detection algorithm to predict 28 false negatives cases.

4.4.2 Cotton Boll Detection

A static test was conducted to assess the ability of the camera to classify and locate bolls without the dynamics of a moving camera. The red rover was set stationary but running, and two consecutive frames were collected and analyzed (Figure 4.11). Using Excel, manual

measurement of sixteen boll locations was compared to camera image measurements. The camera accuracy from the first image frame showed a regression relationship to the manual measurements of R^2 equal to 99% and root mean square error (RMSE) of 11 mm. The second frame gave R^2 equal to 98%, and RMSE of 17 mm. The mean error was -6 mm and 9.9 mm for first and second frames, respectively. The standard deviation was 9.8 mm and 14.8 mm for the first and second frames, respectively (Figure 4.11). Results show the camera system was able to classify and locate bolls under direct sunlight with low cotton boll density.

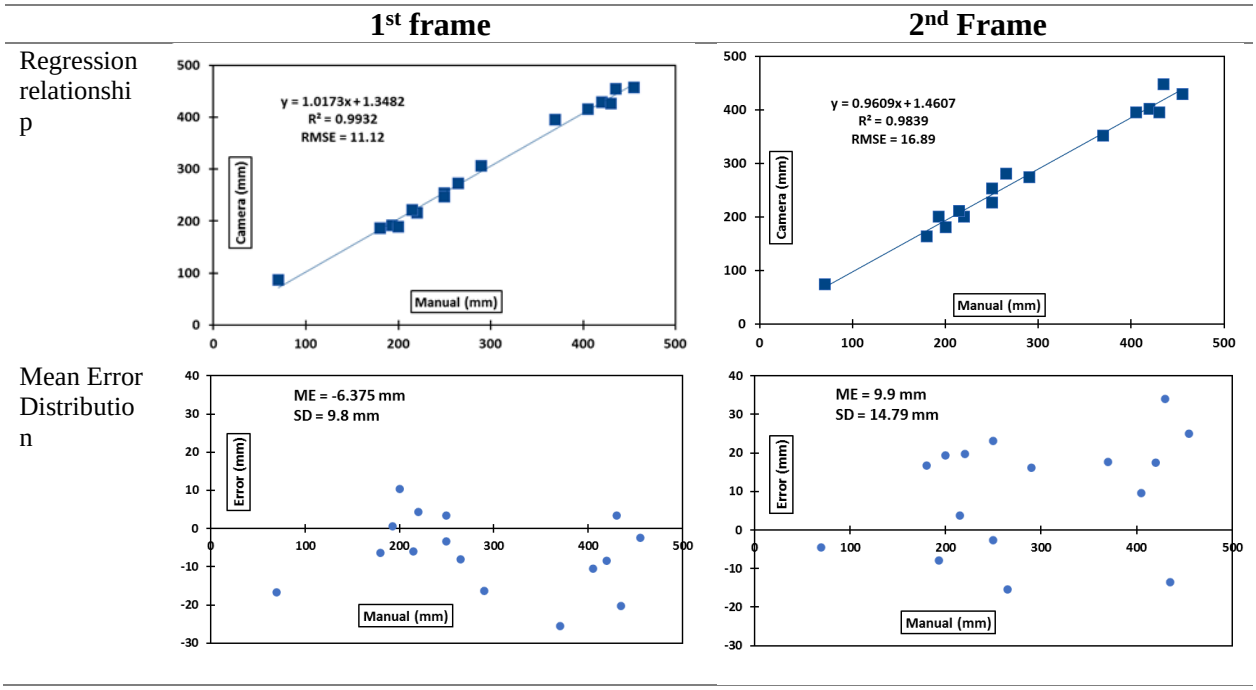
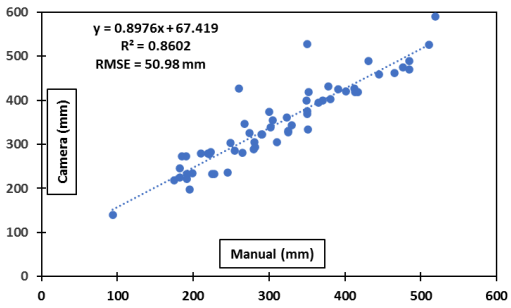
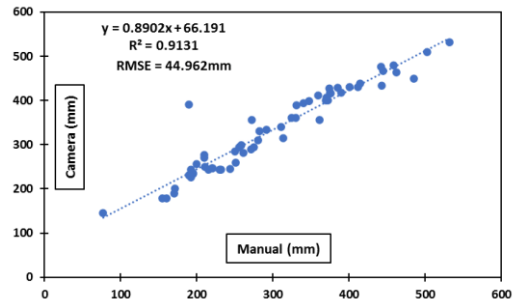


Figure 4.11. Comparison of the image frames when the rover was stationary. The y-axis is the camera measurements, while the x-axis is the manual measurements—the first and second frames were taken consecutively to compare the depth estimation.

The software detected the bolls and recorded multiple depths for the same boll as the vehicle moved over the row during the test. The 15-pixel boll contour (M=15) was only able to detect 92.3% of all 65 bolls available, while when M=5 was introduced, the system was able to detect all the bolls. Figure 4.12 demonstrates the results of color segmentation detection and masking. The white spaces had to form a contour that passed the threshold M-value, 15 or 5

pixels. The color of the boll and obstruction may make one boll detectible in one frame but not the next. The same boll may also be detected in consecutive frames. By being detected more than once, the system was able to obtain more than one depth reading for individual bolls. Multiple values obtained were averaged to get an estimated depth value. Figure 4.13 shows the regression relationship of the experiment for all three different speed tests. Figure 4.14 shows the mean error distribution of the experiment. The boll detection algorithm had the worst R^2 of 0.86 for the highest rover speed (1.04 kph) and M=15 contour, while R^2 of 0.95 for the slowest rover speed (0.64 kph) M=5 contour. In Figure 4.14, results show that the M=15 data had a larger RMSE compared to M=5. These errors were mainly due to the "jerking" of the rover when adjusting the right and left turn and topography of the land to maintain a straight path for the rover.

Figure 4.12. Segmentation results and masking of the images.

	M=15	***M=5***
1.04 kph		

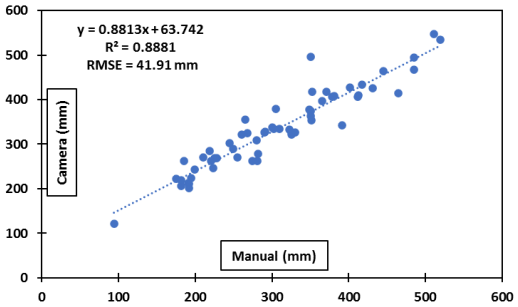
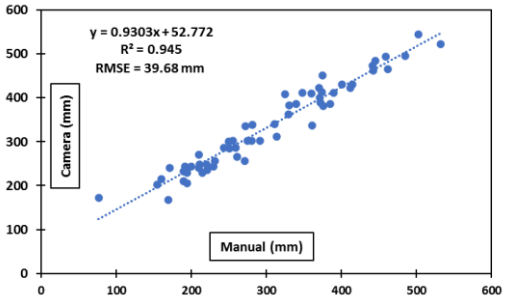
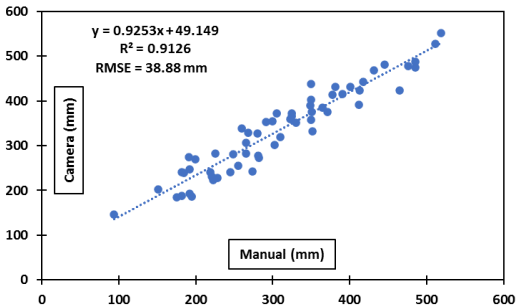
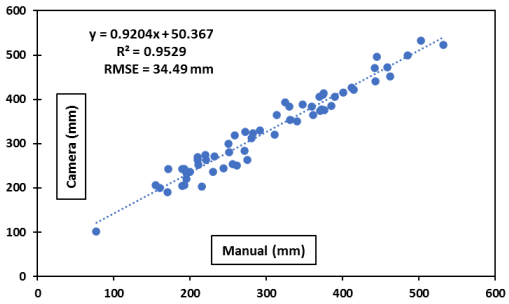
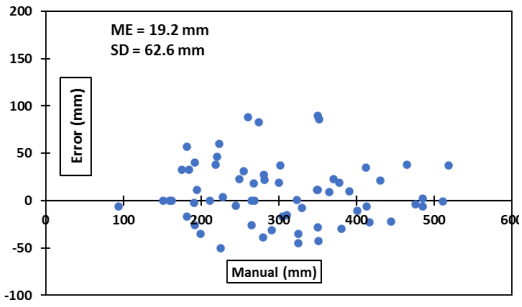
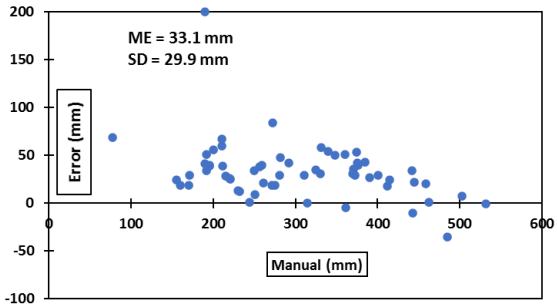
	M=15	*M=5*
0.80 kph		
0.64 kph		

Figure 4.13. Comparison of 15 pixels contour and 5 pixels contour for 1.04 kph (fast speed), 0.80 kph (slow speed), and 0.64 kph (very slow speed) of cotton boll position measurements.

	M=15	*M=5 pixels*
1.04 kph		

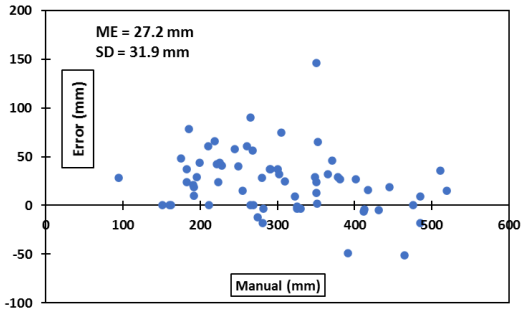
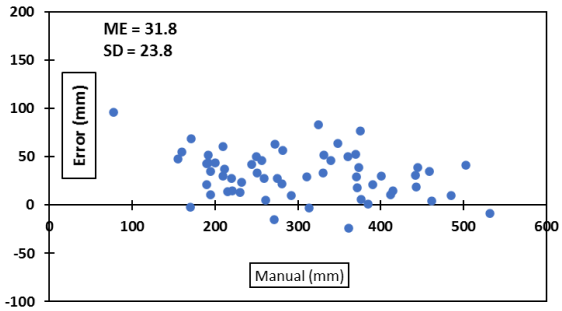
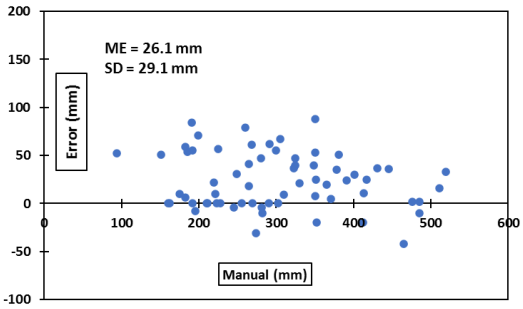
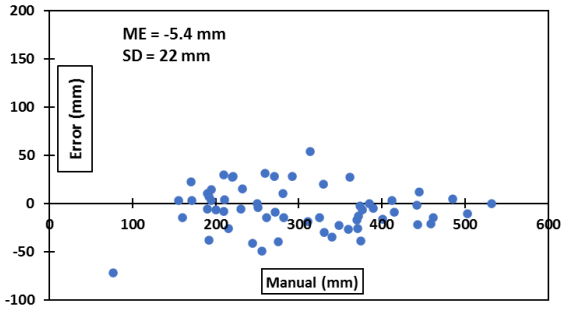
	M=15	*M=5 pixels*
0.80 kph		
0.64 kph		

Figure 4.14. The Mean error distribution of the experiment (15 pixels contour and 5 pixels contour for 1.04 kph (fast speed), 0.80 kph (slow speed), and 0.64 mph (very slow speed) of cotton boll position measurements).

4.5 Conclusion

Imaging systems to determine 3D boll location and row detection were developed and evaluated in this study. The performance of the algorithm developed in this study to detect the rows showed promise as a method to assist with the RTK GNSS navigation of an intelligent rover for harvesting cotton bolls. Adding a visual system to the navigation provided an increased perception of the environment to aid in avoiding obstacles and seeing the actual row path for the vehicle. Furthermore, in case of failures of the RTK-GNSS, the camera system can help the rover

continue moving without pausing to regain localization, which can cost field operational time. The results suggest that the visual system can be deployed for rover navigation assistance and replacement during GNSS interruptions. But the visual system will only work looking downward when rows are not occluded with large plant canopies or in places with no plants. A camera closer to the ground positioned horizontally or slightly upward into the canopy could provide a better perspective of rows closer to the plant crown at the ground.

For boll location, the system was able to acquire images and process two frames per second using the GPU resources of the system. When comparing the boll detection and localizing system to manual measurements, the performance was proportional to the speed of the rover and contour threshold (M) used to detect bolls with better performance. With M=5, the system may detect multiple contours for the same boll compared to M=15. As a result, the boll tracking system decreases time spent errantly tracking multiple contours as different bolls when there is only one boll present. It should help increase harvest speed when using a robotic arm with an end-effector to harvest cotton bolls identified by the boll tracking system. Results showed that the rover would have to stop in some locations to get the best measurement of boll locations in the field for cotton boll picking with a robotic arm. The accuracy achieved by tracking cotton bolls is favorable for proceeding to the development of a cotton harvesting robot. Future research should involve more field testing of the models developed in this study in realistic conditions.

CHAPTER 5

AUTONOMOUS NAVIGATION OF A CENTER-ARTICULATED AND HYDROSTATIC TRANSMISSION ROVER USING A MODIFIED PURE PURSUIT ALGORITHM IN A COTTON FIELD[4]

[4] Fue, K., Barnes, E., Porter, W., Li, C., and Rains, G., Submitted to *Sensors*, July 1, 2020.

5.1 Abstract

This study proposes an algorithm that controls an autonomous, multi-purpose, center-articulated hydrostatic transmission rover to navigate along crop rows. Accurate navigation without damaging plants can promote the development of the spraying, scouting, and harvesting operations used in multiple crops. This multi-purpose rover is being developed to harvest undefoliated cotton to expand the harvest window to up to 50 days. The rover would harvest cotton in teams by performing several passes as the bolls become ready to harvest. We propose that teams of rovers with interchangeable attachments could make cotton production more profitable for farmers and more accessible to owners of smaller plots of land who cannot afford large tractors and harvesting equipment. The rover was localized with a low-cost RTK-GNSS, encoders, and IMUs for heading. ROS-based software was developed to harness the sensor information, localize the rover, and execute path following controls. To test the localization and modified pure-pursuit path-following controls, first, GNSS waypoints were obtained by manually steering the rover over the rows followed by the rover autonomously driving over the rows. The results showed that the robot achieved a mean absolute error (MAE) of 0.04m, 0.06m, and 0.09m for the first, second, and third passes of the experiment, respectively. The robot achieved an MAE of 0.06m. When turning at the end of the row, the mean absolute error (MAE) from the RTK-GNSS-generated path was 0.24m. The turning errors were acceptable for the open field at the end of the row. Errors while driving down the row did damage the plants by moving close to the plants' stems, and these errors likely would not impede operations designed for the multi-purpose rover. The designed robot achieved optimum performance, and it will be acceptable for cotton harvesting operations.

5.2 Introduction

Mechanical harvesting has helped improve crop production significantly since the mid-1900s. Before these machines were developed, crops such as cotton were primarily hand-harvested. The development of the cotton combine helped to reduce labor costs and increase production efficiency but comes with downsides. Harvesting is accomplished using expensive machines that are massive in size and weight, which can lead to soil compaction, and are also costly and time-consuming to repair. Breakdowns during the season may expose cotton to hostile environmental conditions that can diminish the quality of the harvest. Besides, Cotton harvesting takes place only after cotton fields have been defoliated with chemical defoliants. These chemicals can degrade the land and are an added expense to production costs. The defoliation is performed approximately 50 days from the opening of the first bolls. A consequence of waiting many weeks to harvest all the cotton simultaneously, lint quality is affected profoundly by external weather and other environmental elements from the time they open to the time they are picked (UGA, 2019; USDA/NASS, 2018). As a consequence of the current cotton management system, small acreage farmers can not afford to buy these machines or maintain them (Duckett et al., 2018). Furthermore, the fast aging farming community will experience a labor shortage since many of their children are moving to urban areas (Duckett et al., 2018).

Furthermore, some row crops exceptionally need special procedures for harvesting. For example, cotton harvesting takes place only after cotton fields are defoliated using chemical defoliants. These chemicals can degrade the land and are an added expense to production costs. The defoliation is done around 50 days from the opening of the first bolls. Cotton lint quality is profoundly affected due to external weather and other environmental elements that may

contaminate and diminish the quality from the time they open to the time they are picked. (Fue et al., 2018b; Hayes, 2017).

Any solution that could increase the participation of small and family farmers would be well received. One alternative to current large-scale farm and machinery systems is to introduce small, multi-purpose, robotic rovers that can navigate and perform agricultural operations autonomously without the need for human-machine control. Fortunately, there is a booming industry in robotics and machine learning technologies, and robotics have been developed to solve many pertinent issues in agriculture (Duckett et al., 2018; Fue et al., 2020b; Fue et al., 2018b; Hayes, 2017; Rains et al., 2014).

However, for mobile robotic systems to be efficient, they need very effective methods to navigate fields autonomously. Autonomous navigation of robotic systems depends upon four modules: sensors, vehicle mobility, perception, and control algorithms (Figure 5.1). Sensors such as RTK-GNSS, RGB cameras, Stereo camera, Light Detection and Ranging (LiDAR), SOund NAvigation and Ranging (SONAR), Ultrasonic, Radio-frequency identification (RFID), Inertial Measurement Unit (IMU), Laser scanner, RAdio Detection And Ranging (RADAR), encoders, thermal imaging, hyperspectral and infrared have been used extensively to detect fruits and plants in agricultural fields (Figure 5.1). Additionally, many vehicle mobility systems have been developed primarily for agricultural use, such as continuous tracks, the Ackermann four-wheel drive, center-articulated drives, legged-robots, swinging robots, omnidirectional drives, and sliding-on-the-rail robots (Figure 5.1). The mobility is designed to accommodate different soil and topographic conditions, open or greenhouse farming, and maneuverability requirements. Perception is created using sensor output and machine vision algorithms, such as image segmentation, hough transformation, sensor fusion, or machine learning, to obtain environmental

features (Figure 5.1). After perceiving the environment, the mobile robot performs a movement action to the next location. There are multiple control system methods used: fuzzy logic, nonlinear, proportional-integral-derivative (PID), adaptive, model-based, rear-wheel feedback, linear-quadratic regulator, model predictive control (MPC), and machine learning, such as neural networks and reinforcement learning. After assimilating a control system, several techniques must be incorporated to develop autonomous navigation in an unstructured environment, such as localization, mapping, obstacle avoidance, simultaneous localization, and Mapping (SLAM), row-following in row crops and path planning to perform complex farm operation movements (Auat Cheein et al., 2011; Backman et al., 2012; Ball et al., 2016; Boubin et al.; Cheein et al., 2010; Coulter, 1992; Duckett et al., 2018; Farzan et al., 2018; Fue et al., 2020b; Grimstad & From, 2017; Higuti et al., 2019; Kayacan et al., 2018; Liakos et al., 2018; Ouadah et al., 2008; Ramin Shamshiri et al., 2018; Reiser et al., 2019; Tu et al., 2019; H. Wang & Noguchi, 2018; Xiong et al., 2019; Xue et al., 2012).

In this study, two IMUs, a high precision potentiometer, two encoders, a low-cost single-frequency RTK-GNSS, and the sensor fusion algorithm Extended Kalman Filter (EKF), were utilized to perform autonomous localization and navigation of the robot. Proportional control and a modified pure pursuit algorithm were implemented to perform autonomous cotton row following for a multi-purpose rover.

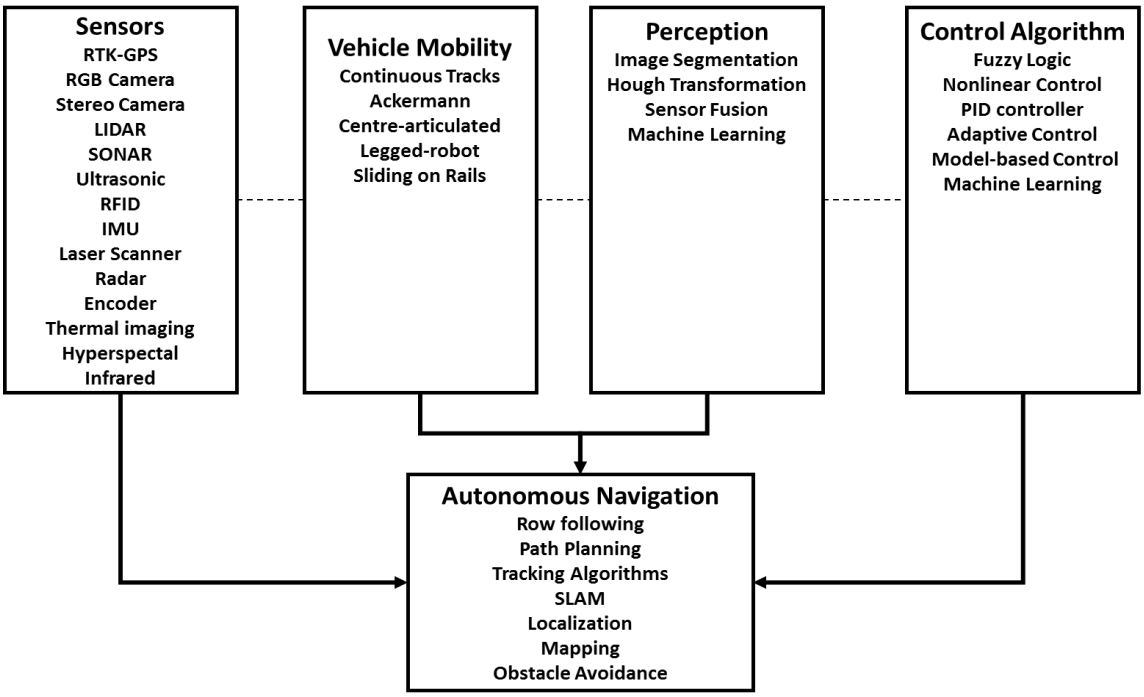


Figure 5.1. Autonomous navigation modules.

The development and performance of the autonomous navigation of a multi-purpose rover are presented in this paper. Autonomous means the rover can navigate itself along cotton rows without the intervention of human subjects and without destroying plants or cotton bolls. High precision is required to achieve acceptable navigation without causing an economic reduction in yield. Therefore, in this study, we present two objectives;

1. Development of the navigation system of the autonomous center-articulated multi-purpose rover
2. Evaluation of the field navigation of the autonomous center-articulated multi-purpose rover

5.3 Materials and Methods

5.3.1 Robot components and System Setup

The rover (Figure 5.2) was a custom-built four-wheel center-articulated robot (West Texas Lee Corp., Lubbock, Texas). The rover was 340 cm long with front and back parts (divided by the center of articulation) being 145 cm and 195 cm long, respectively. The rover's height and width could be adjusted to a maximum of 122cm and 234 cm, respectively. The rover tires were 91 cm from the center of the vehicle. The rover was 212 cm wide, with a tire width of 30 cm. The four tires had a radius of 30.48cm and a circumference of 191.51cm. The rover had a ground clearance of 91 cm. The rover used seven sensors; two IMUs, a high precision potentiometer, two rotary encoders, and RTK-GNSS. Each front tire was connected to a rotary encoder (Koyo incremental (quadrature) TRDA-20R1N1024VD, Automationdirect.com, Atlanta, GA, USA). The two IMUs (Phidget Spatial Precision 3/3/3 High-Resolution model 1044_1B, Calgary, Alberta, Canada) were placed in front of the rover. First, IMU was placed 95 cm above the ground and 31 cm from the front of the vehicle. The second IMU was 132 cm above the ground and 46 cm from the front of the vehicle. The low-cost RTK-GNSS (USD 800) single-frequency receiver (EMLID Reach RS, Hong Kong, China) was placed 246 cm above the ground and 30 cm from the front of the vehicle. An embedded system (NVIDIA Jetson AGX Xavier development kit, Nvidia Corp., Santa Clara, CA, USA) was installed and used to control rover navigation and read sensor data.

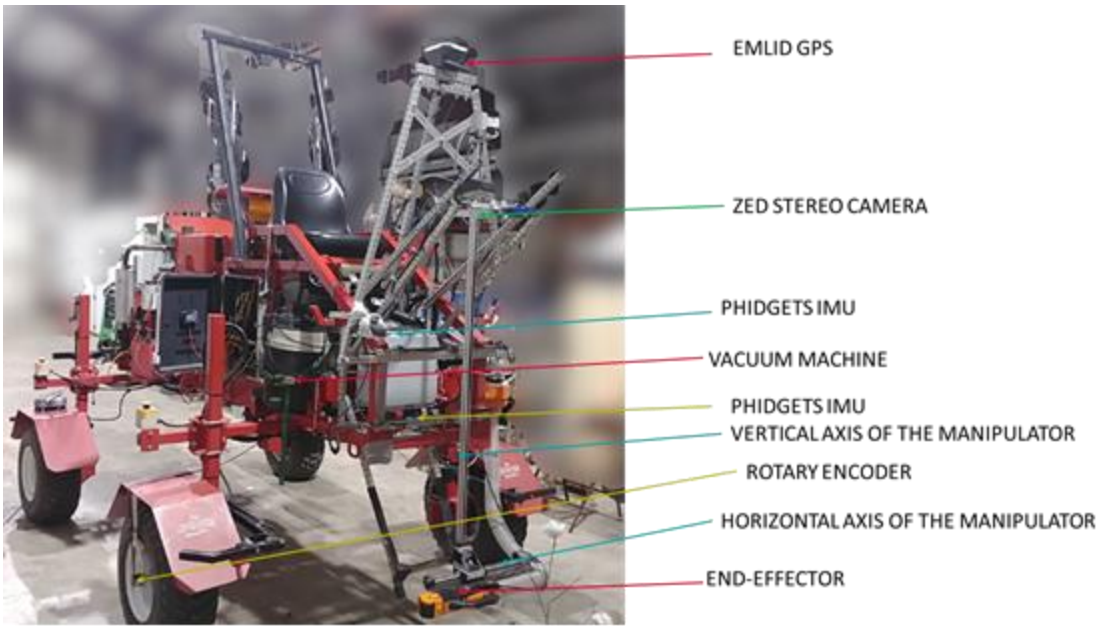


Figure 5.2. The red research rover with manipulator and sensors attached in front of the rover implemented for this study

All sensors except the rotary encoders and potentiometer were connected to the embedded system via a universal serial bus (USB). The encoders were connected to the rover navigation controller (Arduino Mega 2560, Arduino LLC) using four wires; signal A and B, power, and ground so they can register rotation of the tires by detecting the leading edge of rising square waves. A high precision potentiometer (Vishay Spectral Single Turn, Malvern, PA) was used to report the articulation angle of the vehicle by measuring the electric potential caused by the turn of the vehicle, which was correlated to the angle. The potentiometer was connected to the rover navigation controller.

ROS (Robot Operating System), which is robotics middleware used for robot software development, was implemented to connect the embedded system (Jetson Xavier) with the rover navigation controller. The robot software was developed to communicate using ROS topics. ROS topics were named buses that the nodes (embedded system and navigation controller) used to

exchange messages (Koubâa, 2017). The sensors connected to the embedded system published the updates that were utilized by both the embedded system and navigation controller. The topics were set to communicate so that they did not know the other nodes they were communicating with(Koubâa, 2017; Quigley et al., 2009). The rover navigation controller received a signal from the embedded computer to control the rover movement, articulation, and engine throttling. All four wheels of the rover were mounted to hydraulic motors (Parker 2090B 238 cc/rev) that had their rotation controlled using a linear servo to the swashplate lever of a 14.1 cc/rev axial-piston variable rate pump (OilGear, Milwaukee, WI, USA). The swashplate angle was controlled by the rover controller that determined the placement of the linear electric servo (Robotzone HDA4, Servocity, Winfield, KS) that had a maximum movement of approximately 10.16 cm. The left/right articulation was controlled through a 4-port 3-way open-center directional control valve (DCV) connected to 2 hydraulic cylinders and powered by a 0.45 cc/rev fixed displacement pump (Bucher Hydraulics, Italy) in tandem with the variable rate pump. The DCV provided hydraulic fluid to hydraulic cylinders that controlled the rover's articulation. The rover could turn a maximum of 45 degrees with a wheelbase of 190 cm. The engine throttle was connected to an onboard Kohler Command 20HP engine (CH20S, Kohler Co, Wisconsin, USA) with a maximum of 2500 RPM and powered the tandem variable- and fixed-rate pumps. The front tires were connected to a rotary encoder to provide feedback on the movement of the rover along the crop rows. Left/right articulation was controlled by using relays connected to the DCV.

5.3.2 Real-Time Kinematic GNSS and Network Transport of Radio Technical Commission for Maritime Services (RTCM) via Internet Protocol (NTRIP)

The RTK-GNSS receiver used to acquire the global position of the rover used an NTRIP provider (eGPS Solutions, Norcross, GA) to obtain differential correction through the internet

using a Verizon modem (Inseego Jetpack MiFi 8800L, Verizon Wireless, New York, NY) (Figure 5.3). The GNSS correction signal was obtained using the NTRIP signal with a mounting point within 3 kilometers of the test plot and downloaded to the RTK-GNSS through a Verizon Hotspot and wireless signal. NTRIP servers received the message from the base RTK-GNSS receivers connected to it. A data plan subscription was required to use the modem to acquire data through the internet from the eGPS base station network instead of using a local base station. The service to our NTRIP provider was registered, and a username, password, and I.P. address (mount point) to connect to the NTRIP provider via Internet Protocol were provided. Using NTRIP was advantageous because GNSS corrections were acquired without the need to set up a base station.

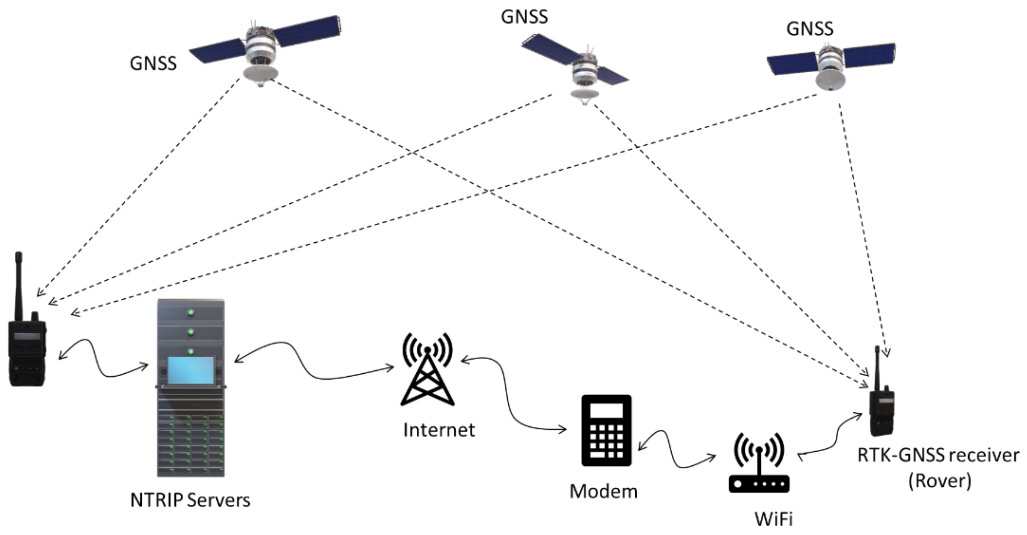


Figure 5.3. Context diagram of the Network Transport of RTCM via Internet Protocol (NTRIP)

5.3.3 Calibration of the Potentiometer, two IMUs, and two Encoders

The potentiometer (at point H in Figure 5.4) measured the articulation angle. Figure 5.4 shows the aerial view of the right turning center articulated rover. When the rover was straight, P_1 is parallel to P_2, and the potentiometer digital signal read 493. The length of l_1 and l_2 was

0.91m each. To obtain the articulation angle γ, manual measurements were made and applied using the Cosine rule progression below(Equation 1);

$$L^2 = {l_1}^2 + {l_2}^2 + 2 * l_2 * l_1 * \cos\theta$$

$$\theta = \cos^{-1}\left(\frac{{l_1}^2 + {l_2}^2 - L^2}{2 * {l_1}^2 * {l_2}^2}\right)$$

$$\gamma = \pi - \cos^{-1}\left(\frac{{l_1}^2 + {l_2}^2 - L^2}{2 * {l_1}^2 * {l_2}^2}\right)$$

$$\gamma = \pi - \cos^{-1}\left(\frac{0.91^2 + 0.91^2 - L^2}{2*0.91^2*0.91^2}\right) \quad (5.1)$$

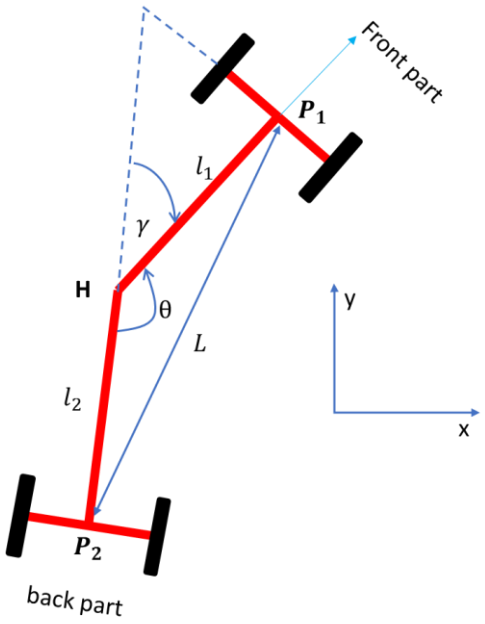
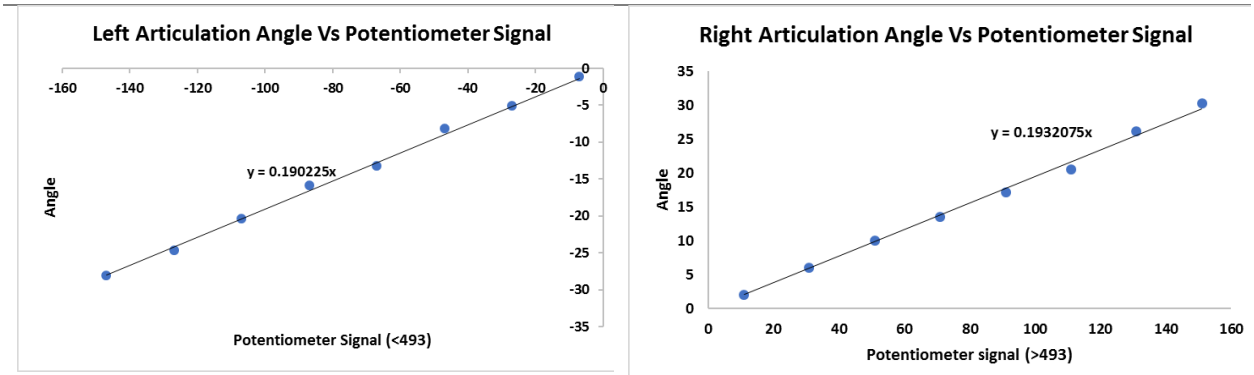


Figure 5.4. Potentiometer Calibration. Potentiometer calibration involves measurements of the voltage reported by the potentiometer in relation to the changing angle θ of the rover when turning left or right. Assume all the points are in a cartesian coordinate system.

To calibrate the angle, the rover was turned left or right. The angle was measured by the potentiometer, which was digitized with a 10-bit ADC, and the signal values ranged from 0 to 1023. The angle was recorded when turning left and right in 20 digital signal intervals from 493 (Figure 5.5a and b). Assume 493 as the center position and going left is negative while going right is positive. The angle γ was plotted together with the potentiometer signal (Figure 5.5). The potentiometer signal decreased when turning left and increased when turning right. The equation obtained from the plots for the left was y = 0.190225x, and for the right was y = 0.1932075x (Figure 5.5). The equations were implemented in algorithm 1 (Table 5.1) at lines 15 to 19. The left/right equations were slightly different due to potentiometer errors, human errors, and slight misalignment of the vehicle.



(a) (b)

Figure 5.5. The calibration of the potentiometer and articulation angle. The left image (5a) presents the relationship of the left articulation angle versus the potentiometer signal. The left image (5b) shows the relationship of the right articulation angle versus the potentiometer signal. The potentiometer signal presented on the graph is the difference between the reported value and center value (493).

Both IMUs were calibrated as advised by the manufacturer's users guide (Phidgets Inc., Calgary, CA). The magnetic error correction was done by the compass calibrator software

downloaded from the Phidgets website. The two IMUs were placed at two different locations on the vehicle, as described in the section "Robot components and System Setup." The IMU was calibrated by connecting the IMU to the embedded computer, which had the Phidget compass calibration program installed. The magnetic field estimated value for Tifton, Georgia, was 0.47459 T obtained online from the NOAA website (http://www.ngdc.noaa.gov/geomag-web/#igrfwmm). After entering the magnetic field value, the program was started, and the rover was driven in a circle behind the Engineering Annex fields (31.475340N, 83.528968W) in Tifton, Georgia, to generate the calibrated compass parameters. After the calibration, the IMUs were then used for localization and navigation experiments.

Encoder calibration was conducted by finding the circumference of the rover wheels and then converting the signal of the encoder to distance for each encoder count. The rotary encoders used a 10-bit Analog-to-digital converter. To make sure that the encoders were accurately calibrated, the tires of the rover were rotated 360°, and the count of the encoders increased from 0 to 1023. Since the circumference of the tire was 1915.1mm, the distance per count (resolution) was 1915.1/1024 = 1.87mm.

5.3.4 Robot Navigation Systems

The navigation system consisted of the embedded development kit and the rover navigation controller. The rover used two algorithms to control navigation: modified pure pursuit and Proportional control (Bergerman et al., 2016; Botterill et al., 2017; Coulter, 1992; Rains et al., 2014; Samuel et al., 2016). The system used a predefined path of the GNSS signal to pass over the rows. The path was obtained by recording the rover path as it was manually driven down cotton rows (Figure 5.6). The predefined path was then used by the rover to navigate autonomously.

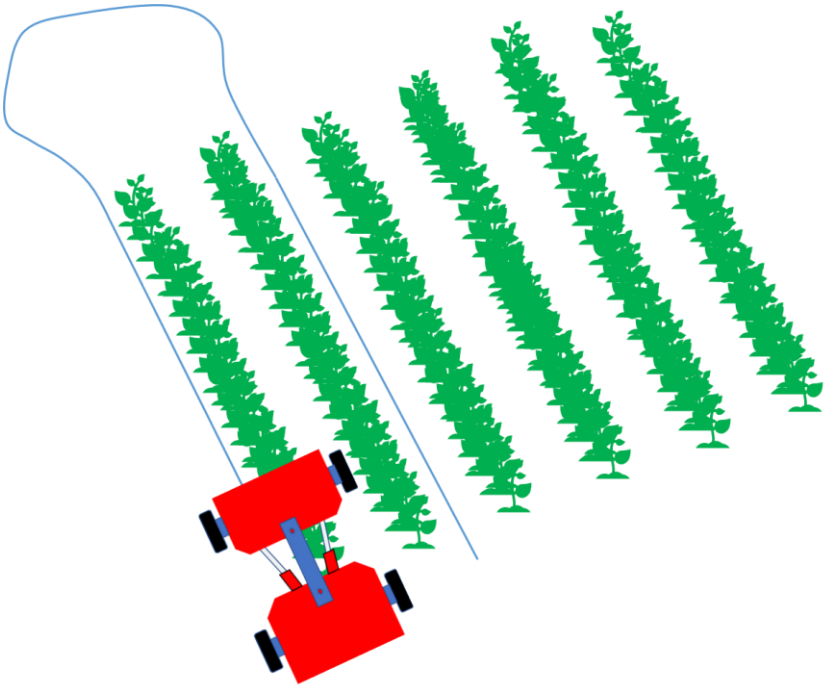


Figure 5.6. The rover driving along the cotton rows. Blueline is the path recorded by the rover after finishing one lap

Since the rover used six sensors (two IMUs, potentiometer, two encoders, and RTK-GNSS) to navigate (Figure 5.7), the Extended Kalman Filter was implemented for simultaneous localization and navigation (Backman et al., 2012; Moore & Stouch, 2016; Post et al., 2017; Wan & Nelson, 2001). Sensor fusion was achieved by using the open-source ROS library "Robot localization," which provided sensor fusion and nonlinear state estimation for IMUs, encoders, and GNSS. The IMUs published two ROS topics (imu_link1/data and imu_link2/data), encoders published wheel odometry (/wheel_odom), and RTK-GNSS published /gps/fix signal (Moore & Stouch, 2016). The EKF localization (Figure 5.7) used the nav_sat_transform library to integrate fixed data from the RTK-GNSS (Post et al., 2017). Basically, "navsat_transform_node" required three sources of information: the robot's current pose estimate in its world frame, an earth-

referenced heading, and a geographic coordinate expressed as a latitude/longitude pair (with optional altitude) (http://docs.ros.org/).

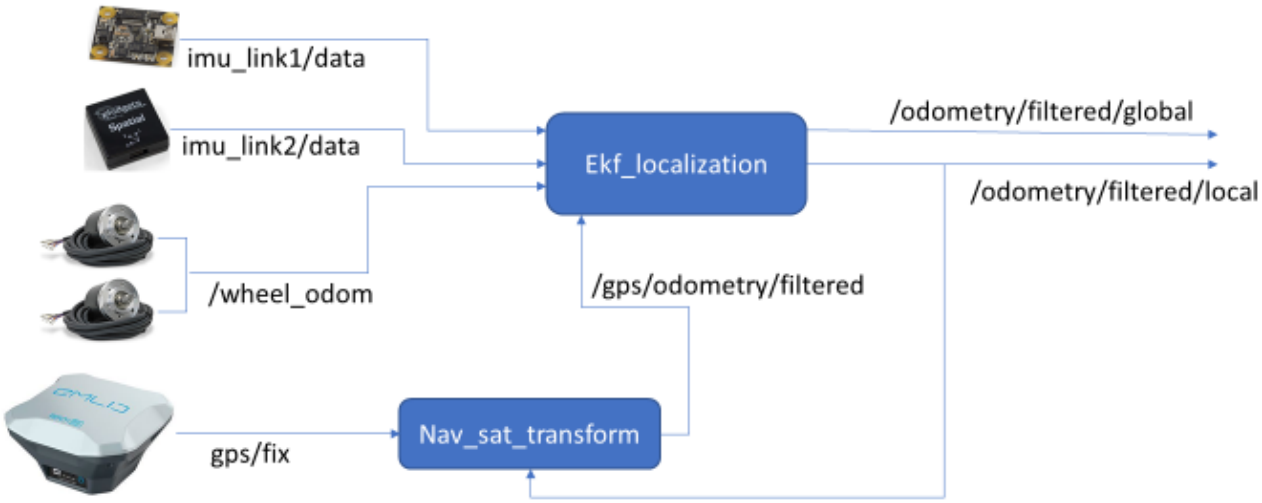


Figure 5.7. Simultaneous localization and navigation of the rover using dual Extended Kalman Filter (dual EKF)

EKF localization was implemented (Figure 5.8) as a dual EKF method that involved running two EKF's concurrently. The state and model could be estimated from the noisy observations of the IMU, wheel odometry, and GPS fix signal.

$$x_{k+1} = F(x_k, u_k, w) + v_k \quad (5.2)$$

$$y_k = H(x_k, w) + n_k$$

We considered the nonlinear problem in which system states ($\mathbf{x}_k$) and model parameters (**w)** were simultaneously estimated from the observed signal ($\mathbf{y}_k$). The processed signal ($\mathbf{u}_k$) determined the dynamical system, while the exogenous input noises ($\mathbf{n}_k$ and $\mathbf{v}_k$) were calculated (Equation 2). In Figure 5.8, using current model estimates $\mathbf{w}_k$, an EKF state filter calculated the new state in every step. Then, it calculated the fresh weights of the current state estimate $\mathbf{x}_k$. The model structure **F** and **H** are multilayer neural networks, while **w** are the weights (Wan &

Nelson, 2001). In this case, the model used IMUs, and encoders to generate localization estimates and then added RTK-GNSS to make global localization estimates of the rover position.

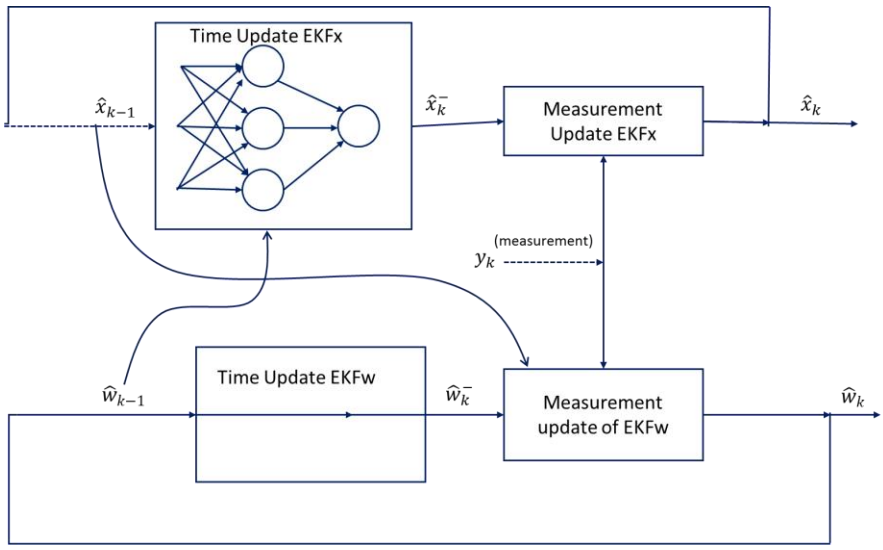


Figure 5.8. The dual extended Kalman filter. The method utilizes two concurrently running EKFs. State estimates are done by the top EKF using $\hat{w}_{k-1}$ for the time update.While weight estimates are generated using the bottom EKF that does measurement updates using $\hat{x}_{k-1}$ (Wan & Nelson, 2001).

After getting the state estimates of the rover, a modified pure pursuit and PID were used to control the rover's navigation.

5.3.5 Modified Pure Pursuit

Pure pursuit (P.P.) is a technique that computes the current vehicle position relative to a goal and then determines the curvature that would bring the vehicle back to the predefined or designated path. P.P. chooses the goal that is some distance in front of the rover. It looks ahead and determines the articulation of the tires to get into the path. The look-ahead distance changes depending on the curvature of the path and speed of the vehicle.

The rover achieved pure pursuit tracking by following six steps; determine the current location of the rover, find the path point closest to the rover, find the goal location, transform the goal location to rover coordinates, calculate the curvature and request the rover to set the articulation to that curvature and then, update the vehicle's position (Coulter, 1992; Samuel et al., 2016).

Figure 5.9 illustrates the center-articulated rover with all the sensors at the front, including GNSS steering to pursue the red dot (goal point) at a distance L. The radius of turning is r. At the same time, s, which is equal to x, is the relative distance of the rover to the goal point.

$$x+d = r$$

$$x = s$$

$$x^2 + y^2 = L^2 \qquad (5.3)$$

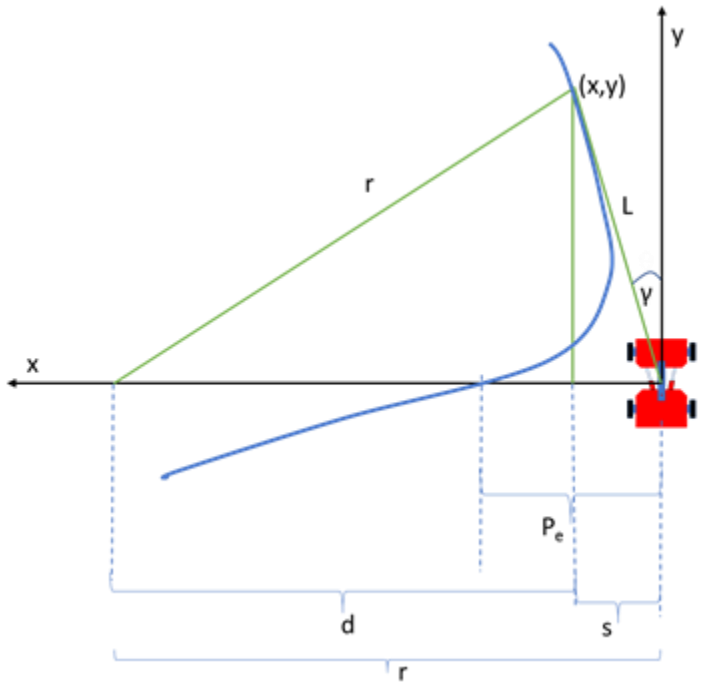


Figure 5.9. The geometry of the Pure Pursuit algorithm. Blueline is the designated path to pursue while point (x,y) is the goal. Black lines represent cartesian coordinates axis (y and x) while the

green lines show the geometry of turning. The notation; r is the turning radius, γ is the articulation angle, P_e is the Path error, (x,y) is the goal of the rover, L is the distance from the rover to goal, d is the horizontal distance of the goal from the center of the turning circle and s is the horizontal distance of the rover to the goal.

Using the relationship of x, y, r, and L in Figure 5.9, the curvature C can be derived. If x+d = r, and $d^2 + y^2 = r^2$, r can be found by computing its' relationship with x, y coordinates, $r = (x^2 + y^2) / (2x)$ (Rains et al., 2014). Then, the turning radius r becomes;

$$r = \frac{L^2}{2*x} \tag{5.4}$$

Hence, the curvature **C** which is 1/r is given as;

$$C = \frac{2*x}{L^2} \tag{5.5}$$

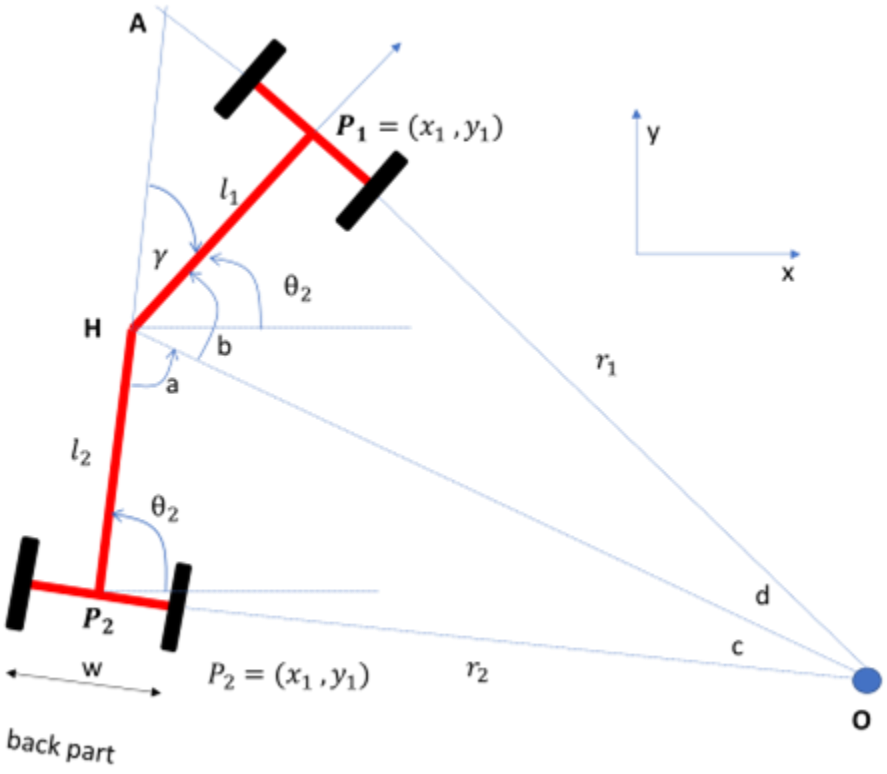


Figure 5.10. The geometry of the center-articulated rover while turning. $\mathbf{r}_1$ is the distance from the origin of the turning circle O (Corke & Ridley, 2001).

Consider Figure 5.10 that shows the geometry of the turning rover (Corke & Ridley, 2001). At the point H, $a + b + \gamma = 180^0$. So, $b + d = 90^0$ and $a + c = 90^0$. Therefore, by substituting constants makes $\gamma = c + d$. γ is the articulation angle while θ is the heading angle.

Now, consider the right-triangle ΔOP_2A,

$$l_2 + \frac{l_1}{\cos\gamma} = r_2 \tan\gamma$$

$$(r_1 + l_1 \tan\gamma)\sin\gamma = l_2 + \frac{l_1}{\cos\gamma} \qquad (5.6)$$

Then, by simplifying Equation 5.6, the turning radii, r_1 and r_2 can be found;

$$r_1 = \frac{l_1 \cos\gamma + l_2}{\sin\gamma}$$

$$r_2 = \frac{l_2 \cos\gamma + l_1}{\sin\gamma} \qquad (5.7)$$

Since the rover was an approximately symmetric vehicle, the distance from the center (L_f) to back tires or front tires was 91 cm. hence, we can simplify the radii equation above,

$$r = r_1 = r_2 = \frac{L_f \cos\gamma + L_f}{\sin\gamma} \qquad (5.8)$$

Then, using trigonometry rules, $\sin 2a = 2\sin a \cos a$ and $\cos 2a = = 2\cos^2 a - 1$

$$\frac{r}{L_f} = \frac{\cos\gamma + 1}{\sin\gamma} = \frac{2\cos^2\frac{\gamma}{2}}{2\sin\frac{\gamma}{2}\cos\frac{\gamma}{2}}$$

Hence, by simplifying the trigonometry,

$$\tan\gamma = \frac{L_f}{r} \quad (5.9)$$

insert equation (5.4) which makes;

$$\tan\gamma = 2 * L_f * \left(\frac{x}{L}\right)\frac{1}{L}.$$

$$\tan\gamma = 2 * L_f * (\sin\theta)\frac{1}{L}.$$

$$\gamma = 2 * \tan^{-1}\left(\frac{2*L_f*\sin\theta}{L}\right) \quad (5.10)$$

So, equation 5.10 shows the relationship of articulation angle γ to look-ahead distance L, half-length of the rover L_f and heading angle θ. Consider the vehicle at state (x_k,y_k) is articulating to the next goal (x_{k+1}, y_{k+1}). Next heading angle θ_{k+1} in relation to the current heading angle θ_k can be found by;

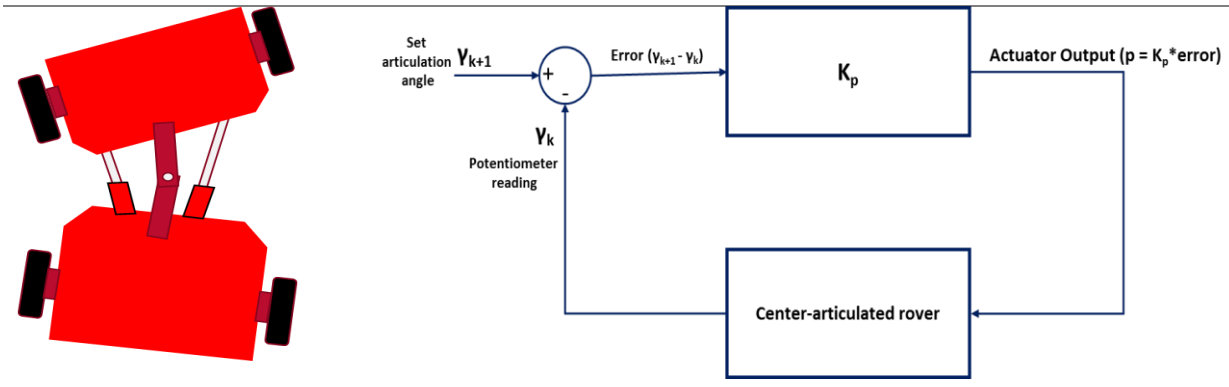
$$\theta_{k+1} = \theta_k - tan^{-1}\frac{x_{k+1} - x_k}{y_{k+1} - y_k} \quad (5.11)$$

Because the rover has a slow turning action when moving, the horizontal distance of the vehicle from the goal should be increased by a factor "K" (See Equation 5.12). The closest distance of the rover to the path is the path error. P_e is found by calculating the perpendicular distance of the vehicle to the designated path. The next position of the path was moved further from the position by the factor K multiplied by P_e. It was the modified Pure Pursuit of the center-articulated rover (Equation 5.12). The value K is a rover-dependent number that should be obtained by testing and experimentation to achieve the best path tracking.

$$x_{k+1} = x_{k+1} + K * P_e \quad (5.12)$$

5.3.6 Proportional Control of the Articulation Angle

The rover turned to the target articulation angle γ, as described in Equation (5.10) by using proportional control. The current angle γ_k and required target angle γ_{k+1} was used to find the error that was used to control the angle. The gain Kp used was set to 1. The articulation angle was controlled by the hydraulic cylinder linear actuators. The actuators were connected to two relays. Proportional control was used since the actuators were only controlled by on/off relays, which limited the ability to control actuator speed. The two relays were connected to the navigation controller pin 7 for the left actuator and pin 8 for the right actuator. The relays used to control the linear actuator were Single Pole Double Throw (SPDT) relays. SPDT relays provided a capability to control the linear actuators by switching into three different connections; normally closed, normally open, and common (Oberhammer et al., 2006). So, if the rover turned left, the left actuator retracted while the right actuator extended until a desired left articulation angle was achieved. Also, if the rover turned right, the right actuator retracted while the left actuator extended until a desired right articulation angle was achieved. It means the circuit opened for the right actuator and closed for the left one. When the angle required was attained, it switched to common.



(a) (b)

Figure 5.11. Proportional control of the articulation angle (a) The rover when turning left (b) the proportional control to achieve a targeted articulation angle

Table 5.1. Algorithm describing the proportional control of the articulation angle.

Algorithm 1: Proportional control of the articulation angle

Input: Angle reported by the high precision potentiometer γ_k , target angle γ_{k+1} and threshold E_t

Output: p which is equal to Kp*(γ_{k+1} - γ_k)

5: Gain Kp is equal to 1

6: p <- Kp * (γk+1 – γk)

7: WHILE p > Et

12: Declare and assign 0 to increment i

13: Declare and assign 0 to temp

14: WHILE i < 20

15: Delay for one microsecond

16: Read the analog signal from the high precision potentiometer γ_k

17: temp add the γ_k to temp

18: Increment i

19: }

20: Get the average temp

21: Assign temp to γ_{k+1}

22: IF temp > analog signal 493

23: γ_{k+1} = (temp - 493) * 0.1932075;

24: ELSE

25: γ_{k+1} = (temp - 493)*0.190225;

26: END IF

27: p <- Kp * (γ_{k+1} - γ_k)

28: IF p > -Et

29: Set the left relay HIGH

30: Set the right relay LOW

31: ELSE IF -p < Et

32: Set the left relay LOW

33: Set the right relay HIGH

34: ELSE

35: Set the left relay LOW

36: Set the right relay LOW

37: END IF

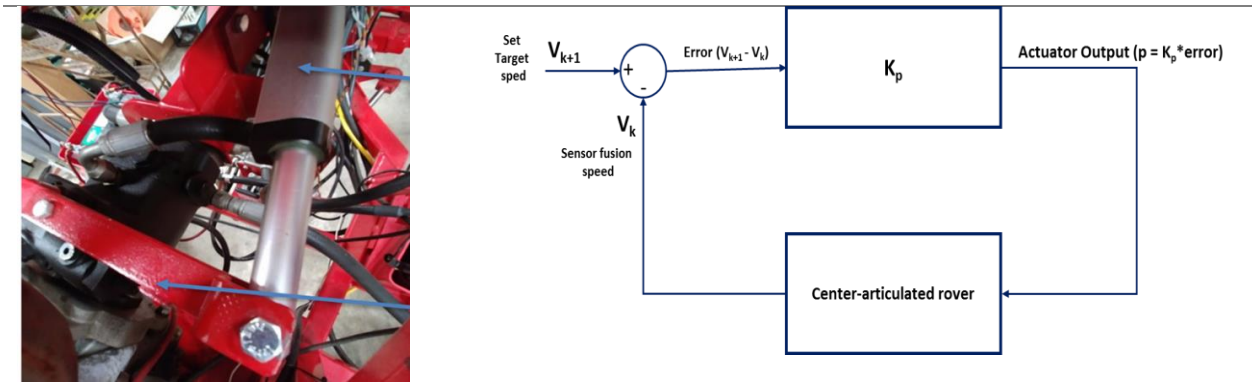
16: END WHILE

17: Return all the error

5.3.7 Proportional Control of the Speed of the Rover

The speed of the rover was controlled by using the proportional controller. The controller had a gain of 22 and a target speed of 1.2 m/s. A PID controller was not implemented because we were not targeting precise speed control; hence proportional control was enough. The controller calculated error from the difference between target speed and the speed estimated by the EKF from encoders, IMU, and GNSS readings. The acceleration of the rover was controlled by extending and retracting the linear actuator, which sets the swashplate angle. By changing the

angle of the swashplate, hydraulic fluid flow rate to the hydraulic motors turning the wheels was changed and effectively changed the speed of the rover. The vehicle remained stationary when the angle was set at 90^0. When the angle of the swashplate changed from 90^0 to 60^0, the vehicle moved backward while increasing the speed to the minimum backward speed. Also, when the angle of the swashplate changed from 90^0 to 120^0, the vehicle moved forward with the increase to the maximum forward speed. However, the change in actuator movement required the swashplate angle to be more than 108^0 to move forward or less than 80^0 to move backward created by an inherent deadband in the pump performance. The deadband was caused by wear (leakage) in the hydraulic system and mechanical compliance of connectors. The neutral position was held until the actuator extended or retracted by more than 2.5 cm.



(a) **(b)**

Figure 5.12. (a) The arrow points to the red arm that controls the swashplate angle. Another arrow points to the grey linear actuator controlled by the navigation controller (b) The proportional control diagram of the speed of the rover.

5.3.8 Waypoints Collection and Cubic Spline Interpolation of the Waypoints

The rover was driven around to obtain the waypoints at the UGA grounds behind the engineering annex located at (31.475340N, 83.528968W) and the Entomology farm

(31.472985N, 83.531228W) near Bunny Run Rd. in Tifton Georgia. The rover recorded the waypoints using RTK-GNSS at the rate of 5Hz. Since the rate 5Hz provided very few data points, the algorithm to interpolate the points using a cubic spline interpolation method was developed. The points were changed to UTM data points. Both fields were located at Zone 17R. Cubic spline interpolation was done by assuming data points were connected by a line whose equation was a polynomial degree of three (McKinley & Levine, 1998). It was assumed the datapoints given were [x_i, y_i], and no two points xi were equal to each other, and the x_i was in sequence such that $x_0 < x_1 < x_2 < \ldots < x_n$. The Spline function $S(x_i) = y_i$. For each subinterval $[x_{i-1} < x < x_i]$, the cubic function is given as $C_i = a_i + b_i x + c_i x^2 + d_i x^3$. So, for every subinterval the Spline function S(x) can be assumed as;

$$S(x) = \begin{cases} C_1(x), & x_0 < x < x_1 \\ C_2(x), & x_1 < x < x_2 \\ \vdots & \\ C_i(x), & x_{i-1} < x < x_i \\ \vdots & \\ C_n(x), & x_{n-1} < x < x_n \end{cases}$$

Table 5.2 is an algorithm to calculate the values of a_i, b_i, c_i, and d_i for every interval of the dataset. More values make it easy to calculate the relative position of the rover to the target path perpendicularly.

Table 5.2. Algorithm to estimate subinterval of UTM waypoints data intervals

Algorithm 2: Cubic Spline Algorithm to estimate subinterval of UTM waypoints data intervals
Input: x_0, x_1, x_2, …, x_n; $a_0 = f(x_0)$, $a_1 = f(x_1)$, $a_2 = f(x_2)$, ….. $a_n = f(x_n)$
Output: ai, bi, ci, di for j = 0,1,2,…..,n-1
1: Assign $P_0 = 0$

2: Assign $Q_0 = 0$

3: Assign $R_0 = 0$

4: FOR j=0 TO n-1

5: Set the interval difference hi <- $x_{i+1} - x_i$

6: Set $\alpha_1 = (3/h_j)*(a_{j+1} – a_j) – (3*(a_j – a_{j-1}) /h_{j-1})$

7: END FOR

8: FOR j = 1 TO n-1

9: $P_j = 2*(x_{j+1} – x_{j-1}) – h_{j-1}Q_{j-1}$

10: $Q_j = h_j / L_j$

11: $R_j = (\alpha_j – h_{j-1}*R_{j-1})/P_j$

12: END FOR

13: Assign $P_n = 1$

14: Assign $R_n = 0$

15: Assign $c_n = 0$

16: FOR i = n-1 TO 0

17: //get the remaining values of the cubic spline b,c, and d

18: $c_i = Z_i – Q_i * c_{i+1}$

19: $b_i = (a_{i+1} – a_i)/h_i – h_i*(c_{i+1} + 2*c_i)/3$

20: $d_i = (c_{i+1} – c_i) / 3*h_i$

21: Return all the values of ai, bi, ci, di

5.3.9 Preliminary Experiment

Preliminary experiments were conducted on 10th October 2019 at the UGA grounds to study the navigation behavior of the rover when parameters such as ROS update rates, look ahead, and path error were altered. These tests served to calibrate the system to perform well in the field. The preliminary experiment involved four tests;

1. A fast ROS rate was set to 10Hz,
2. A short look ahead was set at 1m,
3. Path error was set to 0 which means $K * P_e = 0$, and
4. An optimal condition (long look-ahead is 3m, path error is 1.5 times path error and slow ROS rate at 1Hz).

The experiment was conducted by setting the rover to follow the prescribed path. The predefined path was obtained by moving the rover manually and collecting the GNSS waypoints. Later, the rover was set to autonomous mode to follow the pre-planned path so that behavior and characteristics could be observed and tuned.

5.3.10 Field Experiment

The field experiment was conducted at the Horticulture hill farm (31.472985N, 83.531228W) near Bunny Run Road in Tifton, Georgia, after establishing the calibrated parameters of the rover. The field (Figure 5.13) was planted on 19th June 2019 using a tractor (Massey-Fergurson MF2635 tractor, AGCO, Duluth, GA) and a 2-row planter (Monosem planter, Monosem Inc, Edwardsville, KS). The cotton seeds (Delta DP1851B3XF, Delta & Pine Land Company of Mississippi, Scott, MS) were planted every two-row and skipped two rows. The rows were 36-inch (91.44 cm) wide, and the seed spacing was 4-inch (10.16 cm). The cotton field was undefoliated, and most of the cotton bolls were open already at the time of the

experiment. Three tests were conducted for navigation on 18.5m rows on 21st October 2019. The path was obtained by driving the rover over one of the two rows of cotton plants and collect the waypoints (Figure 5.6). The experiments were conducted by setting the rover to follow the predefined path autonomously.

Figure 5.13. The appearance of the cotton at the time of the field experiment.

5.4 Results and Discussions

5.4.1 Preliminary Experiment

1	*2*

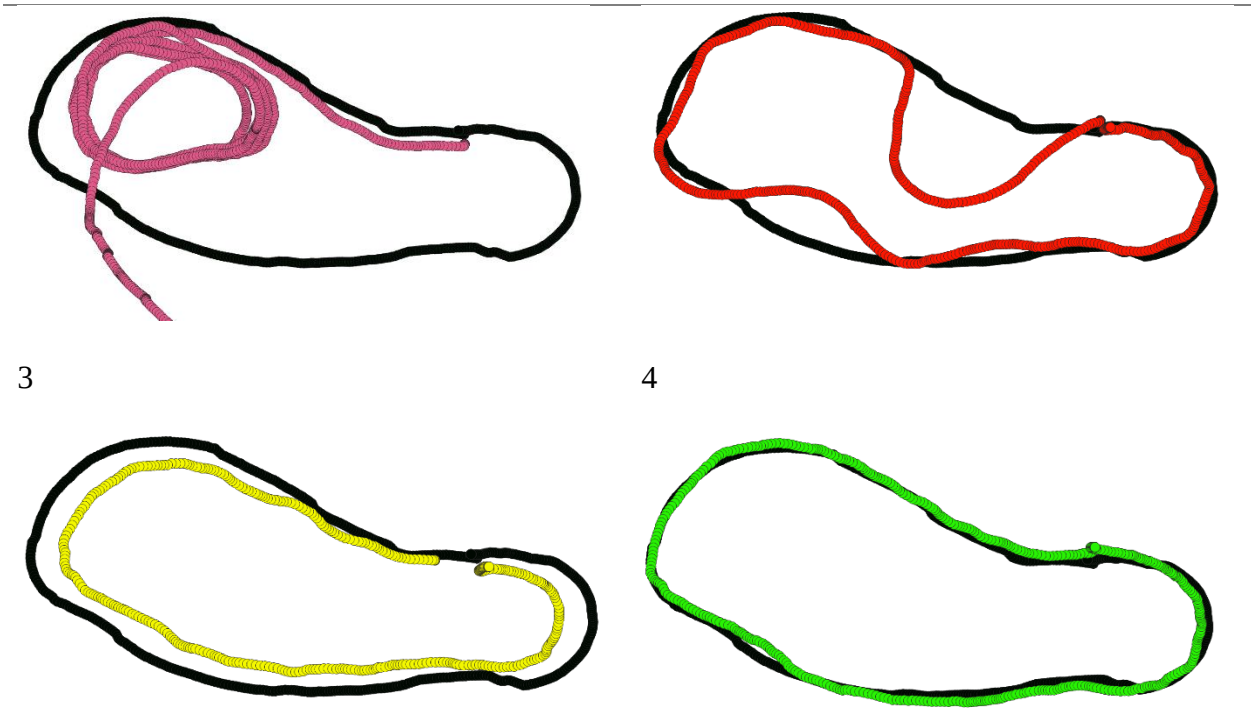


Figure 5.14. . Performance of the rover when the (1)ROS updates were high (10Hz), (2)Short look ahead distance (1m), (3) no path error correction was applied and (4)successful path tracking when look-ahead was 3m, K was 1.5, and ROS updates at 1Hz. Blackline is the predefined waypoints, while colored lines represented the rover passes for each condition from 1 to 4.

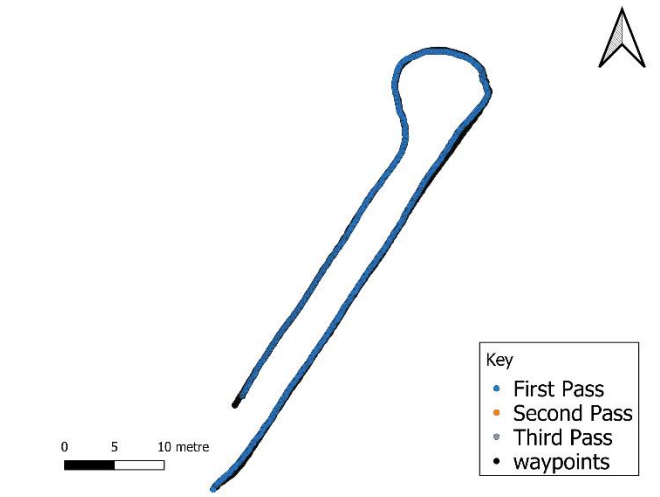
The results of the preliminary experiments show that the rover navigation tracking was negatively affected when ROS update rates were increased, or no path error correction was applied, and when very short look-ahead was used (Figure 5.14). Pure pursuit algorithm has a goal to make sure that the rover regains the designated path by articulation when it loses and maintains when it was on the predefined waypoints. Fast ROS update rates affected system performance since the mechanical responses of the machine were slow compared to the update rate provided by the controller. This short time between the new input reading of the system meant that the reaction time of the vehicle to the control signal was slow, and the vehicle

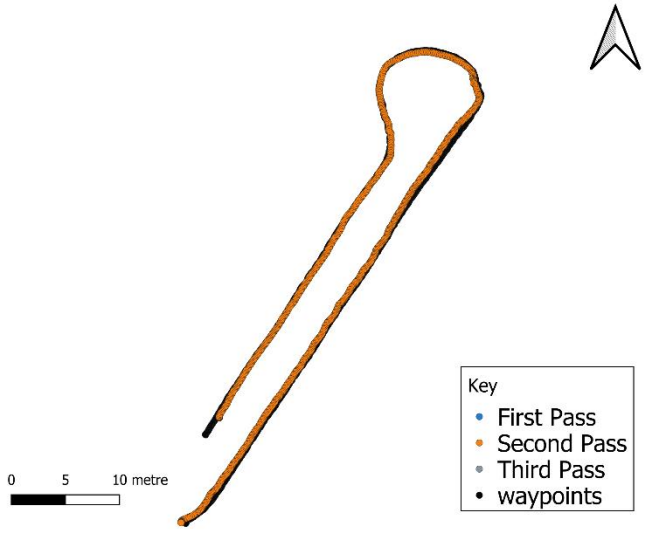
reaction was always lagging behind the control decision, causing the rover to lose tracking control. The ROS rate update was a significant parameter if it was not set right. The performance was obtained with short look-ahead distances. With a short look ahead distance, the rover tried to move quickly to regain the path it has lost. However, this action caused the rover to overshoot the path and oscillate along the prescribed path.

With no path error corrections (when K was Zero), the rover could never converge to the path over time. Figure 5.15(3) shows how the vehicle was not able to converge to the path, which means the error was consistently maintained. To avoid this behavior, modified pure pursuit increased the error of the system by 1.5 using Equation (5.12) to force the system to converge to the path. If the error was significant, the system amplified the error forcing the rover to act aggressively and quickly. When the error was small, the system acted slowly because the amplification of the error also became small (Refer to Equation 5.12).

5.4.2 Field Experiments

Figure 5.15 shows the navigation path traces as recorded by the GNSS for the three experiments. The rover performed well visually. The third pass can clearly show that the rover moves out of the path during the end-of-row turning. Contributing factors were wheel slipping as the rover attempted to turn compounded by control signal updates requiring turning too quick for the rover to respond a follow the designated path.


0 5 10 metre
Key
First Pass
Second Pass
Third Pass
waypoints


0 5 10 metre
Key
First Pass
Second Pass
Third Pass
waypoints

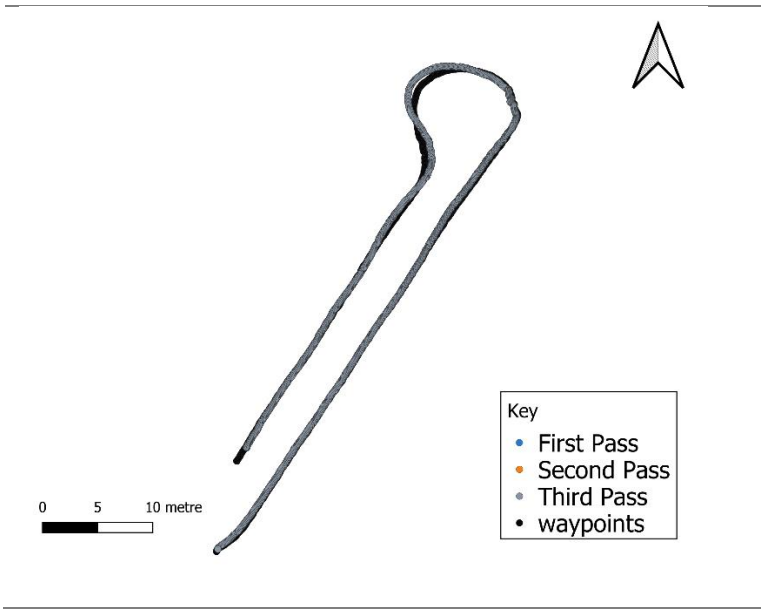


Figure 5.15. Path tracking of the prescribed path (black pass). The blue, orange, and gray passes are the GPS generated path of the rover when following the rows using the prescribed path (black). The blue, orange, and gray path traces are first, second, and third navigation pass experiments, respectively.

The rover was able to follow the path and return to enter the next cotton row. The absolute error distribution was determined to characterize rover navigation behavior. The rover performed well, as Figure 5.16 shows most of the path errors were less than 15 cm (0.15 m).

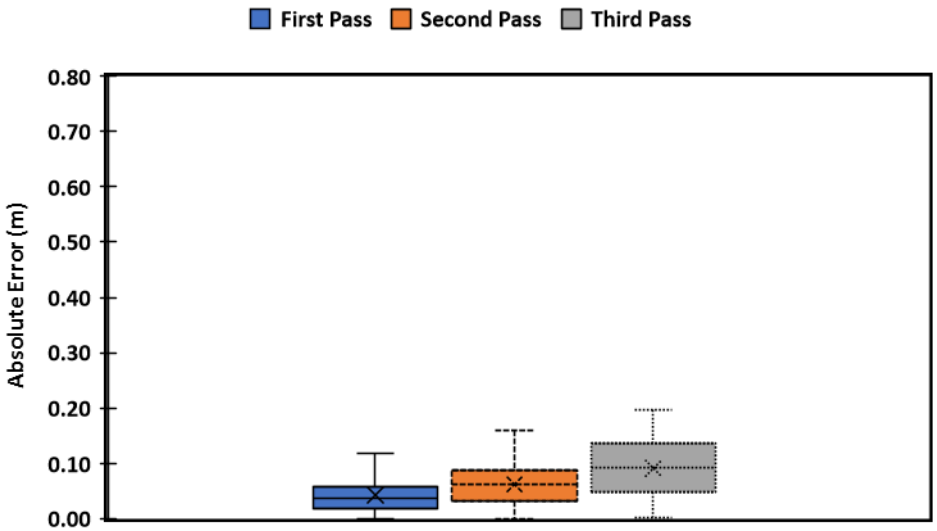


Figure 5.16. The average of absolute errors (A.E.) in the first, second, and third non-turning passes (blue, orange, and gray boxes, respectively). The boxes present the first quartile to the third quartile of A.E. while the whiskers show maximum values and minimum values of each of the passes. The 'x' shows the mean absolute error (MAE) for each of the passes.

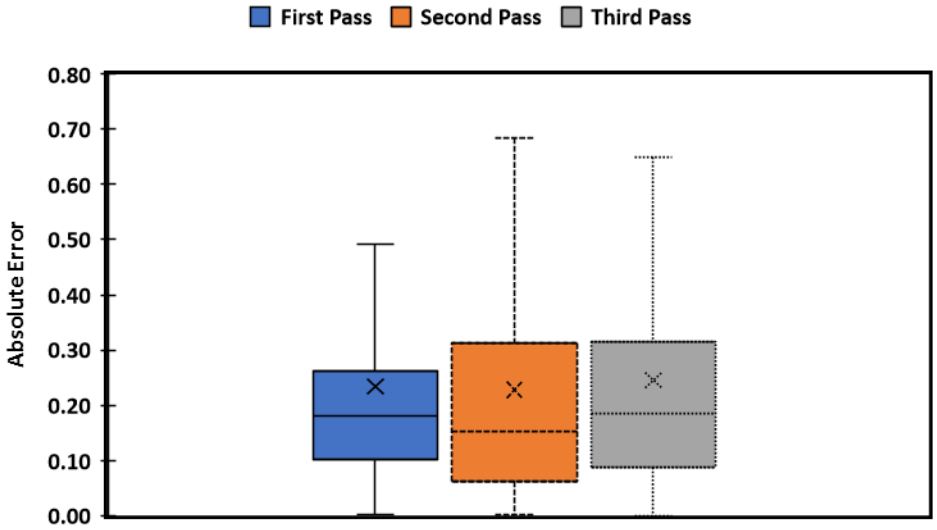


Figure 5.17. The average of absolute errors (A.E.) when turning for the first, second, and third passes (blue, orange, and gray boxes, respectively). The rover had no significant difference in turning performance. The boxes present the first quartile to the third quartile of A.E. while the

whiskers show maximum values and minimum values of each of the passes. The 'x' shows the mean absolute error (MAE) for each of the passes.

Table 5.3. The mean absolute error, and standard deviations of the three passes along the cotton rows.

Mean ± Std. dev (m)	**1st pass**	**2nd pass**	**3rd pass**	**Overall Performance**
1st Row	0.048 ± 0.036	0.048 ± 0.035	0.066 ± 0.0046	0.053 ± 0.041
Turning	0.233 ± 0.198	0.227 ± 0.211	0.244 ± 0.211	0.235 ± 0.206
2nd Row	0.036 ± 0.024	0.081 ± 0.028	0.115 ± 0.043	0.070 ± 0.046
Overall (1st and 2nd Rows)	0.042 ± 0.032	0.062 ± 0.036	0.091 ± 0.053	0.061 ± 0.044

Figure 5.16 and Table 5.3 shows that the rover performance was adequate as it was safely navigating along the rows without driving over the plants. Most of the errors were less than 10 cm from the prescribed path. The rover was perfectly converging back to the predefined path except during the turning maneuver, which provided a significant challenge to the rover. The first and second tests were excellent, but the third one had trouble turning in the muddy end-row that caused the wheels to slide. The MAE of 0.042m, 0.062m, and 0.091m for first, second, and the third pass respectively show that the system performance was acceptable. However, the MAE was significant when turning as it ranged at 0.233m, 0.227m, and 0.244m for the first, second, and third pass, respectively (Figure 5.17). Slippage had effects only in the third pass when turning caused errors to increase to 0.115m in the second pass. Also, the results show a modified

simple pursuit algorithm was struggling to compensate for errors in the third pass. However, we were not as concerned about the errors when turning.

5.5 Conclusion

An autonomous navigation algorithm using a modified pure-pursuit algorithm for a multi-purpose rover was designed, developed, and tested in this study. Results showed that the rover could autonomously navigate safely along the rows of cotton, turn around and enter a second row. Results from the preliminary testing and field testing showed that an affordable single-frequency RTK-GPS could be used with other sensors and a sensor fusion technique to achieve acceptable navigation accuracy. There was an increase in errors when the rover turned that did not impede the rover's ability to enter the next row. As a result, the multi-purpose rover can follow rows and operate autonomously to perform any number of tasks when a predefined path in GNSS coordinates is available or created. Small, intelligent, multi-purpose vehicles could be provided paths and prescriptions for spraying, planting, scouting, or harvesting. Small rovers would eventually need to operate in teams to cover larger acreages with rover-to-rover communication to create built-in task optimization.

CHAPTER 6

AUTONOMOUS CENTER-ARTICULATED HYDROSTATIC COTTON HARVESTING ROVER USING VISUAL-SERVOING CONTROL AND A FINITE STATE MACHINE[5]

[5] Fue, K., Barnes, E., Porter, W., Li, C., and Rains, G., Submitted to *Electronics*, July 1, 2020.

6.1 Abstract

Multiple small rovers can repeatedly pick cotton as bolls begin to open until the end of the season. Several of these rovers can move between rows of cotton, and when bolls are detected, use a manipulator to pick the bolls. To develop such a multi-agent cotton harvesting system, each cotton harvesting rover would need to accomplish three motions: the rover must move forward/backward, turn left/right, and the robotic manipulator must move to harvest cotton bolls. Controlling these actions can involve several complex states and transitions. However, using the ROS-independent finite state machine (SMACH), adaptive and optimal control can be achieved. SMACH provides task level capability for deploying multiple tasks to the rover and manipulator. In this study, a center-articulated hydrostatic cotton harvesting rover using a stereo camera to locate end-effector and pick cotton bolls was developed. The robot harvested the bolls using a 2D manipulator that moves linearly horizontally and vertically perpendicular to the direction of the rover's movement. The boll's 3-D position was determined by calculating stereo camera parameters, and the decision of the finite state machine guided the manipulator and the rover to the destination. PID was deployed to control the rover's movement to the boll. We demonstrate preliminary results in an environment simulating direct sunlight as well as in an actual cotton field. The system achieved a picking performance of 17.3 seconds per boll and the average Action Success Ratio (ASR) of 88.9% in a simulated environment. In each mission, the system was able to detect all of the bolls but one. Furthermore, it completed the task by navigating at a speed of 0.87 cm per second while collecting 0.06 bolls per second. In the field experiments, the rover picked cotton bolls at an average Action Success Ratio (ASR) of 78.5 with 38 seconds per boll, which is twice that of the simulated environment, and the end-effector was able to reach 95% of the bolls. In addition, it completed the task by navigating at a speed of

0.18 cm per second while collecting 0.03 bolls per second. The designed robot demonstrates that it is possible to use a cartesian manipulator for robotic harvesting of cotton; however, to reach commercial viability, the speed of harvest and successful removal of bolls (ASR) must be improved.

6.2 Introduction

Cotton is an essential commercial crop worldwide. The cotton industry in the U.S has been growing and is now the 3rd largest agricultural industry in the U.S, employing more than 200,000 people with a value of $25 billion per year (USDA/NASS, 2018). The U.S is third in the production of cotton in the world behind India and China. As a large industry, however, cotton production operations have faced multiple challenges, of which timely harvesting of quality cotton fiber is among the most pressing. Since its introduction in the 1950s, the practice of mechanical harvesting after defoliation has provided fast harvesting speed, but also substantial losses in quantity and quality of cotton (Burnard, 2017). Open cotton bolls can sit up to 50 days until picked when at least 60% to 75% of the cotton bolls are opened (UGA, 2019). This waiting time exposes the open bolls to harsh conditions that degrade quality. Any change that would reduce cotton losses, improve quality, and increase return on investment would be welcomed by the industry.

In most cases, the mechanical combine is huge and expensive (a 2019 picker costs around $725,000 and sits in storage more than nine months a year without being used). Cotton combines also weigh more than 33 tons, which can cause soil compaction that reduces land productivity. The maintenance of such machines is also expensive and complicated. Repairing breakdowns in the field can take days, which can reduce operating efficiency and expose bolls to further weather-related quality degradation. In addition, most of the machines use proprietary software

and hardware that prevents farmers from repairing their machines and, therefore, deny the use of the right-to-repair tools that they own (Waldman & Mulvany, 2020). Finally, cotton plants must be chemically defoliated to harvest, which can cause an additional expense to the farmer (UGA, 2019).

Furthermore, the labor shortage in agriculture is getting worse while the cost of available labor is skyrocketing (Zahniser et al., 2018). The emergence of robotics in agriculture, particularly in specialty crops, has created an opportunity in the domain of row crops such as cotton, which have received little attention until recently (Bergerman et al., 2016; Comba et al., 2010; Fue et al., 2020b; Ramin Shamshiri et al., 2018). To the best of our knowledge, there have been no commercial cotton harvest robots developed (Fue et al., 2020b). Most robotic harvesting investigations are examining how fruit can be picked individually to mimick conventional hand-harvesting (Bergerman et al., 2016; Fue et al., 2020b).

Consequently, robotics may provide an efficient and cost-effective alternative for small farmers unable to buy and own large machines (Bergerman et al., 2016; Comba et al., 2010; Naoshi Kondo & Ting*, 1998; Lowenberg-DeBoer et al., 2019). To achieve a robust robotic harvesting system, the robot designs must consider four aspects carefully: sensing, mobility, planning, and manipulation (Fue et al., 2020b). A predefined navigation path and RTK-GNSS can provide absolute path following for row crops, but the design of the manipulators must be considered carefully to match the harvesting speed and efficiency of harvest requirements to be economically viable for farmers. There have been several approaches proposed by scientists using high degrees of freedom (DOF) manipulators. Nonetheless, the systems have proven to be slow because of the extensive matrix calculations required to determine joint manipulations that move the end-effector to the fruit for harvesting (Hohimer et al., 2019). For row-crops with a

high number of fruits to harvest, the best option would be a capable manipulator that harvests while moving. A potential solution is to use multiple harvesting manipulators with low degrees of freedom.

In this study, a four-wheel center-articulated hydrostatic rover with a 2DOF cartesian manipulator that moves linearly, vertically, and horizontally was developed. The kinematics of the manipulator were analyzed to calculate the arm trajectories for picking cotton bolls. Image-based visual servoing was achieved using a stereo camera to detect and localize cotton bolls, determine the position of the end-effector, and decide the movement and position of the rover. Additionally, a PID control was developed to enhance the robot's movement and control position along the cotton rows using feedback control after obtaining visual information. A PID was used to control the hydrostatic transmission, which rotated to the robot's tires. The articulation angle was controlled by using a proportional controller to ensure that the vehicle maintained its path within the cotton rows. The robot used an extended Kalman filter to fuse the sensors to localize the the rover's position while harvesting cotton bolls (Moore & Stouch, 2016). Therefore, the specific objectives of this study were to:

1. Design and develop a cotton harvesting robot using a center-articulated hydrostatic rover and 2D Cartesian manipulator; and
2. Perform preliminary and field experiments of the cotton harvesting rover

6.3 Materials and Methods

6.3.1 System Setup

The rover (Figures Figure 6.1 and Figure 6.6) used in this study was a four-wheel hydrostatic vehicle (West Texas Lee Corp., Lubbock, Texas) that was customized to be controlled remotely using an embedded system to navigate autonomously in an unstructured

agricultural environment (Fue et al., 2018b; Rains et al., 2014). The rover was 340 cm long and with front and back parts being 145 cm and 195 cm long, respectively. The rover was 212 cm wide, with a tire width of 30 cm. Each of the four tires had a radius of 30.48 cm and a circumference of 191.51 cm. The rover's front-axle and rear-axle were 91 cm from the center of the vehicle, and the ground clearance was 91 cm. To make turns, the rover articulated (bent) in the middle. A 2DOF manipulator was attached to the front of the rover. The manipulator consisted of two parts a horizontal axis, which was 70 cm, and a vertical axis, which was 195 cm. The manipulator was placed in the front of the rover with a 27 cm ground clearance. The rover had three electronic controllers: a master controller, navigation controller, and manipulation controller.

6.3.2 Master Controller System

The master controller (Figure 6.2) was installed with the embedded system (NVIDIA Jetson AGX Xavier development kit, Nvidia Corp., Santa Clara, CA, USA) connected to four sensors: two IMUs, a stereo camera, and an RTK-GPS. The two IMUs (Phidget Spatial Precision 3/3/3 High-Resolution model 1044_1B, Calgary, Alberta, Canada) were placed in front of the rover. The first IMU was placed 95 cm above the ground and 31 cm from the front of the vehicle (Figure 6.3). The second IMU was placed 132 cm above the ground and 46 cm from the front of the vehicle. The RTK-GNSS receiver (EMLID Reach R.S., Mountain View, California) with an integrated antenna was placed 246 cm above the ground and 30 cm from the front of the vehicle. The RTK correction signal was obtained using NTRIP signal (eGPS Solutions, Norcross, GA) with a mounting point within 2 miles of the test plot and downloaded to the Emlid through a Verizon modem (Inseego Jetpack MiFi 8800L, Verizon Wireless, New York, NY) hotspot and 802.11 wireless signals.

An RGB stereo camera (a ZED camera, Stereo labs Inc, San Francisco, CA, USA) was installed and used to acquire images. The ZED was 175 x 30 x 33 mm and had 4M pixel sensor per lens with large 2-micron pixels. The left and right sensors were 120 cm apart. The sensor was placed 220 cm above the ground in front of the vehicle facing down so it could image cotton bolls. The sensor took 720p resolution images at the rate of 60 frames per second. The ZED camera was chosen because of the need to work outdoors and provide depth data in real-time. The ZED camera provided a 3D rendering of the scene using the ZED software development kit (SDK), which was compatible with the robot operating system (ROS)and OpenCV library.

ROS provided all the services required for robot development, such as device drivers, visualizers, message-passing, package design, and management and hardware abstraction (Quigley et al., 2009). ROS was initiated remotely by using a client machine, and images were acquired directly from the ZED SDK, which took images and processed the 3D point clouds. Images were then parsed to the processing unit and analyzed using OpenCV (version 3.3.0) machine vision algorithms.

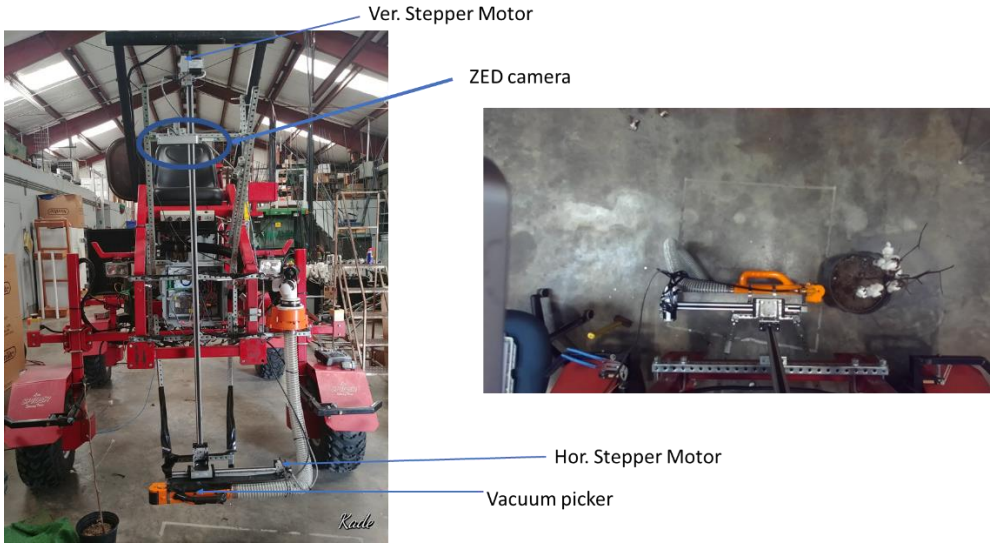


Figure 6.1. Left: Cotton Harvesting Robot view from the front. Right: Image from the ZED camera that shows cotton manipulator and potted cotton plant with bolls

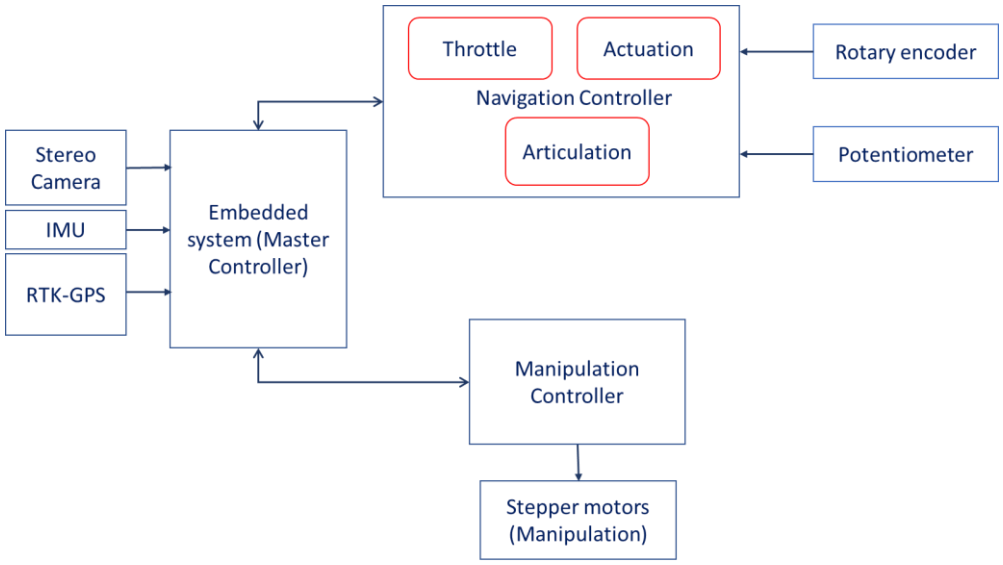


Figure 6.2. A contextual block diagram of the robotic system hardware

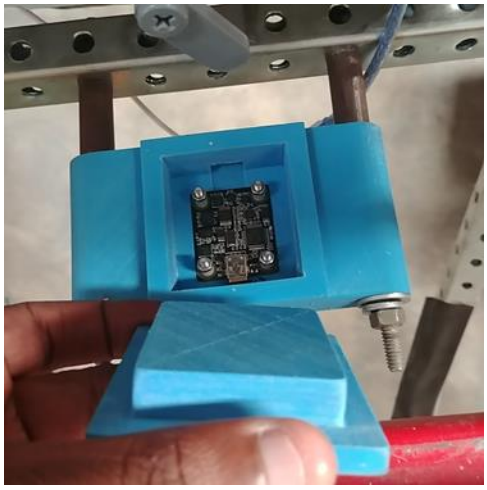


Figure 6.3. The first IMU Phidget Spatial Precision 3/3/3 High-Resolution model 1044_1B attached using 3D printed box to attach to the rover.

6.3.3 Rover Navigation Controller using Extended Kalman Filter for Robot Localization

The front two wheels of the rover were connected to rotary encoders (Koyo incremental (quadrature) TRDA-20R1N1024VD, Automationdirect.com, Atlanta, GA, USA). The encoders (Figure 6.4) were connected to the Rover navigation controller (Arduino Mega 2560, Arduino LLC) using four wires (signal A, signal B, power and ground) to register wheel rotation by detecting the rising edge of the generated square waves. A high precision potentiometer (Vishay Spectral Single Turn, Malvern, PA), reported the articulation angle of the vehicle by measuring the electric potential caused by the turn of the vehicle. The potentiometer was connected to the rover navigation controller.

Rover navigation controller received a signal from the embedded computer to control the rover articulation, actuation, and throttling. The sensors [RTK-GNSS, IMU, and encoders] were fused by using a ROS software package "robot_localization" (Moore & Stouch, 2016). The package "robot_localization" provided continuous nonlinear state estimation of the mobile vehicle in a 3D space by using signals obtained from the RTK-GPS, two IMUs, and wheel encoders. The fusion of the data was done by using state estimation node "ekf_localization_node" and "navstat_transform_node" (Moore & Stouch, 2016). The vehicle was configured using the procedure described in http://docs.ros.org. The IMUs published two ROS topics (imu_link1/data and imu_link2/data), the encoders published wheel odometry (/wheel_odom), and RTK-GNSS published /gps/fix signal. The topics were both sent to ekf_localization to get vehicle state estimation simultaneously (Moore & Stouch, 2016). The system used two IMUs, RTK-GNSS, and odometry of the encoders because Table 6.1 shows that this configuration provided excellent results (Moore & Stouch, 2016). Unlike the GNSS configuration shown in Table 6.1, our GNSS was an RTK-GNSS system that provided

centimeter-level accuracy, so configuration (odometry + two IMUs + one GNNS) was a proper configuration to adopt.

Table 6.1. Errors for five difference sensor configurations (Moore & Stouch, 2016)

Sensor Set	Loop Closure Error x, y (m)	Estimate Std. Dev. x, y (m)
Odometry (dead reckoning)	69.65, 160.33	593.09, 359.08
Odometry + one IMU	10.23, 47.09	5.25, 5.25
Odometry + two IMUs	12.90, 40.72	5.23, 5.24
Odometry + two IMUs + one GNSS	1.21, 0.26	0.64, 0.40
Odometry + two IMUs + two GNSSs	0.79, 0.58	0.54, 0.34

The hydraulic motors mounted to the rover wheels were controlled using a servo to the engine throttle and a servo to the variable-displacement pump swashplate arm. Each front tire was connected to a rotary encoder to provide feedback on the movement of the rover along the crop rows. The throttle was connected to the onboard Kohler Command 20HP engine (CH20S, Kohler Co, Wisconsin, USA) with a maximum of 2500 RPM and driving an axial-piston variable rate pump (OilGear, Milwaukee, WI, USA) with a maximum displacement of 14.1 cc/rev. The OilGear pump displacement was controlled by a swashplate for directing the forward and rearward movement of the rover. The swashplate angle was controlled by the rover controller that determined the placement of the linear electric servo (Robotzone HDA4, Servocity, Winfield, KS). Left and right articulation were controlled by using linear hydraulic actuators connected to a 4-port 3-way open-center directional control valve (DCV) powered by a 0.45

cc/rev fixed displacement pump (Bucher Hydraulics, Klettgau-Griessen, Germany) in tandem with the Oilgear pump. The rover could turn a maximum of 45 degrees with a wheelbase of 190 cm.

Figure 6.4. Encoder installed on the front tire of the rover to provide input pulses which indicates how far the rover has moved from one point to another

6.3.4 Manipulation Controller

The robot manipulation controller received a 4-byte digital signal from the Jetson Xavier embedded computer. The signal provided the number of steps and direction of the manipulator (Up, Down, back and forth). Then, the controller sent the signal to the micro-stepping drive (Surestep STP-DRV-6575 micro-stepping drive, AutomationDirect, Cumming, Georgia), which in turn sent it to the appropriate stepper signal for action. Manipulator controller was connected to the Jetson using a USB 3.0 hub shared by the ZED camera.

The robotic manipulator was designed to work within a two-dimensional cartesian coordinate system (Figure 6.5). Each arm of the manipulator was moved using two 2-phase stepper motors (MS048HT2 and MS200HT2, ISEL Germany AG, Eichenzell, Germany). The MS048HT2 model stepper motor was installed to drive the horizontal linear axis arm (60 cm

long), and the MS200HT2 to drive the vertical linear axis arm (190 cm long). The connecting plates and mounting brackets used a toothed belt that was driven by the stepper motor. Two micro-stepping drives (Surestep STP-DRV-6575 micro-stepping drive, Automation Direct, Cumming, Georgia) were installed to provide accurate position and speed control with a smooth motion. The micro-stepping drive DIP switch that controlled the motors was set to run a step pulse at 2MHz and 400 steps per revolution. This setting provided smooth motion for the manipulator. The step angle was 1.8^0. The arms of the manipulator were connected in a vertical crossbench Cartesian configuration (Figures Figure 6.5 and Figure 6.6). The sliding toothed belt drive (Lez 1, ISEL Germany AG, Eichenzell, Germany) was used to move the end effector. The toothed belt had 3 mm intervals and was 9 mm wide. The error of the toothed belt was +/- 0.2 mm per 3 mm interval.

A wet/dry vacuum was installed on the rover to help pick and transport the picked cotton bolls—the vacuum connected to the end-effector via a flexible 90-cm flexible plastic hose. Cotton bolls were vacuumed into the hose, which was placed close to the cotton bolls (Figure 6.7). The end-effector had a rotating brush roll that grabbed and pulled cotton bolls through a combination of vibration and sweeping. The brush roll was powered by a 12DC motor (Figure 6.7). The cotton bolls were grabbed and passed through a flexible hose to a porous impeller mounted with the suction opening from the vacuum. The porous impeller was rotated using the 12 VDC motor, and the cotton bolls were dropped into a bag.

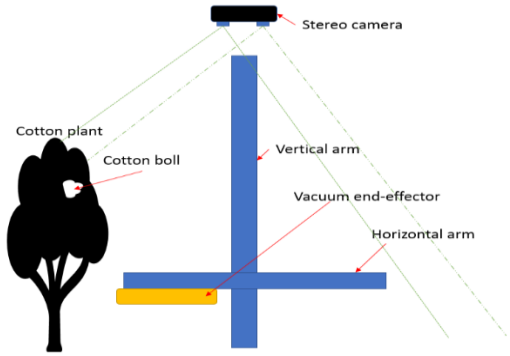


Figure 6.5. Robotic cartesian arm contextual diagram

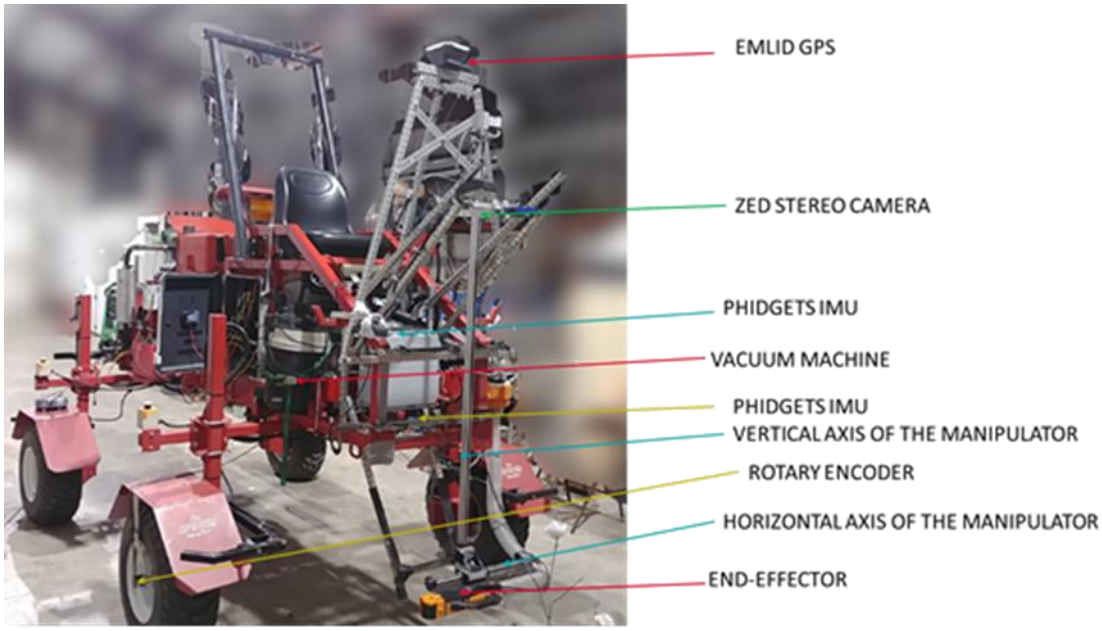


Figure 6.6. The robotic arm, vacuum, and sensors mounted on the red rover

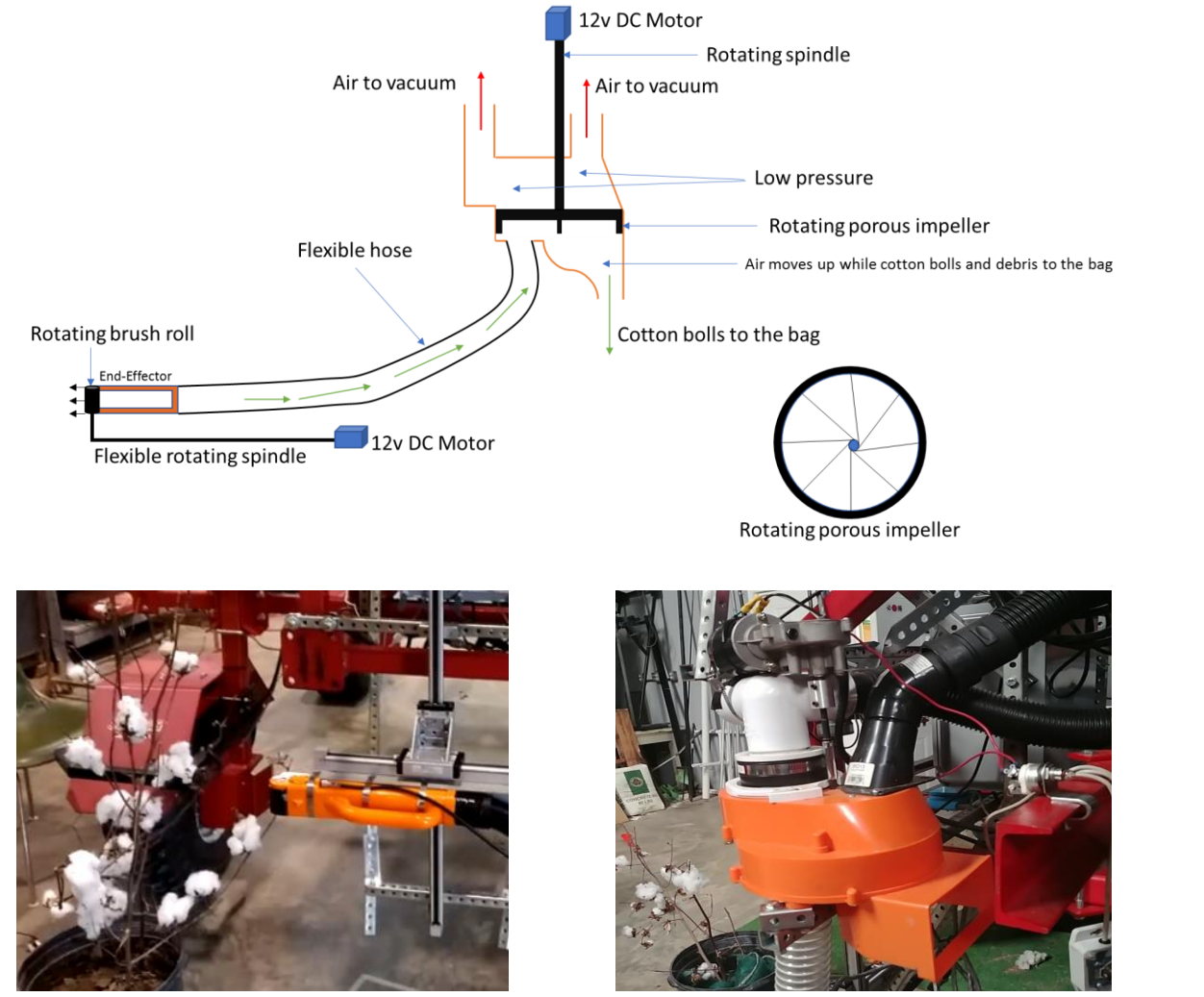


Figure 6.7. The end-effector was moved to the cotton bolls using the Cartesian arm system, and then a rotating brush roll grabbed the bolls into the end-effector. An onboard vacuum transported the bolls to the rotating porous impeller where the bolls were dropped into a collection bag.

6.3.5 Boll Detection Algorithm Improvements using Tiny YOLOv3

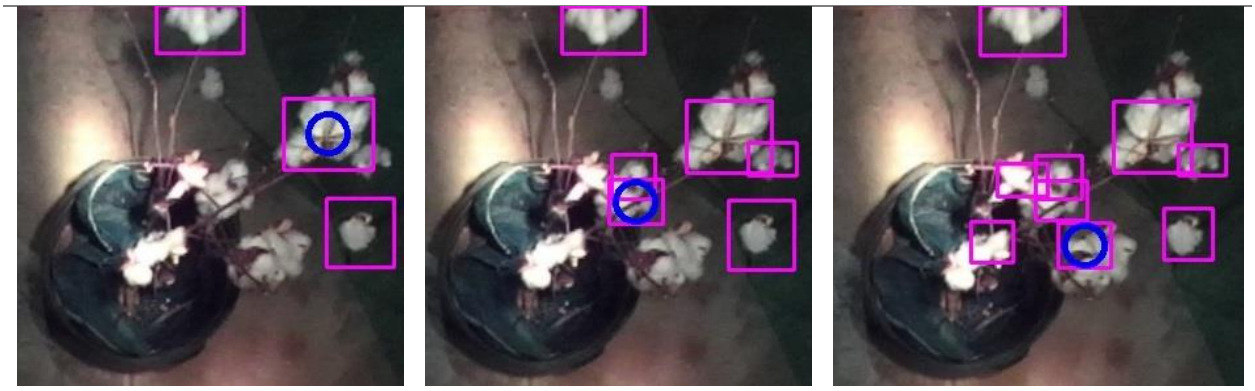
Cotton boll images were used to train a tiny YOLOv3 deep neural network model (Redmon & Farhadi, 2018). The tiny YOLOv3 is a simplified version of YOLOv3, which has seven convolutional layers (Redmon & Farhadi, 2018). The tiny YOLOv3 is optimized for speed with reduced accuracy compared to YOLOv3 (Redmon & Farhadi, 2018). The images were

augmented 26 times using various techniques which are average blurring, bilateral blurring, blurring, cropping, dropout, elastic deformation, equalize histogram, flipping, gamma correction, Gaussian noise, Gaussian blurring, median blurring, raise blue channel, raise green channel, raise hue, raise red channel, raise saturation, raise value, resizing, rotating, salt and pepper, sharpen, shift channels, shearing, and translation. Augmentation was accomplished using the CLoDSA library. CLoDSA is an open-source image augmentation library for object classification, localization, detection, semantic segmentation, and instance segmentation (*https://github.com/joheras/CLoDSA*). We collected and labeled 2085 images from the Internet and camera images. The new augmented and labeled image dataset consisted of 56,295 images.

The dataset was used to train the tiny YOLOv3 model using a Lambda Server (Intel Core i9-9960X (16 Cores, 3.10 GHz) with two GPUs RTX 2080 Ti Blowers with NVLink and Memory of 128 GB, Lambda Computers, San Francisco, CA 94107). The learning rate was set to 0.001 and maximum iterations to 2000 (Redmon & Farhadi, 2018). The batch size was 32. The model was trained for 4 hours, and then, it was then frozen, a process of combining graph and trained weights, and transported to the rovers' embedded system.

The ZED Library, Zed-yolo in Github (*https://github.com/stereolabs/zed-yolo*), the free, open-source library package provided by the manufacturer of the ZED camera (Stereolabs Labs) was used to connect ZED camera images to tiny YOLOv3 model to perform object detection. The library used image bilinear interpolation to convert ZED SDK images to OpenCV images so that it could perform detection tasks. The bilinear interpolation was compared with the nearest-neighbor and no interpolation to evaluate object detection. The results (Figure 6.8) show that the model performed well when no interpolation was applied (Figure 6.8). The images in Figure 6.8 show that the bilinear interpolation detected only three bolls, the nearest-neighbor interpolation

detected five bolls, while with no interpolation, the system detected eight bolls. Therefore, the boll detection algorithm improved its accuracy when no interpolation was performed. The library code was modified to avoid image interpolation and implemented in this study. The algorithm was able to perform well with illuminated images of undefoliated cotton (Figure 6.9), where the right image in Figure 6.9 shows cotton boll detection in natural direct sunlight.



(a) *(b)* *(c)*

Figure 6.8. Detection of cotton bolls using bilinear transformation [8a] detected 3 bolls, Nearest-neighbor interpolation 8b detected 5 bolls, and [8c] without interpolation detected 8 bolls. The pink boxes are bounding boxes of the detected bolls, while blue represents the nearest boll to be picked.

Figure 6.9. Boll detection using tiny YOLOv3 and ZED camera. Left image: cloudy day, Right image: sunny day.

6.3.6 End-effector Detection using Color Segmentation

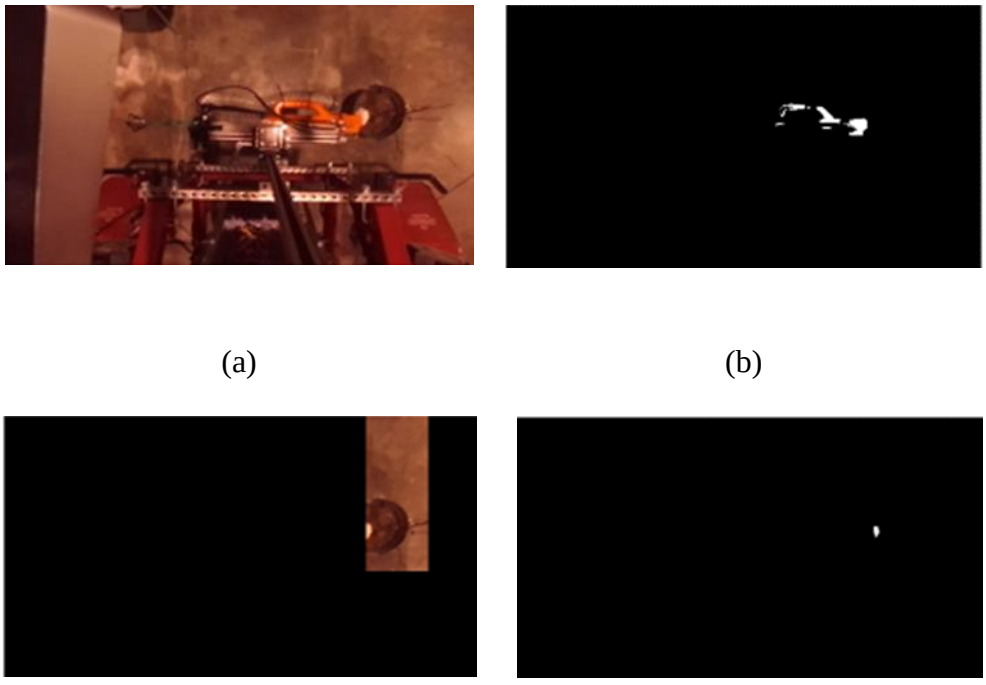
Each image frame was acquired and analyzed to detect the orange end-effector using a 4-step process (1. depth processing, 2. color segmentation, 3. feature extraction, and 4. depth matching with features). These steps were handled by the graphics optimized rugged development kit (NVIDIA Jetson Xavier) to achieve improved performance as image calculations require massive graphics computing resources like NVIDIA CUDA cores. The ZED SDK acquired and processed the images to get depth disparity and rectified images for both lenses. In this case, the ZED SDK was provided with 60 fps of 720p quality images and 3D point clouds.

The images acquired (Figure 6.10a) were first analyzed for arm movement. Since the arm was orange in color, the threshold color was determined so that the arm could be segmented from the rest of the image(Figure 6.10b). The cotton boll and end-effector segmentation task involved the following four steps(Gong & Sakauchi, 1995):

1. Grab an image
2. Using the RGB color threshold, separate each RGB component of the image. The end-effector, which is orange in color, can be masked. The threshold was (Red from 200 to 255, Green from 0 to 255 and Blue from 0 to 60).
3. Subtract the image background from the original image.
4. Remove all the regions where the contours were less than value M. Value M was determined by estimating the number of pixels defining the smallest boll.

Feature extraction was performed by finding contours of the consecutive points which have the same intensity and were clustered. The cotton boll was then detected using tiny YOLOv3 after segmenting the contour of the arm (Figure 6.10c and Figure 6.10d).

After obtaining the contours for the end-effector, the centroid of the contour was calculated. All the depths were calculated for bolls by matching YOLOv3 detected bolls and end-effector coordinates with 3D point clouds or depth disparity maps, as described in algorithm 1 (Table 6.2). The cotton bolls coordinates (x,y,z) and robot manipulator coordinates (x_0,y_0,z_0) were obtained and used by the robot for picking decisions.



(a) (b)

(c) (d)

Figure 6.10. Color segmentation to localize the end-effector of the manipulator. (a) is the RGB image of the manipulator, (b) is the mask image of the end effector, (c) is the masking of all over image parts leaving only an area that can be reached by the end effector and hence it might contain bolls to pick. (d) is masking of the cotton bolls.

6.3.7 Inverse Kinematics of the Robot

The robot was designed to operate in 3D space and for a 2D cartesian manipulator to pick a cotton boll as the end-effector reached the bolls. Figure 6.11 shows the inverse kinematics of

the robot. The red cube represents the front of the rover, while horizontal axis arm and vertical axis arm make up the rover's manipulator.

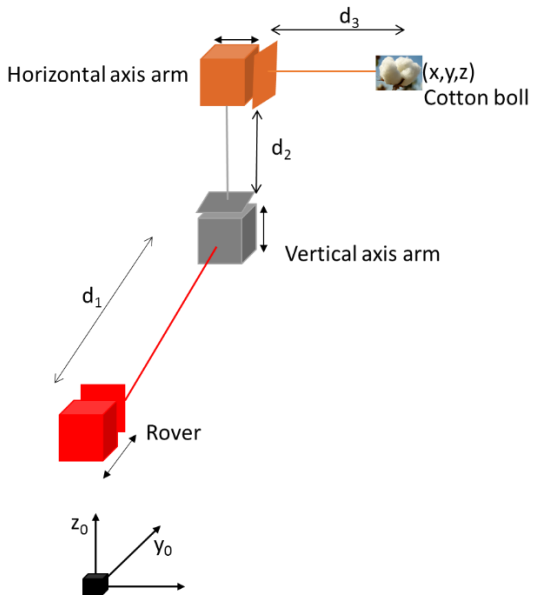
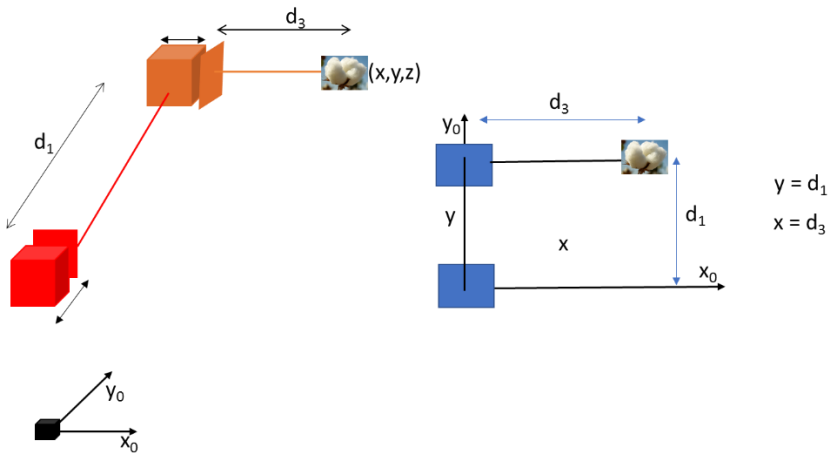


Figure 6.11. The inverse kinematics of the cotton harvesting robot

The inverse kinematics (Figure 6.11 Figure 6.12) was obtained by getting the values of the point (x,y,z), which is boll position from the origin of the rover (x_0,y_0,z_0). The robot could move distances d_1, d_2, and d_3 to pick the cotton boll at the point (x,y,z).



Top View

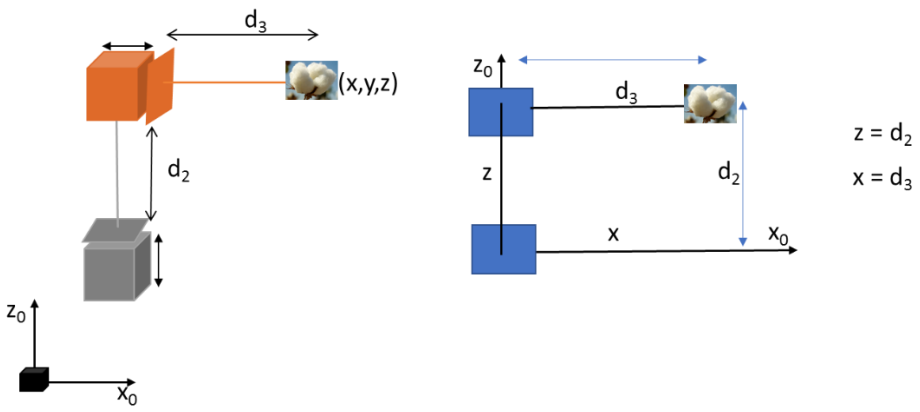


Figure 6.12. The inverse kinematics of the rover in detail (It is given by the finding the distances d1, d2, and d3 of the rover using depth and coordinates of the boll (x,y,z) found by the ZED stereo camera)

6.3.8 Depth and Coordinates of Cotton Bolls and Manipulator

After matching the depths and contours of the Cartesian arm and cotton bolls, each reading of the arm and boll position were logged. Then, by using the tip of the end-effector (Figure 10c), the system obtained the image coordinates of the front part of the end-effector. Then, using the centroids of each boll, the system calculated the real-world coordinates (W) of the bolls from the image coordinates obtained (I) by using image geometry. The stereo camera was calibrated using the ZED Calibration tool of the ZED SDK to obtain the camera transformation matrix (Equation 6.2) parameters. The procedures to calibrate the ZED camera were learned from their website (https://www.stereolabs.com/zed/). The camera matrix consists of f_x and f_y (the focal length in pixels), C_x and C_y (the optical center coordinates in pixels), and k_1 and k_2 (distortion parameters). The real-world coordinates of a cotton boll, W_x and W_y (Equations 6.4 and 6.5) can be obtained if the algorithm was provided with the value of I_x and I_y which was the coordinate of the centroid of the front part of the end-effector. Alternatively, by finding the inverse of the camera matrix and multiplying the vector image (I_x and I_y), the world

coordinates (W_x and W_y) could be obtained. C_x, f_x, C_y, and f_y were found by calibrating the camera, while W_z was determined from the 3D point cloud provided by the ZED SDK. The parameter values of the calibrated camera used were:

$C_x = 685.286$

$C_y = 361.248$

$f_x = 699.936$ (6.1)

$f_y = 699.936$

$$C = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (6.2)$$

$$\begin{bmatrix} I_x \\ I_y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} W_x \\ W_y \\ W_z \\ 1 \end{bmatrix} \quad (6.3)$$

$$W_x = (I_x + C_x) * \left(\frac{W_z}{f_x}\right) \quad (5.4)$$

$$W_y = (I_y + C_y) * \left(\frac{W_z}{f_y}\right) \quad (5.5)$$

where

fx, and **fy** = the focal length in pixels,

C**x** and **Cy** = the optical center coordinates in pixels,

$\mathbf{W_x}$ and $\mathbf{W_y}$ = the real-world coordinates, and

$\mathbf{I_x}$ and $\mathbf{I_y}$ = the coordinate of the centroid of the front part of the horizontal arm.

Object depth in an image (W_z) was obtained from the 3D point clouds. For each boll and end-effector location, the distance points (d_x, d_y, d_z) were provided by the ZED SDK. It was recommended to use the 3D point cloud instead of the depth map when measuring depth distance. The Euclidean distance (Equation 6.6) is the calculated distance of an object (end-effector or bolls) relative to the left lens of the camera.

$$W_z = \sqrt{{d_x}^2 + {d_y}^2 + {d_z}^2} \quad (6.6)$$

Depth distance ($\mathbf{W_z}$) should be greater than zero and less than the distance of the camera to the lowest position of the end-effector to avoid the rover attempting to harvest unreachable bolls. Also, the horizontal distance ($\mathbf{W_x}$) of the boll from the center of the camera should not exceed the length of the horizontal axis arm. After obtaining such measurements, the system executed other tasks like controlling the arm or moving the rover. For the machine to be able to execute each task independently and in coordination with the other tasks, the finite state machine was developed to manage task-based requests.

6.3.9 Finite State Machine

Robot tasks and actions were categorized as states, and state "transitions" were modeled in a task-level architecture to create the rover actions required to harvest cotton bolls. This approach provided a maintainable and modular code. Using an open-source ROS library known as SMACH (http://wiki.ros.org/smach), the tasks were smoothly implemented to build complex behavior (Figure 6.13).

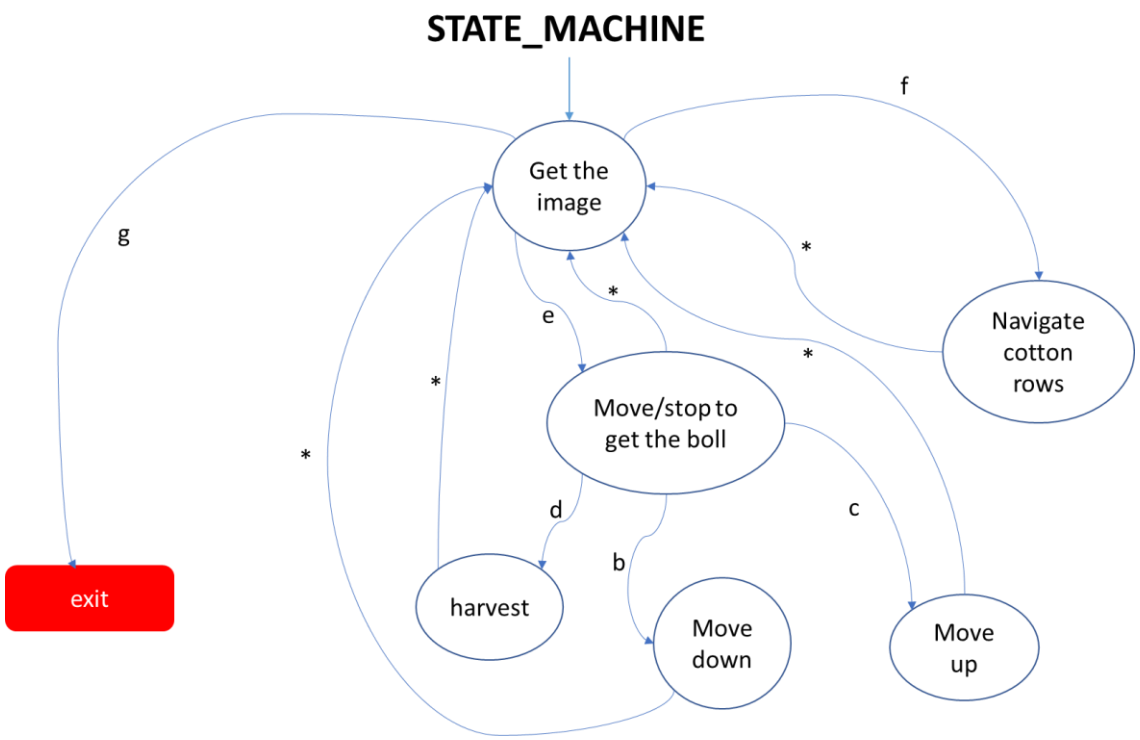


Figure 6.13. Finite State Machine Diagram of the rover states and transitions. * means a return instruction to get a new image.

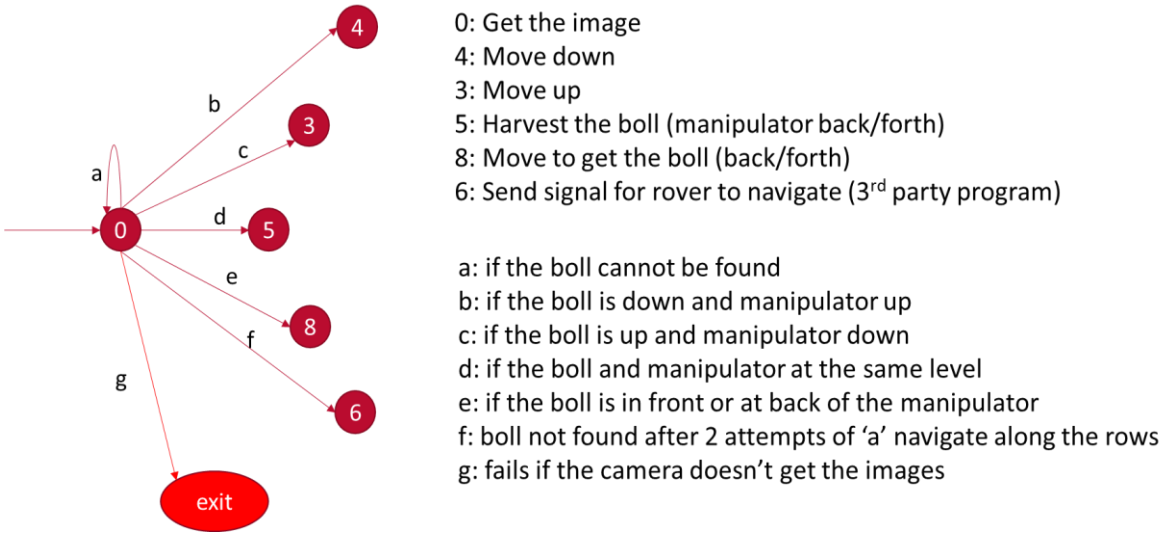


Figure 6.14. Linear transformation of the Finite State Machine Diagram (Figure 6.13) of the rover states and transitions

The state machine had six necessary states and seven transitions (Figure 6.13 and Figure 6.14). After every state, the system reverted to state 0 (get the image) and searched for cotton bolls. If a cotton boll was found, the system would calculate the distance of the boll from the manipulator in 3-dimensional space (X, Y, and Z) as described in the inverse kinematics section. If the boll was lined up horizontally, then the system would get the manipulator to move up or down relative to the position of the manipulator to the boll. If the boll was at the same level, then the system would harvest it. If the boll was in front or back, the system would send a signal for the rover to move forward or back using PID control. If the system failed to see any bolls, the rover proceeded to pass over the cotton rows. Table 6.2 describes the detection of the bolls and actions taken by the state machine algorithm to accommodate the rover transition of the tasks.

Table 6.2. Algorithm describing the detection of cotton bolls

Algorithm 1: Algorithm describing the detection of cotton bolls

Input: current video frame,

Output: Decision to move manipulator or rover [Ci]

1. Get end-effector position Xm, Ym and Zm
2. Get prediction results of the YOLO model
3. Get the list of centroids for each boll detected [Oj]
4. FOR EACH Oj in [Oj]
    a. Boll_depth <- calculate the closest distance of the centroid using left lens point cloud
5. END FOR
6. Get the closest boll position
7. Calculate the position of the boll Xb, Yb, and Zb

8. Find the difference between (Xm and Xb), (Ym and Yb) and (Zm and Zb)
9. IF (Yb > Ym) transition e (move forward)
10. IF (Yb < Ym) transition e (move backward)
11. IF (Yb = Ym and Zm > Zb) transition b (move the arm up)
12. IF (Yb = Ym and Zm < Zb) transition c (move the arm down)
13. IF (Yb = Ym and Zm = Zb and Xm – Xb < 37 cm) transition d (pick the boll) ##the manipulator can only cover bolls close from 0 to 37 cm from vertical arm
14. Return the state decision [Ci]

6.3.10Calibration of the Manipulator

The manipulator was calibrated for its horizontal and vertical movements. The equation was obtained by first moving the arm to the furthest location away from the ZED camera. The distance of the arm from the camera was then recorded and was continually recorded as the arm was moved by each step of the stepper drive until it was closest to the camera (Figure 6.15). The equation obtained by fitting the points was:

$$\text{motor steps} = -410.7 * (\text{distance}) + 483.29 \quad (5.7)$$

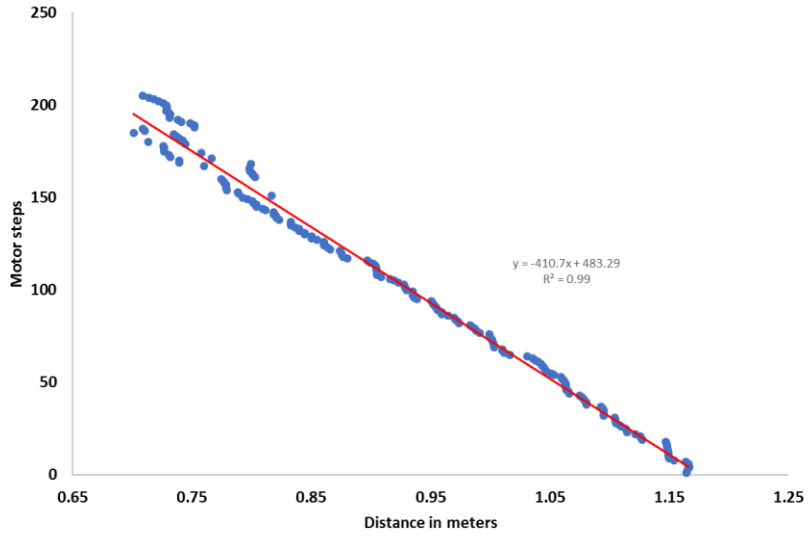


Figure 6.15. Calibration of the distance against steps of the stepper motor

6.3.11 Rover Movement Controller

The rover movement controller ran an adaptive PID (Figure 6.13) to control the rover forward and rearward movement. The position of the rover and the target position were published from the master, and the Arduino clients subscribed to the topic accordingly. The throttle was set at a constant maximum RPM to maintain constant power to the hydraulic systems. A topic that subscribed to this message was developed in the rover microcontroller. Articulation was done after the rover controller received a topic that published the instruction for the rover to turn or go straight. For this study, the rover only moved straight. The master controller published an articulation message at the rate of 50Hz to the rover controller, which had subscribed to the topic. The rover movement controller published the gains of the adaptive PID controller together with the position of the rover relative to the encoder pulses. The master controller subscribed to the messages so as to provide an accurate position of the rover, which was used to pass the signal to the arm controller for boll picking.

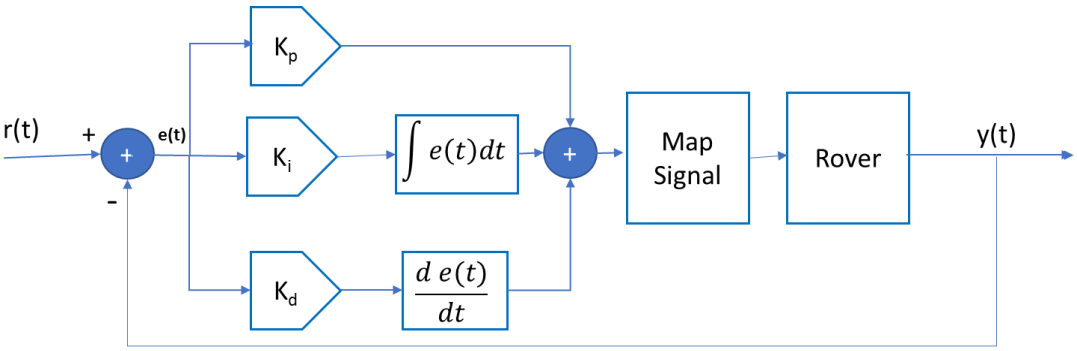


Figure 6.16. The PID controller implemented in a rover controller to achieve accurate target position

The control function (Equation 6) of the controller above is presented as;

$$u(t) = P + I + D$$

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} \quad (6.8)$$

where,

constants K_p, K_i, and K_d, are given as proportional gain, integral gain, and differential gain respectively,

r(t) is a set point which is the number of pulses required to reach a certain distance, and

y(t) is the measured number of pulses read by the rotary encoder.

The system minimized error e(t), which was given e(t) = r(t) – y(t). K_p, K_i, and K_d denoted the coefficients for the proportional, integral, and derivative gains, respectively. Tuning was done by first identifying the deadband of the hydraulic swashplate arm for the variable displacement pump. Up to a particular movement angle of the swashplate arm, there was insufficient fluid flow and pressure to move the rover. For operation, the rover movement controller sent a servo signal to the linear actuator (Figure 6.17), to proportionally increase fluid flow to the hydraulic wheel motors. The linear actuator pushed the swashplate to a certain angle and was directly controlled using a rover navigation controller. The linear actuator extends from

0 to 20 cm after receiving an analog servo signal. Retracting and extending the linear actuator changed the position of the swashplate arm, which in turn controlled the speed and direction of the hydraulic fluid, which provided the motion capability of the rover. The angle of the swashplate can be changed by sending a servo PWM signal that ranged from 65 to 120 using the Arduino servo library. When a 90 PWM signal was sent to the rover controller, the rover stops since the swashplate was positioned in its neutral position, and the fluid flow was zero. If the PWM signal was decreased slowly from 90 to 65, the rover moved in reverse motion, while if it was increased from 90 to 120, the rover moved forward. It meant the 65 PWM signal provided maximum speed in reverse motion, while 120 PWM signal provided maximum speed forward. The rotary encoder installed on the wheel of the rover sends back input pulses to the PID, which measures how far the rover has moved. However, the system had a large deadband from a PWM signal of 80 to 98, which means the angle change was not enough to make the rover move. The deadband (Figure 6.18) was the signal angle sent from the rover Arduino microcontroller to the linear actuator servo that cannot make the rover move either forward or reverse. Consequently, when the rover missed the target, it became challenging to get close to the target as the PID can either be a direct relationship or reverse relationship between output (actuator signal) and input (encoder pulses).

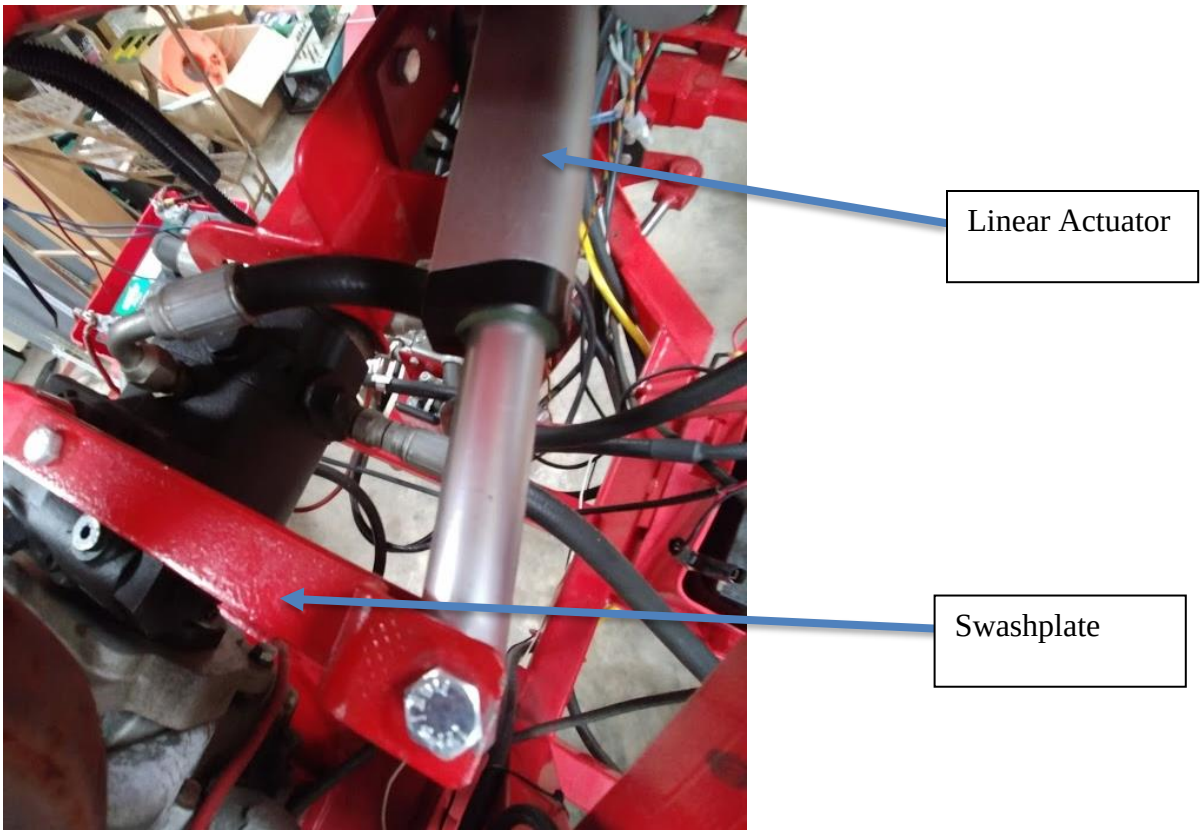


Figure 6.17. The linear actuator moving the swashplate arm to determine an angle for movement of the rover (forward when extending or rearward when retracting)

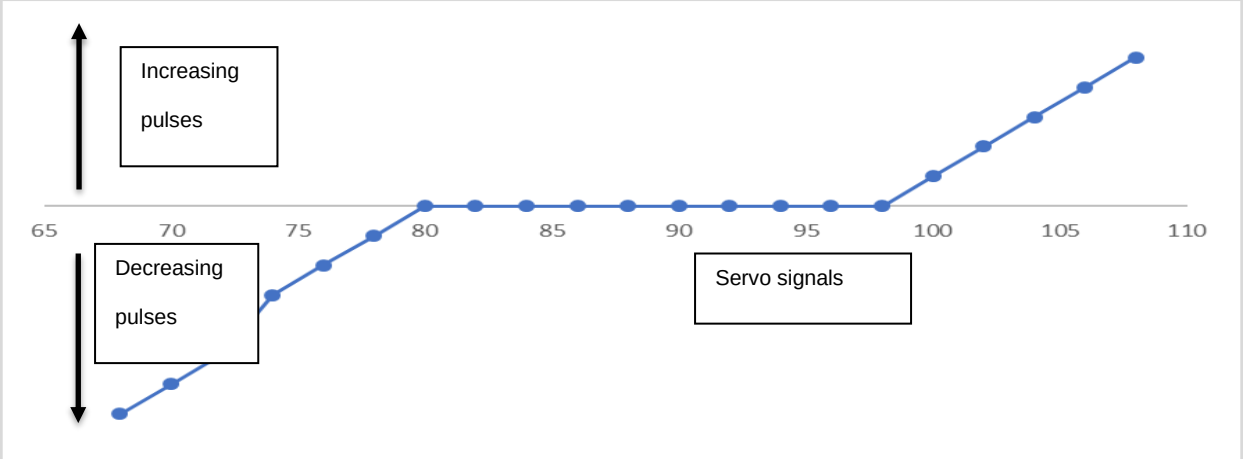


Figure 6.18. Preliminary data shows deadband between 80 to 98 of the actuator PWM signal (actuator should move forward when the PWM signal to the linear actuator is increased from 90 to 120 and move reverse if the PWM signal is decreased from 90 to 65)

In order to remove the deadband, the system was redesigned such that it gave the output u(x) from -100 to 100 (Equation 6.9). Then, the signal was mapped to the correct settings of the rover. The actual servo signal was set to move (extend) from 98 to 108 by mapping positive output values (0 to 100) of u(x) while moving back (retract) from 70 to 80 for negatives output values (-100 to 0) while zero was set to be 90 signal.

$$u(x) = \begin{cases} 10 * (x - 98) \; if \; 98 \leq x \leq 108 \\ 10 * (x - 80) \; if \; 70 \leq x \leq 80 \end{cases} \quad (6.9)$$

The manual tuning technique was used to tune the PID controller. It was done by increasing K_p until the rover oscillated with neutral stability while setting K_i and K_d values to zero. Then, K_i was increased until the rover oscillated around the setpoint. After that, K_d was increased until the system was settling at the given setpoint quickly. The PID gains were obtained; $K_p = 0.5$, $K_i = 0.15$ and $K_d = 1$. Later, the navigation controller was recalibrated to determine if the performance could be improved. The PID gains that create a small overshoot response but aggressively moved the rover forward were obtained; Kp = 0.7, Ki = 0.04, and Kd = 5.0. The rqt_graph library (http://wiki.ros.org/rqt_graph), which is the ROS computation visualizing graph, was used to study the pulses. Each of the encoder pulses was equivalent to 1.8 mm distance. In Figure 6.19, the master controller sent a servo signal to the rover controller to instruct the system to move from the 500th pulse position to the 2000th pulse position and then go back to the initial position. The rover movement settled after 1250 ms (Figure 6.19). The system performed the same way going forward or backward. The overshoot was approximately 18 cm.

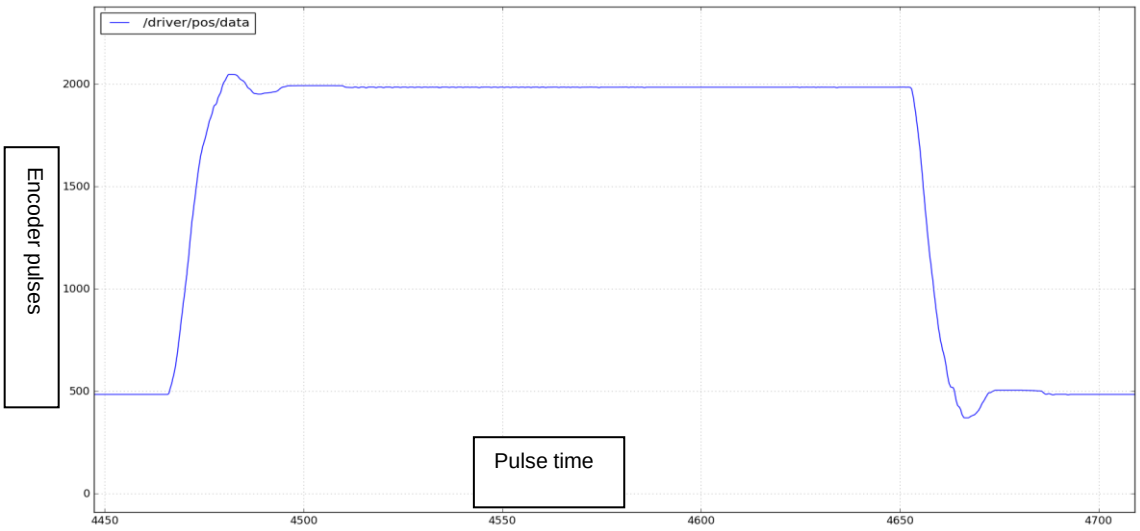


Figure 6.19. The PID control graph of the rover. The signal was sent from the Master controller to the rover navigation controller to move the vehicle from point 500th position to 2000th position and then back to 500th position. Each encoder pulse is equivalent to 1.8mm. Each pulse time unit is equivalent to 50 ms. This graph is obtained using rqt_graph by subscribing to a published topic driver/pos, which provides position feedback from the encoder. The rqt_graph provides a GUI plugin for visualizing the ROS computation graph.

6.3.12 Proportional control of the articulation angle

The rover turned to the target articulation angle γ by using proportional control. The current articulation angle γ_k and required target angle γ_{k+1} was used to find the error used to control the signal to turn the rover. The gain Kp was set to 1. The articulation angle was controlled by the hydraulic linear actuators, which were connected to two relays. The two relays were used to connect the left and right control signals from the navigation controller to a 12v power source to move the hydraulic directional control valve spool. Hydraulic cylinders in series were used to push and pull the front and rear halves of the rover to create a left or right turn. If

the rover turned left, the left actuator retracted while the right actuator expanded until a desired left articulation angle was achieved. If the rover turned right, the right actuator retracted while the left actuator expanded until a desired right articulation angle was achieved. The angle was reported by a calibrated high precision potentiometer.

6.3.13 Preliminary Experimental Setup

The navigation and manipulation of the system were tested as one unit of the robot. It is essential to determine the performance of the whole robot in navigating to cover 3D space and picking of the bolls.

Five experiments were set up at the University of Georgia (UGA) Tifton campus grounds (N Entomology Dr, Tifton, GA, 31793). The experiments were undertaken on 29th and 30th May 2019. Six cotton plants were placed 30 cm apart, three bolls per plant. The slope of the ground surface in the direction of the forwarding movement was 0.2513^{0}. The rover was driven over the plants, and the manipulator moved into a position to pick the bolls from the plants (Figure 6.21 Figure 6.20). The camera was 220 m above the ground (l). The first boll detected by the system was distance “m” from the camera. The camera also checked the distance of the arm from the camera, and then, the manipulator controller sent a signal to move the arm to harvest the first boll. Subsequent boll picking was repeatedly accomplished by getting the distance of the current end-effector position to the next boll position in the camera images.

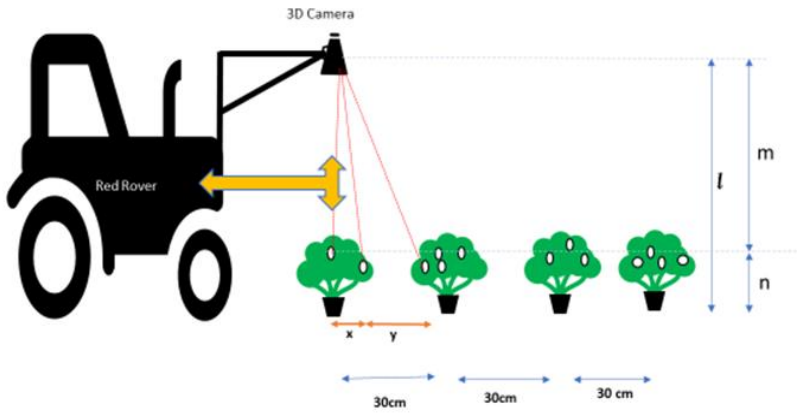


Figure 6.20. System testing was done by putting the six potted defoliated plants in front of the system to collect preliminary performance data.

Figure 6.21. The experiment was set up at UGA grounds to test the robot on picking the bolls on a simulated environment consisting of six potted cotton plants.

The results of the five experiments with six plants with three bolls in each were collected and analyzed using the robot performance metrics mentioned by (Bechar & Vigneault, 2017).

6.3.14 Field Experiment

The field experiment in undefoliated cotton was conducted at the Horticulture Hill Farm (31.472985N, 83.531228W) near Bunny Run Road in Tifton, Georgia, after establishing the

calibrated parameters of the rover. The field (Figure 6.13) was planted on 06/19/2019 using a tractor (Massey-Fergurson MF2635 tractor, AGCO, Duluth, GA) and a 2-row planter (Monosem planter, Monosem Inc, Edwardsville, KS). The cotton seeds (Delta DP1851B3XF, Delta & Pine Land Company of Mississippi, Scott, MS) were planted every two rows, and two rows skipped. The rows were 36-inch (91.44 cm) wide, and the seed spacing was 4-inch (10.16 cm). The field experiment was done after finishing the preliminary experiments. Two tests (5.3 m) for picking the cotton bolls were conducted on 22nd November 2019 and 2nd December 2019. The slope of the ground surface in the direction of the forwarding movement was 2.7668^{0}. It is steeper than preliminary experiment field.

We measured Action Success Ratio (ASR) which is the ratio of the number of the picked bolls to a number of all cotton bolls present and Manipulator Reaching Ratio (MRR), which is the ratio of the bolls seen and attempted to be picked to the number of all bolls present.

6.4 Results and Discussions

6.4.1 Simulated Harvesting of Potted and Defoliated Cotton

The results (Table 6.3) of the test with defoliated cotton in pots were obtained by counting the cotton bolls that the robot was able to pick and the time it took to collect the boll. Also, images taken by the camera were checked to determine if the system could detect cotton bolls using only color segmentation. OV is the average velocity measured during a mission under real-time. P.R. is the number of cotton bolls picked per time unit. C.T. is the average time required to complete one cotton-picking action. ASR is the ratio of success in the action of picking the bolls. D.P. is the ratio of the number of appropriate detections (True positives + True negatives) over the sum of all boll detection attempts made by the system.

Table 6.3. The collected data for the experiments done to validate the performance of the system in the simulated environment. The parameters determined are; Operation Velocity (OV) under real-time conditions (cm s-1), Production Rate (PR) (bolls s-1), Cycle Time (CT) (s), Action Success Ratio (ASR) (%) and Detection Performance (DP) (%). .

Cotton bolls	**Detected**	**Reached**	**Picked**	**Time (sec)**	**OV (cm s^{-1})**	**PR (boll $s\ s^{-1}$)**	**CT (sec)**	**ASR (%)**	**DP (%)**	**MMR (%)**
18	17	16	15	214	1.12	0.07	14.27	83.33	94.44	88.89
18	18	17	16	343	0.70	0.05	21.43	88.89	100.00	94.44
18	18	17	15	219	1.10	0.07	14.60	83.33	100.00	94.44
18	17	17	17	323	0.74	0.05	19.00	94.44	94.44	94.44
18	18	18	17	285	0.84	0.06	16.76	94.44	100.00	100.00
Average	17.4	17	16	276.8	0.87	0.06	17.3	88.88	96.67	94.44

The system picked a boll at an average of 17.3 seconds at a total distance of 2.4 meters. The experiment showed that our rover design is a viable solution for cotton harvesting. However, the system was observed to attempt (MRR) twice or more to pick an average of 17 of the 18 bolls. The manipulator missed an average of two bolls because of the overshoot by the PID controller that reduced the production rate.

6.4.2 Field Picking of Cotton

Two experiments (Table 6.4) to investigate the effectiveness of the rover to pick the bolls were successfully conducted in a field with undefoliated cotton. In the first test, the rover picked 67 bolls and left behind 17 bolls for an Action Success Ratio (ASR) of 80%. The rover was able to reach (Manipulator Reaching ratio (MRR)) 94% of the bolls (79 bolls). The distance covered by the rover was 5.3 meters. For the second test, the robot picked 89 and left behind 26 bolls, for an ASR of 77%. The MRR was 96%, and the distance covered was 5.3 meters. The average ASR was 78.5%. The average MRR was 95%, while C.T. and D.P. were 38.35 seconds and 0.03 bolls per second, respectively. The CT for field testing was over twice the cycle time in the defoliated and potted cotton plant test, while D.P. was about half the performance in defoliated and potted cotton plants. MRR was comparable in both field and simulated testing (95.3% and 94.4%,

respectively). This performance indicated the system was having trouble picking bolls in the real field due to the unstructured topography of the field and the leaf and stem obstruction of the bolls. Also, in the simulated environment, the bolls were loosely attached to the plant; hence it was easier to remove from the branch in a single attempt.

Table 6.4. The collected data for the experiments done to validate the performance of the system in the real field environment. The parameters determined are; Operation Velocity (OV) under real-time conditions (cm s-1), Production Rate (PR) (bolls s-1), Cycle Time (CT) (s), Action Success Ratio (ASR) (%), Manipulator Reaching ratio (MRR) and Detection Performance (DP) (%).

Cotton bolls	Detected	Reached	Picked	Time (sec)	OV	PR	CT	ASR (%)	DP (%)	MRR (%)
84	83	79	67	2700	0.20	0.02	40.30	79.76	98.81	94.05
115	112	111	89	3240	0.16	0.03	36.40	77.39	97.39	96.52
Average	97.5	95	78	2970	0.18	0.03	38.35	78.58	98.10	95.28

6.5 Conclusion

The preliminary design and performance of a prototype cotton harvesting rover were reported. The system was optimized to use visual-based controls to pick cotton bolls in undefoliated cotton. The system used SMACH, an ROS-independent finite state machine library that provides task-level capabilities, to design a robotic architecture to control the rover's behavior. The cotton harvesting system consisted of a hydrostatic, center-articulated rover that provided mobility through the field and a 2-D manipulator with a vacuum and rotating mechanism for picking cotton bolls and transporting them to a collection bag. The system achieved a picking performance of 17.3 seconds per boll in simulated field conditions and 38 seconds per boll in a real cotton field. The increased time to pick each boll resulted from the cotton plants' overlapping leaves and branches obstructing the manipulator. Cotton boll detachment was much more difficult because the bolls were not placed artificially (as done in the

potted plants), and the level of uphill and downhill movement resulted in poor control of the hydraulic navigation system.

The overall goal of the harvesting rover was to develop a system to harvest cotton that uses multiple, small harvesting units per field. These units would be deployed throughout the harvest season, beginning right after the opening of the first cotton bolls. If this team of harvesting rovers is to reach commercial viability, the speed of harvest (CT) and successful removal of bolls (ASR) must be improved. To address these shortcomings, a modified end-effector and an extra upward-looking camera at the bottom of the manipulator in undefoliated cotton will be studied in future studies of the prototype. Furthermore, future research will be conducted to improve the rover's overall navigation to improve the rover's and manipulator's alignment with pickable bolls.

CHAPTER 7

CONCLUSIONS, LIMITATIONS AND FUTURE WORK

7.1 Conclusions and Limitations

In this dissertation, a cotton harvesting rover with robotic arm was successfully demonstrated. It is among the first autonomous cotton harvesting robots developed and tested in the field. The performance and accuracy of the robot were satisfactory, but it did not reach the target speed of 1 boll per 3 seconds. It is due to the hydrostatic engine platform that was used in this research. It was the platform that was available for the time we started our research, and hence, we had to develop algorithms to control it optimally.

Furthermore, there are a few limitations for the systems and methodologies as follows:

1. The hydrostatic transmission used is a difficult technology to obtain precise control of the robot's position, especially in fields with sloping terrain.
2. The stereo camera system from above the canopy alone was not satisfactory. The camera was not able to see all the bolls since they were occluded by crop branches and leaves.
3. The Cartesian system is very efficient, but it had a tough time reaching cotton bolls behind the stems and branches.
4. The stereo camera system was satisfactory to do 3D location estimation of the cotton bolls, but it was only accurate when the rover was moving slowly (0.64 kph) or when stationary.

5. The robot platform was a very large vehicle, and it was challenging to control it precisely. Probably, a smaller platform will provide a sound system for cotton harvesting
6. The robotic system developed was not able to harvest as it moves due to low image processing speed. Future harvesting system should be able to harvest while moving and with multiple robotic arms to harvest a large number of bolls.

7.2 Future Work

Experimental results in this dissertation demonstrated that the robotic harvesting of cotton bolls is a possible technology, and the time to develop such a system is now. The developed algorithms for cotton sensing, localization, navigation, and manipulation can be improved by applying emerging technologies such as advanced robotics and reinforcement learning algorithms. Due to time and scope limitations of this dissertation, future studies are expected to address several issues in modern cotton harvesting such as:

1. More experiments to validate the developed robotic system in harvesting the cotton bolls in the field under direct sunlight should be done.
2. More advanced algorithms to control the hydrostatic transmission rover should be developed. It will include using hydraulic brakes to help stop the rover at an appropriate position because swashplate angle control precision is challenging.
3. The braking system of the hydrostatic system should be integrated to improve the PID control of the rover.
4. The integration of the electric system to control the harvester movement should be developed. The electric system would provide the capability to control the position of the robot and increase the precision and speed of harvesting.

5. More sensors that can be integrated, such as more cameras, vibration sensors, encoders for rear tires, low-cost RTK-GPS, and others to increase the accuracy of boll mapping, should be studied.
6. Multiple camera systems with multiple viewpoints to increase the detection of the cotton bolls and estimate 3D position in real-time should be developed.
7. Multi-modal imaging system to improve the detection of the rows and cotton bolls will provide improved navigation and harvesting accuracy to the robotic system
8. A multi-manipulator system to improve the picking of the cotton bolls should be developed to improve the speed of picking to 1 boll every 3 seconds.
9. A cotton flower mapping system can be developed to provide a prior prescription map and position of the bolls for the robotic system to harvest.

REFERENCES

Antille, D. L., Bennett, J. M., & Jensen, T. A. (2016). Soil compaction and controlled traffic considerations in Australian cotton-farming systems. *Crop and Pasture Science, 67*(1), 1-28.

ASABE. (2019). Coming soon to an orchard near you: The Global Unmanned Spray System (GUSS). *Resource Magazine, 26,* 9-10.

Auat Cheein, F., Steiner, G., Perez Paina, G., & Carelli, R. (2011). Optimized EIF-SLAM algorithm for precision agriculture mapping based on stems detection. *Comput. Electron. Agric., 78*(2), 195-207. doi:10.1016/j.compag.2011.07.007

Babenko, B., Yang, M.-H., & Belongie, S. (2009). *Visual tracking with online multiple instance learning.* IEEE. CVPR.

Bac, C. W., Hemming, J., van Tuijl, B. A. J., Barth, R., Wais, E., & van Henten, E. J. (2017). Performance Evaluation of a Harvesting Robot for Sweet Pepper. *Journal of Field Robotics, 34*(6), 1123-1139. doi:10.1002/rob.21709

Bac, C. W., van Henten, E. J., Hemming, J., & Edan, Y. (2014). Harvesting robots for high-value crops: State-of-the-art review and challenges ahead. *Journal of Field Robotics, 31*(6), 888-911.

Backman, J., Oksanen, T., & Visala, A. (2012). Navigation system for agricultural machines: Nonlinear model predictive path tracking. *Comput. Electron. Agric., 82*, 32-43.

Bak, T., & Jakobsen, H. (2004). Agricultural robotic platform with four wheel steering for weed detection. *Biosystems Engineering, 87*(2), 125-136.

Bakker, T., Bontsema, J., & Müller, J. (2010). Systematic design of an autonomous platform for robotic weeding. *Journal of Terramechanics, 47*(2), 63-73.

Bakker, T., van Asselt, K., Bontsema, J., Müller, J., & van Straten, G. (2006, 2006). *An Autonomous Weeding Robot for Organic Farming.* Field and Service Robotics, Berlin, Heidelberg.

Ball, D., Upcroft, B., Wyeth, G., Corke, P., English, A., Ross, P., . . . Bate, A. (2016). Vision-based obstacle detection and navigation for an agricultural robot. *Journal of field robotics, 33*(8), 1107-1130.

Bechar, A., & Vigneault, C. (2016). Agricultural robots for field operations: Concepts and components. *Biosystems Engineering, 149*, 94-111.

Bechar, A., & Vigneault, C. (2017). Agricultural robots for field operations. Part 2: Operations and systems. *Biosystems engineering, 153*, 110-128.

Bergerman, M., Billingsley, J., Reid, J., & van Henten, E. (2016). Robotics in Agriculture and Forestry. In S. B. & K. O. (Eds.), *Springer Handbook of Robotics* (pp. 1065-1077): Springer,Charm.

Bloch, V., Bechar, A., & Degani, A. (2017). Development of an environment characterization methodology for optimal design of an agricultural robot. *Industr. Rob., 44*(1), 94-103.

Bolme, D. S., Beveridge, J. R., Draper, B. A., & Lui., Y. M. (2010). *Visual object tracking using adaptive correlation filters.* IEEE. CVPR.

Boman, R. (2012). Estimating Cotton Yield Using Boll Counting In O. S. R. a. E. Center (Ed.), *cotton.okstate.edu*. Altus, Oklahoma: OSU South. Res. Ext.

Botterill, T., Paulin, S., Green, R., Williams, S., Lin, J., Saxton, V., . . . Corbett-Davies, S. (2017). A Robot System for Pruning Grape Vines. *Journal of Field Robotics, 34*(6), 1100-1122. doi:10.1002/rob.21680

Boubin, J., Chumley, J., Stewart, C., & Khanal, S. (2019). *Autonomic computing challenges in fully autonomous precision agriculture*.

Bulanon, D. M., Kataoka, T., Okamoto, H., & Hata, S.-i. (2004). *Development of a real-time machine vision system for the apple harvesting robot.* SICE 2004 annual conference, Sapporo, Japan.

Burnard, T. (2017). The American South and its global commodities. *Slavery & Abolition, 38*(1), 215-217. doi:10.1080/0144039X.2017.1284458

Burud, I., Lange, G., Lillemo, M., Bleken, E., Grimstad, L., & From, P. J. (2017). Exploring robots and UAVs as phenotyping tools in plant breeding. *IFAC-PapersOnLine, 50*(1), 11479-11484.

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010, 2010). *Brief: Binary robust independent elementary features.* The 11th European Conference on Computer Vision Crete, Greece.

Cantelli, L., Bonaccorso, F., Longo, D., Melita, C. D., Schillaci, G., & Muscato, G. (2019). A Small Versatile Electrical Robot for Autonomous Spraying in Agriculture. *AgriEngineering, 1*(3), 391-402.

Cao, P. M., Hall, E. L., & Zhang, E. (2003). *Soil sampling sensor system on a mobile robot.* Intelligent Robots and Computer Vision XXI: Algorithms, Techniques, and Active Vision, Providence, RI, United States.

Chang, Y., Zaman, Q., Schumann, A., Percival, D., Esau, T., & Ayalew, G. (2012). Development of color co-occurrence matrix based machine vision algorithms for wild blueberry fields. *Applied engineering in agriculture, 28*(3), 315-323.

Cheein, F. A. A., Carelli, R., Cruz, C. D. l., & Bastos-Filho, T. F. (2010, 14-17 March 2010). *SLAM-based turning strategy in restricted environments for car-like mobile robots.* 2010 IEEE International Conference on Industrial Technology, Vina del Mar, Chile.

Chen, W., Xu, T., Liu, J., Wang, M., & Zhao, D. (2019). Picking Robot Visual Servo Control Based on Modified Fuzzy Neural Network Sliding Mode Algorithms. *Electronics, 8*(6). doi:10.3390/electronics8060605

Cheng, H.-D., Jiang, X. H., Sun, Y., & Wang, J. (2001). Color image segmentation: advances and prospects. *Pattern. Recog, 34*(12), 2259-2281.

Cho, S. I., Chang, S. J., Kim, Y. Y., & An, K. J. (2002). AE—Automation and Emerging Technologies. *Biosystems Engineering, 82*(2), 143-149. doi:10.1006/bioe.2002.0061

Choi, D., Lee, W. S., Ehsani, R., & Roka, F. M. (2015). A machine vision system for quantification of citrus fruit dropped on the ground under the canopy. *Trans. ASABE, 58*(4), 933-946.

Choi, D., Lee, W. S., Ehsani, R., Schueller, J., & Roka, F. M. (2016). Detection of dropped citrus fruit on the ground and evaluation of decay stages in varying illumination conditions. *Comput. Electron. Agric., 127*, 109-119.

Choi, D., Lee, W. S., Schueller, J. K., Ehsani, R., Roka, F., & Diamond, J. (2017). *A performance comparison of RGB, NIR, and depth images in immature citrus detection using deep learning algorithms for yield prediction*. 2017 ASABE Annual International Meeting, St. Joseph, MI. http://elibrary.asabe.org/abstract.asp?aid=48371&t=5

Comba, L., Gay, P., Piccarolo, P., & Ricauda Aimonino, D. (2010). *Robotics and automation for crop management: trends and perspective.* International Conference Ragusa SHWA2010, Ragusa Ibla Campus, Italy.

Corke, P. I., & Ridley, P. (2001). Steering kinematics for a center-articulated mobile robot. *IEEE Transactions on Robotics and Automation, 17*(2), 215-218.

Coulter, R. C. (1992). *Implementation of the pure pursuit path tracking algorithm*. Retrieved from

Cubero, S., Aleixos, N., Moltó, E., Gómez-Sanchis, J., & Blasco, J. (2011). Advances in Machine Vision Applications for Automatic Inspection and Quality Evaluation of Fruits and Vegetables. *Food and Bioprocess Technology, 4*(4), 487-504. doi:10.1007/s11947-010-0411-8

Davies, E., Garlow, A., Farzan, S., Rogers, J., & Hu, A.-P. (2018). *Tarzan: Design, Prototyping, and Testing of a Wire-Borne Brachiating Robot.* 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain.

Deery, D., Jimenez-Berni, J., Jones, H., Sirault, X., & Furbank, R. (2014). Proximal Remote Sensing Buggies and Potential Applications for Field-Based Phenotyping. *Agronomy, 4*(3), 349-379. doi:10.3390/agronomy4030349

Devang, P. S., Gokul, N. A., Ranjana, M., Swaminathan, S., & Binoy, B. N. (2010). *Autonomous arecanut tree climbing and pruning robot.* 2010 International Conference on Emerging Trends in Robotics and Communication Technologies, Chennai, India.

Dong, J., Burnham, J. G., Boots, B., Rains, G., & Dellaert, F. (2017, 29 May-3 June 2017). *4D crop monitoring: Spatio-temporal reconstruction for agriculture.* 2017 IEEE International Conference on Robotics and Automation (ICRA).

Duckett, T., Pearson, S., Blackmore, S., Grieve, B., Chen, W.-H., Cielniak, G., . . . Fox, C. (2018). Agricultural robotics: the future of robotic agriculture. *arXiv preprint arXiv:1806.06762*.

Farzan, S., Hu, A.-P., Davies, E., & Rogers, J. (2018). *Modeling and control of brachiating robots traversing flexible cables.* 2018 IEEE International Conference on Robotics and Automation (ICRA).

Faverjon, B., & Tournassoud, P. (1987). *A local based approach for path planning of manipulators with a high number of degrees of freedom.* Proceedings. 1987 IEEE International Conference on Robotics and Automation.

Feng, Q., Wang, X., Wang, G., & Li, Z. (2015, 8-10 Aug. 2015). *Design and test of tomatoes harvesting robot.* 2015 IEEE International Conference on Information and Automation.

Feng, Q., Zou, W., Fan, P., Zhang, C., & Wang, X. (2018). Design and test of robotic harvesting system for cherry tomato. *International Journal of Agricultural and Biological Engineering, 11*(1), 96-100.

Fentanes, J. P., Gould, I., Duckett, T., Pearson, S., & Cielniak, G. (2018). 3-D Soil Compaction Mapping Through Kriging-Based Exploration With a Mobile Robot. *IEEE Robotics and Automation Letters, 3*(4), 3066-3072. doi:10.1109/LRA.2018.2849567

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM, 24*(6), 381-395.

Fue, K. G., Porter, W. M., Barnes, E. M., & Rains, G. C. (2019a). *Visual Inverse Kinematics for Cotton Picking Robot*. 2019 Beltwide Cotton Conferences, New Orleans, LA. http://www.cotton.org/beltwide/proceedings/2005-2019/data/conferences/2019/papers/19379.pdf

Fue, K. G., Porter, W. M., Barnes, E. M., & Rains, G. C. (2019b). *Visual Row Detection Using Pixel-Based Algorithm and Stereo Camera for Cotton Picking Robot*. 2018 Beltwide Cotton Conferences, New Orleans, LA. http://www.cotton.org/beltwide/proceedings/2005-2019/data/conferences/2019/papers/19376.pdf

Fue, K. G., Porter, W. M., Barnes, E. M., & Rains, G. C. (2020a). An Extensive Review of Mobile Agricultural Robotics for Field Operations: Focus on Cotton Harvesting. *AgriEngineering, 2*(1), 150-174.

Fue, K. G., Porter, W. M., Barnes, E. M., & Rains, G. C. (2020b). An Extensive Review of Mobile Agricultural Robotics for Field Operations: Focus on Cotton Harvesting. *AgriEngineering, 2*(1), 150-174. doi:https://doi.org/10.3390/agriengineering2010010

Fue, K. G., Porter, W. M., & Rains, G. C. (2018a). *Deep Learning based Real-time GPU-accelerated Tracking and Counting of Cotton Bolls under Field Conditions using a Moving Camera*. 2018 ASABE Annual International Meeting, St. Joseph, MI. http://elibrary.asabe.org/abstract.asp?aid=49298&t=5

Fue, K. G., Porter, W. M., & Rains, G. C. (2018b). *Real-Time 3D Measurement of Cotton Boll Positions Using Machine Vision Under Field Conditions*. 2018 BWCC, San Antonio, TX. http://www.cotton.org/beltwide/proceedings/2005-2019/index.htm

García-Santillán, I., Guerrero, J. M., Montalvo, M., & Pajares, G. (2018). Curved and straight crop row detection by accumulation of green pixels from images in maize fields. *Precision Agriculture, 19*(1), 18-41.

Gauch, J. M., & Hsia, C. W. (1992). *Comparison of three-color image segmentation algorithms in four color spaces.* Visua. Comm. Imag. Proc., Bellingham, WA.

Gaus, C.-C., Urso, L.-M., Minßen, T.-F., & de Witte, T. (2017). *Economics of mechanical weeding by a swarm of small field robots*. Retrieved from

Ghaffarzadeh, K. (2019). *Agricultural Robots and Drones 2018-2038: Technologies, Markets and Players*. Retrieved from https://www.idtechex.com/en/research-report/agricultural-robots-and-drones-2018-2038-technologies-markets-and-players/578

Gong, Y., & Sakauchi, M. (1995). Detection of regions matching specified chromatic features. *Comput.Vis. Image Under., 61*(2), 263-269.

Grabner, H., Grabner, M., & Bischof, H. (2006). *Real-time tracking via on-line boosting.* BMVC Durham, England.

Grimstad, L., & From, P. J. (2017). The Thorvald II Agricultural Robotic System. *Robotics, 6*(4), 24.

Grötschel, M., Lovász, L., & Schrijver, A. (2012). *Geometric algorithms and combinatorial optimization* (Vol. 2): Springer Science & Business Media.

Haibo, L., Shuliang, D., Zunmin, L., & Chuijie, Y. (2015). Study and Experiment on a Wheat Precision Seeding Robot. *Journal of Robotics, 2015*, 1-9. doi:10.1155/2015/696301

Hannan, M. W., Burks, T. F., & Bulanon, D. M. (2007). *A Real-time Machine Vision Algorithm for Robotic Citrus Harvesting*. 2007 ASAE Annual Meeting, St. Joseph, MI. http://elibrary.asabe.org/abstract.asp?aid=23429&t=5

Haruhisa, K., Suguru, M., Hideki, K., & Satoshi, U. (2008). *Novel climbing method of pruning robot.* 2008 SICE Annual Conference, Tokyo, Japan.

Hayashi, S., Yamamoto, S., Saito, S., Ochiai, Y., Kamata, J., Kurita, M., & Yamamoto, K. (2014). Field operation of a movable strawberry-harvesting robot using a travel platform. *Japan Agricultural Research Quarterly: JARQ, 48*(3), 307-316.

Hayes, L. (2017). Those Cotton Picking Robots. Retrieved from http://georgia.growingamerica.com/features/2017/08/those-cotton-picking-robots/

Held, D., Thrun, S., & Savarese, S. (2016). *Learning to track at 100 fps with deep regression networks.* ECCV, Berlin, Germany.

Henriques, J. F., Caseiro, R., Martins, P., & Batista, J. (2012). *Exploiting the circulant structure of tracking-by-detection with kernels.* ECCV, Berlin, Heidelberg.

Higuti, V. A. H., Velasquez, A. E. B., Magalhaes, D. V., Becker, M., & Chowdhary, G. (2019). Under canopy light detection and ranging-based autonomous navigation. *Journal of Field Robotics, 36*(3), 547-567. doi:10.1002/rob.21852

Hohimer, C. J., Wang, H., Bhusal, S., Miller, J., Mo, C., & Karkee, M. (2019). Design and Field Evaluation of a Robotic Apple Harvesting System with a 3D-Printed Soft-Robotic End-Effector. *Trans. ASABE, 62*(2), 405-414. doi:https://doi.org/10.13031/trans.12986

Iida, M., Kang, D., Taniwaki, M., Tanaka, M., & Umeda, M. (2008). Localization of CO2 source by a hexapod robot equipped with an anemoscope and a gas sensor. *Comput. Electron. Agric., 63*(1), 73-80.

Jaulin, L., & Godon, A. (1999). *Motion planning using interval analysis.* MISC.

Jensen, M. A. F., Bochtis, D., Sørensen, C. G., Blas, M. R., Lykkegaard, K. L. J. C., & Engineering, I. (2012). In-field and inter-field path planning for agricultural transport units. *63*(4), 1054-1061.

Jiang, Y., Li, C., & Paterson, A. H. (2016). High throughput phenotyping of cotton plant height using depth images under field conditions. *Comput. Electron. Agric., 130*, 57-68.

Kalal, Z., Mikolajczyk, K., & Matas, J. (2010). *Forward-backward error: Automatic detection of tracking failures.* 20th CVPR.

Kalal, Z., Mikolajczyk, K., & Matas, J. (2011). Tracking-learning-detection. *IEEE. trans. PAMI, 34*(7), 1409-1422.

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Comput. Electron. Agric., 147*, 70-90. doi:10.1016/j.compag.2018.02.016

Kayacan, E., Kayacan, E., Chen, I. M., Ramon, H., & Saeys, W. (2018). On the comparison of model-based and model-free controllers in guidance, navigation and control of agricultural vehicles. In *Type-2 Fuzzy Logic and Systems* (pp. 49-73): Springer.

Kicherer, A., Herzog, K., Pflanz, M., Wieland, M., Rüger, P., Kecke, S., . . . Töpfer, R. (2015). An Automated Field Phenotyping Pipeline for Application in Grapevine Research. *Sensors, 15*(3). doi:10.3390/s150304823

Kim, G., Kim, S., Hong, Y., Han, K., & Lee, S. (2012). *A robot platform for unmanned weeding in a paddy field using sensor fusion.* 2012 IEEE International Conference on Automation Science and Engineering (CASE), Seoul, South Korea.

Kise, M., Zhang, Q., & Más, F. R. (2005). A stereovision-based crop row detection method for tractor-automated guidance. *Biosystems engineering, 90*(4), 357-367.

Kohan, A., Borghaee, A. M., Yazdi, M., Minaei, S., & Sheykhdavudi, M. J. (2011). Robotic harvesting of rosa damascena using stereoscopic machine vision. *World Applied Sciences Journal, 12*(2), 231-237.

Kondo, N. (1991). Study on grape harvesting robot. *IFAC Proceedings Volumes, 24*(11), 243-246.

Kondo, N., & Ting*, K. C. (1998). Robotics for Plant Production. *Artificial Intelligence Review, 12*(1), 227-243. doi:10.1023/A:1006585732197

Koubâa, A. (2017). *Robot Operating System (ROS)* (A. Koubâa Ed. 1 ed.): Springer.

Li, J., Karkee, M., Zhang, Q., Xiao, K., & Feng, T. (2016). Characterizing apple picking patterns for robotic harvesting. *Comput. Electron. Agric., 127*, 633-640. doi:10.1016/j.compag.2016.07.024

Li, Y., Cao, Z., Lu, H., Xiao, Y., Zhu, Y., & Cremers, A. B. (2016). In-field cotton detection via region-based semantic image segmentation. *Comput. Electron. Agric., 127*, 475-486. doi:10.1016/j.compag.2016.07.006

Li, Y., Cao, Z., Xiao, Y., & Cremers, A. B. (2017). DeepCotton: in-field cotton segmentation using deep fully convolutional network. *Electron. Imag., 26*(5), 053028.

Liakos, G. K., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018). Machine Learning in Agriculture: A Review. *Sensors*, *18*(8). doi:10.3390/s18082674

Lili, W., Bo, Z., Jinwei, F., Xiaoan, H., Shu, W., Yashuo, L., . . . Chongfeng, W. (2017). Development of a tomato harvesting robot used in greenhouse. *International Journal of Agricultural and Biological Engineering*, *10*(4), 140-149.

Lowenberg-DeBoer, J., Huang, I. Y., Grigoriadis, V., & Blackmore, S. (2019). Economics of robots and automation in field crop production. *Precision Agriculture*. doi:10.1007/s11119-019-09667-5

Lucas, B. D., & Kanade, T. (1981). *An iterative image registration technique with an application to stereo vision.* 7th international joint conference on Artificial intelligence, San Francisco, CA.

Lukezic, A., Vojir, T., Luka ˇCehovin Zajc, J. M., & Kristan, M. (2017). *Discriminative correlation filter with channel and spatial reliability.* IEEE. CVPR.

Lumelsky, V. (1986, 1986). *Continuous motion planning in unknown environment for a 3D cartesian robot arm.* 1986 IEEE International Conference on Robotics and Automation San Fransisco, CA.

Luo, L., Tang, Y., Zou, X., Ye, M., Feng, W., & Li, G. (2016). Vision-based extraction of spatial information in grape clusters for harvesting robots. *biosystems engineering*, *151*, 90-104.

McKinley, S., & Levine, M. (1998). Cubic spline interpolation. *College of the Redwoods, 45*(1), 1049-1060.

Moghimi, A., Hossein Aghkhani, M., Reza Golzarian, M., Rohani, A., & Yang, C. (2015). *A Robo-vision Algorithm for Automatic Harvesting of Green Bell Pepper*. 2015 ASABE Annual International Meeting, St. Joseph, MI. http://elibrary.asabe.org/abstract.asp?aid=46320&t=5

Monkman, G. J. (1995). Robot Grippers for Use With Fibrous Materials. *The International journal of robotics research, 14*(2), 144-151. doi:10.1177/027836499501400204

Monta, M., Kondo, N., & Ting, K. C. (1998). End-Effectors for Tomato Harvesting Robot. In S. Panigrahi & K. C. Ting (Eds.), *Artificial Intelligence for Biology and Agriculture* (pp. 1-25). Dordrecht: Springer Netherlands.

Moore, T., & Stouch, D. (2016). A generalized extended kalman filter implementation for the robot operating system. In *Intelligent autonomous systems 13* (pp. 335-348): Springer.

Mousazadeh, H. (2013). A technical review on navigation systems of agricultural autonomous off-road vehicles. *Journal of Terramechanics, 50*(3), 211-232.

Mu, L., Liu, Y., Cui, Y., Liu, H., Chen, L., Fu, L., & Gejima, Y. (2017). *Design of End-effector for Kiwifruit Harvesting Robot Experiment*. 2017 ASABE Annual International Meeting, St. Joseph, MI. http://elibrary.asabe.org/abstract.asp?aid=47996&t=5

Muja, M., & Lowe, D. G. (2014). Scalable Nearest Neighbor Algorithms for High Dimensional Data. *IEEE. trans. PAMI, 36*(11), 2227-2240. doi:10.1109/TPAMI.2014.2321376

Mulan, W., Jieding, W., Jianning, Y., & Kaiyun, X. (2008). *A research for intelligent cotton picking robot based on machine vision.* 2008 International Conference on Information and Automation, Changsha, China.

Nakao, N., Suzuki, H., Kitajima, T., Kuwahara, A., & Yasuno, T. (2017). Path planning and traveling control for pesticide-spraying robot in greenhouse. *Journal of Signal Processing, 21*(4), 175-178.

Noguchi, N., & Terao, H. (1997). Path planning of an agricultural mobile robot by neural network and genetic algorithm. *Comput. Electron. Agric., 18*(2-3), 187-204.

Oberhammer, J., Tang, M., Liu, A.-Q., & Stemme, G. (2006). Mechanically tri-stable, true single-pole-double-throw (SPDT) switches. *Journal of Micromechanics and Microengineering, 16*(11), 2251.

Obregón, D., Arnau, R., Campo-Cossio, M., Arroyo-Parras, J. G., Pattinson, M., Tiwari, S., . . . Reyes, J. (2019, 2019//). *Precise Positioning and Heading for Autonomous Scouting Robots in a Harsh Environment.* From Bioinspired Systems and Biomedical Applications to Machine Learning, Cham.

Ouadah, N., Ourak, L., & Boudjema, F. (2008). Car-Like Mobile Robot Oriented Positioning by Fuzzy Controllers. *International Journal of Advanced Robotic Systems, 5*(3), 25. doi:10.5772/5603

Paul, R. P. (1981). *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*. Cambridge, Massachusetts and London, England: The MIT Press.

Pedersen, S. M., Fountas, S., & Blackmore, S. (2008). Agricultural robots—Applications and economic perspectives. In *Service robot applications*: IntechOpen.

Post, M. A., Bianco, A., & Yan, X. T. (2017). *Autonomous navigation with ROS for a mobile robot in agricultural fields*. 14th International Conference on Informatics in Control, Automation and Robotics (ICINCO), Universidad Rey Juan Carlos. https://strathprints.strath.ac.uk/61247/

Powers, D. M. W. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness & correlation. *Journal of Machine Learning Technologies, 2*(1), 37-63.

Prostko, E., Lemon, R., & Cothren, T. (2018). *Field Estimation of Cotton Yields*. Retrieved from College Station, TX: http://publications.tamu.edu/COTTON/PUB_cotton_Field%20Estimation%20of%20Cotton%20Yields.pdf

Qixin, C., Yanwen, H., & Jingliang, Z. (2006). *An evolutionary artificial potential field algorithm for dynamic path planning of mobile robot.* 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., . . . Ng, A. Y. (2009, 2009). *ROS: an open-source Robot Operating System.*

Qureshi, W., Payne, A., Walsh, K., Linker, R., Cohen, O., & Dailey, M. (2017). Machine vision for counting fruit on mango tree canopies. *Precision Agriculture, 18*(2), 224-244.

Rahaman, M. M., Chen, D., Gillani, Z., Klukas, C., & Chen, M. (2015). Advanced phenotyping and phenotype data analysis for the study of plant growth and development. *Frontiers in Plant Science, 6*, 619.

Rains, G. C., Bazemore, B. W., Ahlin, K., Hu, A.-P., Sadegh, N., & McMurray, G. (2015). *Steps towards an Autonomous Field Scout and Sampling System.* 2015 ASABE Annual International Meeting.

Rains, G. C., Faircloth, A. G., Thai, C., & Raper, R. L. (2014). Evaluation of a simple pure pursuit path-following algorithm for an autonomous, articulated-steer vehicle. *Applied engineering in agriculture, 30*(3), 367-374.

Ramin Shamshiri, R., Weltzien, C., A. Hameed, I., J. Yule, I., E. Grift, T., K. Balasundram, S., . . . Chowdhary, G. (2018). Research and development in agricultural robotics: A perspective of digital farming. *International Journal of Agricultural and Biological Engineering*, *11*(4), 1-11. doi:10.25165/j.ijabe.20181104.4278

Rao, U. S. N. (2013, 2013). *Design of automatic cotton picking robot with Machine vision using Image Processing algorithms.* 2013 International Conference on Control, Automation, Robotics and Embedded Systems (CARE), Jabalpur, MP, India.

Redmon, J. (Producer). (2016). Darknet: Open Source Neural Networks in C. *Darknet*. Retrieved from https://pjreddie.com/darknet/

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). *You only look once: Unified, real-time object detection.* IEEE. CVPR, Piscataway, NJ.

Redmon, J., & Farhadi, A. (2016). *YOLO9000: better, faster, stronger*. IEEE. CVPR.

Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Reiser, D., Sehsah, E.-S., Bumann, O., Morhard, J., & Griepentrog, W. H. (2019). Development of an Autonomous Electric Robot Implement for Intra-Row Weeding in Vineyards. *Agriculture, 9*(1). doi:10.3390/agriculture9010018

Ritchie, G. L., Bednarz, C. W., Jost, P. H., & Brown, S. M. (2007). Cotton growth and development. In U. o. Georgia (Ed.): University of Georgia.

Rodríguez, F., Moreno, J. C., Sánchez, J. A., & Berenguel, M. (2013). Grasping in Agriculture: State-of-the-Art and Main Characteristics. In G. Carbone (Ed.), *Grasping in Robotics* (pp. 385-409). London: Springer London.

Roldán, J. J., del Cerro, J., Garzón-Ramos, D., Garcia-Aunon, P., Garzón, M., de León, J., & Barrientos, A. (2018). Robots in agriculture: State of art and practical experiences. *Service Robots*.

Romeo, J., Pajares, G., Montalvo, M., Guerrero, J. M., Guijarro, M., & Ribeiro, A. (2012). Crop row detection in maize fields inspired on the human visual perception. *The Scientific World Journal, 2012*.

Rosten, E., & Drummond, T. (2006, 2006). *Machine learning for high-speed corner detection.* European Conference on Computer Vision, Berlin, Heidelberg.

Rovira-Más, F., Zhang, Q., Reid, J., & Will, J. (2005). *Hough-transform-based vision algorithm for crop row detection of an automated agricultural vehicle.* Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). *ORB: An efficient alternative to SIFT or SURF*. 2011 International conference on computer vision, Barcelona, Spain.

Safren, O., Alchanatis, V., Ostrovsky, V., & Levi, O. (2007). Detection of Green Apples in Hyperspectral Images of Apple-Tree Foliage Using Machine Vision. *Trans. ASABE, 50*(6), 2303-2313. doi:https://doi.org/10.13031/2013.24083

Salas Fernandez, M. G., Bao, Y., Tang, L., & Schnable, P. S. (2017). A High-Throughput, Field-Based Phenotyping Technology for Tall Biomass Crops. *Plant physiology, 174*(4), 2008-2022. doi:10.1104/pp.17.00707

Sammons, P. J., Furukawa, T., & Bulgin, A. (2005, December 5 – 7, 2005). *Autonomous pesticide spraying robot for use in a greenhouse.* 2005 Australasian Conference on Robotics and Automation Sydney, Australia.

Samuel, M., Hussein, M., & Mohamad, M. B. (2016). A review of some pure-pursuit based path tracking techniques for control of autonomous vehicle. *International Journal of Computer Applications, 135*(1), 35-38.

Sankaran, S., Khot, L. R., Espinoza, C. Z., Jarolmasjed, S., Sathuvalli, V. R., Vandemark, G. J., . . . Pavek, M. J. (2015). Low-altitude, high-resolution aerial imaging systems for row and field crop phenotyping: A review. *European Journal of Agronomy, 70*, 112-123. doi:10.1016/j.eja.2015.07.004

Scholz, C., Moeller, K., Ruckelshausen, A., Hinck, S., & Goettinger, M. (2014). *Automatic soil penetrometer measurements and GIS based documentation with the autonomous field robot platform bonirob.* 12th International Conference of Precision Agriculture.

Sengupta, S., & Lee, W. S. (2014). Identification and determination of the number of immature green citrus fruit in a canopy under different ambient light conditions. *117*, 51-61. doi:10.1016/j.biosystemseng.2013.07.007

Sharma, S., & Borse, R. (2016, 2016//). *Automatic Agriculture Spraying Robot with Smart Decision Making.* Intelligent Systems Technologies and Applications 2016, Cham.

Shi, J., & Tomasi, C. (1994). *Good features to track.* IEEE. CVPR, Seattle, WA.

Shockley, J. M., & Dillon, C. R. (2018, 2018). *An economic feasibility assessment for adoption of autonomous field machinery in row crop production.* 2018 international conference on precision agriculture, Montreal, QC, Canada.

Shvalb, N., Moshe, B. B., & Medina, O. (2013). A real-time motion planning algorithm for a hyper-redundant set of mechanisms. *Robotica, 31*(8), 1327-1335. doi:10.1017/S0263574713000489

Silwal, A., Davidson, J. R., Karkee, M., Mo, C., Zhang, Q., & Lewis, K. (2017). Design, integration, and field evaluation of a robotic apple harvester. *Journal of Field Robotics, 34*(6), 1140-1159. doi:10.1002/rob.21715

Sokolova, M., Japkowicz, N., & Szpakowicz, S. (2006). *Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation.* Austr. Conf. Arti. Intel, Berlin, Heidelberg.

Srinivasan, N., Prabhu, P., Smruthi, S. S., Sivaraman, N. V., Gladwin, S. J., Rajavel, R., & Natarajan, A. R. (2016, 21-23 Dec. 2016). *Design of an autonomous seed planting robot.* 2016 IEEE Region 10 Humanitarian Technology Conference (R10-HTC).

Subramanian, V., Burks, T. F., & Arroyo, A. A. (2006). Development of machine vision and laser radar based autonomous vehicle guidance systems for citrus grove navigation. *Comput. Electron. Agric., 53*(2), 130-143.

Sun, S., Li, C., & Paterson, H. A. (2017). In-Field High-Throughput Phenotyping of Cotton Plant Height Using LiDAR. *Remote Sensing, 9*(4). doi:10.3390/rs9040377

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2015). *Going deeper with convolutions.* Proceedings of the IEEE conference on computer vision and pattern recognition.

Tai, K., El-Sayed, A.-R., Shahriari, M., Biglarbegian, M., & Mahmud, S. (2016). State of the Art Robotic Grippers and Applications. *Robotics, 5*(2). doi:10.3390/robotics5020011

Tao, Y., H. Heinemann, P., Varghese, Z., T. Morrow, C., & J. Sommer Iii, H. (1995). Machine Vision for Color Inspection of Potatoes and Apples. *Transactions of the ASAE, 38*(5), 1555-1561. doi:https://doi.org/10.13031/2013.27982

Trieu, T. H. (2018). Darkflow. *Darkflow*. Retrieved from https://github.com/thtrieu/darkflow

Tu, X., Gai, J., & Tang, L. (2019). Robust navigation control of a 4WD/4WS agricultural robotic vehicle. *Comput. Electron. Agric., 164*, 104892.

Ueki, S., Kawasaki, H., Ishigure, Y., Koganemaru, K., & Mori, Y. (2011). Development and experimental study of a novel pruning robot. *Artificial Life and Robotics, 16*(1), 86-89. doi:10.1007/s10015-011-0892-1

UGA (Producer). (2018). 2018 Georgia cotton production guide. *UGA Cotton Web Page*. Retrieved from http://www.ugacotton.com/production-guide

UGA. (2019). Georgia cotton production guide. In U. E. Team (Ed.), *ugacotton.org*. Tifton, GA: UGA Extension Team.

USDA/NASS. (2018). *2017 State Agriculture Overview for Georgia*. Retrieved from https://www.nass.usda.gov/Quick_Stats/Ag_Overview/stateOverview.php?state=GEORGIA

Van Henten, E. J., Van Tuijl, B. A. J., Hemming, J., Kornet, J. G., Bontsema, J., & Van Os, E. A. (2003). Field Test of an Autonomous Cucumber Picking Robot. *Biosystems Engineering, 86*(3), 305-313. doi:10.1016/j.biosystemseng.2003.08.002

Waldman, P., & Mulvany, L. (2020). Farmers Fight John Deere Over Who Gets to Fix an $800,000 Tractor. *Bloomberg*. Retrieved from https://www.bloomberg.com/news/features/2020-03-05/farmers-fight-john-deere-over-who-gets-to-fix-an-800-000-tractor

Wan, E. A., & Nelson, A. T. (2001). Dual extended Kalman filter methods. *Kalman filtering and neural networks, 123*.

Wang, C., Lee, W. S., Zou, X., Choi, D., Gan, H., & Diamond, J. (2018). Detection and counting of immature green citrus fruit based on the Local Binary Patterns (LBP) feature using illumination-normalized images. *Precision Agriculture, 19*(6), 1062-1083. doi:10.1007/s11119-018-9574-5

Wang, H., & Noguchi, N. (2018). Adaptive turning control for an agricultural robot tractor. *International Journal of Agricultural and Biological Engineering, 11*(6), 113-119.

Wang, N., Zhang, N., & Wang, M. (2006). Wireless sensors in agriculture and food industry—Recent development and future perspective. *Comput. Electron. Agric., 50*(1), 1-14. doi:10.1016/j.compag.2005.09.003

Wang, Y., Zhu, X., & Ji, C. (2008, 2008//). *Machine Vision Based Cotton Recognition for Cotton Harvesting Robot.* Computer And Computing Technologies In Agriculture, Volume II, Boston, MA.

Winterhalter, W., Fleckenstein, F. V., Dornhege, C., & Burgard, W. (2018). Crop row detection on tiny plants with the pattern hough transform. *IEEE Robotics and Automation Letters, 3*(4), 3394-3401.

Xiong, Y., Peng, C., Grimstad, L., From, P. J., & Isler, V. (2019). Development and field evaluation of a strawberry harvesting robot with a cable-driven gripper. *Comput. Electron. Agric., 157*, 392-402. doi:10.1016/j.compag.2019.01.009

Xu, S., Wu, J., Zhu, L., Li, W., Wang, Y., & Wang, N. (2015). *A novel monocular visual navigation method for cotton-picking robot based on horizontal spline segmentation.* MIPPR 2015: Automatic Target Recognition and Navigation, Enshi, China.

Xue, J., Zhang, L., & Grift, T. E. (2012). Variable field-of-view machine vision based row guidance of an agricultural robot. *Comput. Electron. Agric., 84*, 85-91. doi:10.1016/j.compag.2012.02.009

Yaguchi, H., Nagahama, K., Hasegawa, T., & Inaba, M. (2016, 9-14 Oct. 2016). *Development of an autonomous tomato harvesting robot with rotational plucking gripper.* 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, South Korea.

Young, S. N., Kayacan, E., & Peschel, J. M. (2018). Design and field evaluation of a ground robot for high-throughput phenotyping of energy sorghum. *Precision Agriculture, 20*(4), 697-722. doi:10.1007/s11119-018-9601-6

Yuanshen, Z., Gong, L., Liu, C., & Huang, Y. (2016). Dual-arm robot design and testing for harvesting tomato in greenhouse. *IFAC-Papers Online, 49*(16), 161-165.

Zahniser, S., Taylor, J. E., Hertz, T., & Charlton, D. (2018). *Farm labor markets in the United States and Mexico Pose challenges for US agriculture*. Retrieved from University of Minnesota Website: https://ageconsearch.umn.edu/record/281161/files/EIB201.pdf

Zefran, M. (1996). Continuous methods for motion planning. *IRCS Technical Reports Series*, 111.

Zeng, J., Ju, R., Qin, L., Hu, Y., Yin, Q., & Hu, C. (2019). Navigation in Unknown Dynamic Environments Based on Deep Reinforcement Learning. *Sensors, 19*(18), 3837.

Zhai, Z., Zhu, Z., Du, Y., Song, Z., & Mao, E. (2016). Multi-crop-row detection algorithm based on binocular vision. *Biosystems Engineering, 150*, 89-103.

Zhao, Y., Gong, L., Huang, Y., & Liu, C. (2016). Robust tomato recognition for robotic harvesting using feature images fusion. *Sensors, 16*(2), 173.

Zion, B., Mann, M., Levin, D., Shilo, A., Rubinstein, D., & Shmulevich, I. (2014). Harvest-order planning for a multiarm robotic harvester. *Comput. Electron. Agric., 103*, 75-81. doi:10.1016/j.compag.2014.02.008

Zuo, G., Zhang, P., & Qiao, J. (2010, 2010). *Path planning algorithm based on sub-region for agricultural robot.* 2nd International Asia Conference on Informatics in Control, Automation and Robotics.